Actes du Septième Colloque sur l'Optimisation et les Systèmes d'Information - COSI'2010

18-20 Avril 2010, Ouargla, Algérie

Université KASDI Merbah

Faculté des Sciences et de la Technologie et Sciences de la Matière-STSM

Département Mathématiques et Informatique

Contents

Préface

Ce volume contient les actes de la septième édition du Colloque international sur l'Optimisation et les Systèmes d'Information (COSI'2010), qui a eu lieu à Ouargla, Algérie, du 18 au 20 Avril 2010. Sept ans est un âge de maturité pour une manifestation scientifique de cette envergure. Au fil du temps, le colloque COSI a acquis une place très particulière dans le paysage scientifique algérien, fruit d'une volonté affirmée de mettre en avant des critères de qualité et d'ouverture et suscitant une implication forte de la part de chercheurs des deux rives de la Méditerranée et au delà. Cette édition est également particulière par sa localisation. Le choix de la ville de Ouargla, au sud algérien, permet de boucler un circuit de conférences qui aura fait le tour de toutes les régions d'Algérie, perpétuant ainsi l'esprit d'ouverture de COSI et sa volonté de contribuer au rapprochement et à la fédération des forces de recherche en Algérie.

Le programme scientifique de COSI'2010 contient la présentation de 47 articles sélectionnés parmi 183 soumissions (soit un taux de sélection inférieur à 26%) ainsi que 5 conférences plénières qui couvrent des thématiques de recherche très actuelles autour de l'optimisation combinatoire, du web sémantique, de l'intégration d'information, des graphes pour les systèmes P2P et de la qualité de service dans les réseaux autonomes. Une innovation dans le programme de cette année réside dans l'organisation d'une session industrielle animée par la société KLS Logistic Systems. Il devient aussi une tradition d'accompagner chaque édition de COSI par une école d'été à destination des jeunes chercheurs. Cette année, une école d'une journée suivra la conférence et portera sur la thématique générale des réseaux informatique.

L'organisation d'un tel événement nécessite un effort important. Tout d'abord, nous souhaitons remercier très vivement les auteurs des soumissions ainsi que les membres du comité de programme qui ont fait un travail remarquable et souvent ingrat. Nous sommes très reconnaissant au Recteur de l'Université Kasdi Merbah de Ouargla, le Professeur Ahmed BOUTARFAIA, d'avoir accepté et soutenu l'organisation de ce colloque au sein de son Université. Nos remerciements les plus vifs vont aux membres du comité d'organisation, et d'une manière très particulière à son président, le Docteur Dris Korichi, véritable pierre angulaire dans l'édifice de l'organisation. Sa très grande disponibilité et son enthousiasme ont facilité grandement la mise en place de cette manifestation. Nous remercions également le Docteur Mourad Baiou, LIMOS-CNRS, qui a pris en charge l'organisation de la session industrielle.

La tenue de ce colloque à Ouargla est due à une proposition et au soutien du Professeur Mohamed Tayeb Laskri. Qu'il reçoit ici notre plus profonde gratitude.

Enfin, nous ne pouvons terminer cette préface sans un remerciement particulier à nos sponsors, Microsoft Research et la société KLS Logistic Systems.

Pr. Farouk Toumani

Président du comité de programme

Organisation

Université KASDI Merbah, Ouargla, Algérie

Président d'honneur

Professeur Ahmed Boutarfaia
Recteur de l'Université KASDI Merbah, Ouargla, Algérie

Comité d'Organisation

Président

Dr. Driss Korichi, Université KASDI Merbah, Ouargla

Membres

Mustafa Acila, Université KASDI Merbah, Ouargla
Kamel Eddine Aiadi, Université KASDI Merbah, Ouargla
Djamel Ahmed Chacha, Université KASDI Merbah, Ouargla
Ahmed Korichi, Université KASDI Merbah, Ouargla
Mabrouk Meflah, Université KASDI Merbah, Ouargla
Redhouane Kafi, Université KASDI Merbah, Ouargla

Comité de Pilotage

Mohamed Aidene, Université Mouloud Mammeri de Tizi-Ouzou, Algérie
Nacéra Benamrane, Université des Sciences et Technologie d'Oran, Algérie
Abdelhafidh Berrachedi, Université des Sciences et Technologie Houari Boumédienne, Alger, Algérie
Mohand-Saïd Hacid, Université de Lyon I, France
Lhouari Nourine, Université de Clermont-Ferrand II, France
Brahim Oukacha, Université de Tizi-Ouzou, Algérie
Jean Marc Petit, INSA de Lyon, France
Bachir Sadi, Université de Tizi-Ouzou, Algérie
Lakhdar Saïs, CRIL - CNRS, Université d'Artois, France
Kamel Tari, Université Abderahmane Mira de Bejaia, Algérie

Comité de Programme

Président

Farouk Toumani, LIMOS, CNRS, Université Blaise Pascal, Clermont-Ferrand

Membres

Alexandre Aussem, Univ Lyon 1
Mohamed Tayeb Laskri, Université dAnnaba (Algérie)
Ahmed Korichi, Université Kasdi Merbah, Ouargla
Jalel Akaichi, ISG, Tunis
Riadh Farah, ENSI,Tunis
Babahenini Mohamed Chaouki, Université Mohamed Khider, Biskra
Rachid Nourine, Université d'Oran
Mohand Ouanes, UMM Tizi-Ouzou
Sadok Ben Yahia, Faculté des Sciences de Tunis
Brahim Oukacha, Université Mouloud Mammeri, Tizi-Ouzou
Rokia Missaoui, Universite de Quebec en Outaouais
Emmanuel Trelat, Université d'Orléans
Fatiha Sais, LRI, Université de Paris Sud
Pierre Spiteri, INP, Toulouse
Hayett Merouani, Université Badji Mokhtar, Annaba
Mokhtar Sellami, Université Badji Mokhtar, Annaba
Bornia Tighiouart, Université Badji Mokhtar, Annaba
Assef Chmeiss, CRIL, Université d'Artois
Belaid Benhamou, LSIS, Marseille
Lakhdar Sais, CRIL
Lhouari Nourine, LIMOS
Mohamed Aidene, Université de Tizi-Ouzou
M. Aider, USTHB (Alger)
H. Ait haddadene, USTHB (Alger)
M. Baiou, Clermont-Ferrand II
Souhila Kaci, CRIL, Université d'Artois, France
H. Belbachir, USTO (Oran)
Nacéra Benamrane, USTO (Oran)
Boualem Benatallah, University of New South Wals (Australie)
Salima Benbernou, Univerdsité Paris Descartes, France
Abdelhafid Berrachedi, USTHB (Alger)
Bertrand Mazure, CRIL, CNRS Université d'ARtois
Isma Bouchemakh, USTHB (Alger)
Korichi Driss, Université Kasdi Merbah, Ouargla
Mohand Ou Idir Khemmoudj, Université Paris 13
Michel Habib, Université de Paris VII (France)
Mohand-Said Hacid, LIRIS, Université de Lyon I (France)
Mephu Engelbert, CRIL - Université d'Artois
Youssef Hamadi, Microsoft Research Cambridge
Philippe Mahey, Université de Clermont-Ferrand II
Frédéric Messine, ENSEEIHT, IRIT Toulouse (France)
Laurent Gourvès, Lamsade, France
Jean-Marc Petit, INSA de Lyon (France)
MS. Radjef, Université de Bejaia
Michel Schneider, Université de Clermont-Ferrand II
Bachir Sadi, UMMTO, Tizi-Ouzou
Tatiana Tchemisova, University of Aveiro, Portugal
H. Kheddouci, Université de Lyon I
Rachid Ahmed-Ouamer, UMMTO, Tiszi-Ouzou
Vincent Barra, Clermont-Ferrand II (France)
Mohand Boughanem, IRIT, Toulouse
BENABDESLEM Khalid, LIESP, Université Lyon 1, France
M.O. Bibi, Université de Béjaia
Mustapha Belaissaoui, Université Mohammed V (Maroc)
Tahar Kechadi, UCD, Irlande
Fouilhoux Pierre, LIP6, Paris
Marie Agier, LIMOS
Marie-Christine Fauvet, Université Joseph Fourier, Grenoble
Allel Hadjali, ENSSAT, Lannion
Alain Leger, France Télécom

Christophe Rey, Université Blaise Pascal, Clermont-Fd
Fethi Rabhi, UNSW, Sydney, Australie
Yassine Salem, Université de Sétif
Hélène Jaudoin, ENSSAT, Lannion, France
Mahmoud Boufaida, Université Mentouri, Constantine
Mohamed Saidi, Université de Sétif
Seridi Hassina, Université Badji Mokhtar, Annaba
SERIDI Hamid, Université 8 Mai 1945, Guelma
Haddadi Salim, Université 8 mai 1945, Guelma
Elghazel Haytham, Université Claude Bernard Lyon 1
KHOLLADI Mohamed-Khireddine, Université Mentouri, Constantine
Bilel Derbel, Université des Sciences et Technologies
Khouloud Boukadi, LIMOS, Université Blaise Pascal
Odile Papini, Université de Toulon
Ladjel BELLATRECHE, LISI, Université de Poitier, France
Abdessamad Imine, Loria, Université Nancy 2, France
Okba Kazar, Université de Biskra
Mohamed Zine Aissaoui, Université 8 Mai 1945, Guelma
Abdelhani BOUKROUCHE, Université 8 Mai 1945, Guelma
Fatima Zohra Laallam, Université Kasdi Merbah Ouargla
Djamel Ahmed CHACHA, Université KASDI Merbah de Ouargla
Mustafa ACILA, Université KASDI Merbah de Ouargla
Karima Sedki, Université de valenciennes
Gouinaud Christophe, LIMOS, Université Blaise Pascal, France

Relecteurs additionnels

Van Munin Chhieng, UNSW, Sydney, Australie
Laurent D'orazio, LIMOS, CNRS, Université Blaise Pascal, Clermont-Ferrand
Thierry GARCIA, IRIT, Toulouse
Adnene Guabtni, UNSW, Sydney, Australie
Said Jabbour, INRIA-Microsoft Rsearch
Tarek Hamrouni, Université Tunis El Manar
Mamadou Kante, Limos, CNRS, Université Blaise Pascal, Clermont-Ferrand
Ali khebizi, Université Badji Mokhtar, Annaba
Philippe Lacomme, LIMOS, CNRS, Université Blaise Pascal, Clermont-Ferrand
Freddy Lécué, University of Manchester
Mondher Maddouri, Faculté des Sciences de Gafsa
Ingo Weber, UNSW, Sydney, Australie
Vincent Limouzy, Limos, CNRS, Université Blaise Pascal, Clermont-Ferrand
Philippe MARTHON, ENSEEIHT, Toulouse

Théorie des graphes

Le nombre de domination par contraction

[1]Kamel Tablennehas et[2]Mustapha Chellali.
[1]Département de Mathématiques, Faculté des Sciences,
Université de Médea. E-mail: Tablennehas1@yahoo.fr
[2]Laboratoire LAMDA-RO Département des Mathématiques,
Université de Blida. E-mail: m_chellali@hotmail.com

Résumé

Soit $G = (V, E)$ un graphe simple. Un sous-ensemble S de V est un dominant de G si tout sommet de $V - S$ est adjacent à au moins un sommet de S. Le cardinal minimum d'un ensemble dominant de G, noté $\gamma(G)$, est appelé nombre de domination. Un ensemble dominant stable d'un graphe G est un ensemble dominant dont le sous-graphe induit est un stable. Le cardinal minimum d'un ensemble dominant stable de G, noté $i(G)$, est appelé nombre de domination stable. Etant donné un paramètre de domination μ d'un graphe G, on défini le nombre de domination par contraction d'un graphe G connexe, noté $Ct\mu(G)$ comme étant le nombre minimum d'arêtes à contracter successivement pour faire diminuer le nombre de domination $\mu(G)$.

Dans [4] Huang et Jun-Ming ont montrés que $Ct\gamma(G) \leq 3$ pour tout graphe G. Dans ce papier, On donne une réponse au problème posé par Huang et Jun-Ming dans l'article [4], en caractérisant les arbres T ayant $Ct_\gamma(T) = 3$. Ensuite, on montre qu'il existe des graphes où le nombre de domination stable par contraction est très grand.

Keywords: domination, domination stable, nombre de domination par contraction.

1 Introduction

On considère un graphe simple connexe $G = (V, E)$ ayant $V(G)$ comme ensemble de sommets et $E(G)$ comme ensemble d'arêtes. Le nombre de sommets $|V(G)|$ dans un graphe G est appelé ordre de G et noté souvent par n. Le voisinage ouvert d'un sommet est $N(v) = \{u \in V/uv \in E\}$, son voisinage fermé est $N[v] = N(v) \cup \{v\}$. Le degré d'un sommet v, noté par

$d_G(v)$ est $|N(v)|$. Un sommet de degré nul est dit isolé. Un sommet de degré un est appelé sommet pendant, et son voisin est dit sommet support.

Le voisinage privé d'un sommet v par rapport à un ensemble S noté $pn[v, S]$ est l'ensemble des sommets du voisinage fermé de v qui n'ont pas d'autres voisins dans S, i-e: $pn[v, S] = \{u : N[u] \cap S = \{v\}\}$.

Soient u et v deux sommets d'un graphe G. On appelle distance entre u et v, notée $d(u, v)$, la longueur de la plus courte chaîne joignant u et v. L'excentricité d'un sommet v dans un graphe $G = (V, E)$ est $exc(v) = \max\{d(v, w) : w \in V\}$ et le diamètre de G, noté $Diam(G)$, est égal à $\max\{exc(v) : v \in V\}$.

Pour un sous-ensemble $S \subset V$, le sous-graphe induit par S noté $G[S]$ est le graphe ayant S comme ensemble de sommets et ses arêtes sont celles de E ayant leurs extrémités dans S.

Pour un-sous ensemble $U \subseteq E$, le graphe partiel de G défini par U noté G_U est le graphe dont les ensembles de sommets et d'arêtes sont respectivement V et U.

Un graphe $G = (V, E)$ est dit $multi - parti$ s'il existe une partition de V en k sous-ensembles $V_1, V_2, ..., V_k$ tels que chacun des $G[V_i]$ ne contient aucune arête (stable). Si $k = 2$ le graphe G est dit *biparti*. On appelle graphe *biparti−complet*, un graphe biparti tel que pour tout sommet $u \in V_1$ et $v \in V_2, uv \in E$. Si $|V_1| = p$ et $|V_2| = q$ alors le graphe biparti complet est noté $K_{p,q}$.

L'*étoile* noté par $K_{1,t}$ est un cas particulier d'un graphe biparti complet tel que $|V_1| = 1$ et $|V_2| = t$

Soit $G = (V, E)$ un graphe simple. Un sous-ensmble $S \subseteq V(G)$ est un ensemble dominant si tout sommet de $V - S$ est adjacent à au moins un sommet de S. Le cardinal minimum d'un ensemble dominant de G est appelé nombre de domination et est noté par $\gamma(G)$.Un ensemble dominant S de cardinal $\gamma(G)$ est appelé $\gamma(G)$-ensemble ou simplement γ-ensemble.

Un sous-ensmble $S \subseteq V(G)$ est un ensemble dominant stable si S est un ensemble dominant et le sous-graphe induit par les sommets de S ne contient pas d'arêtes. Le nombre de domination stable noté par $i(G)$ est le cardinal minimum d'un ensemble dominant stable de G.

Soit $G = (V, E)$ un graphe simple, pour une arête uv de G, on note par G_{uv} le graphe obtenu à partir de G en contractant l'arête uv . On note aussi par $\overline{uv}$ le nouveau sommet obtenu. Donc le graphe G_{uv} est obtenu à partir de G en supprimant les sommets u et v et en ajoutant le sommet $\overline{uv}$ qui est adjacent à tous les sommets adjacents à u ou v.

Dans ce papier on considère l'effet de la contraction d'arêtes sur le nombre de domination $\gamma(G)$. Dans [4], Huang et Jun-Ming ont défini le nom-

bre de domination par contraction d'un graphe $G = (V, E)$ connexe tel que $\gamma(G) \geq 2$, noté par $Ct_\gamma(G)$ comme étant le nombre minimum d'arêtes à contracter successivement pour faire diminuer le nombre de domination $\gamma(G)$.

2 Resultats préliminaires

Nous citons quelques résultats obtenus par Huang et Ming Xu dans [4].

Proposition 1 *[4] Pour les chaînes P_n et les cycles C_n d'ordre $n \geq 4$, $Ct_\gamma(P_n) = Ct_\gamma(C_n) = i$ tel que $n = 3k + i$ et $1 \leq i \leq 3$.*

Huang et Jun-Ming ont montré que $Ct_\gamma(G)$ est inférieur à trois pour tout graphe connexe avec $\gamma(G) \geq 2$.

Théorème 2 *[4] Pour tout graphe G connexe on a : $Ct_\gamma(G) \leq 3$.*

Par la proposition suivante, les graphes ayant $Ct_\gamma(G) = 0$ sont caractérisés.

Proposition 3 *Pour un graphe G connexe, $Ct_\gamma(G) = 0$ si et seulement si G admet une étoile comme sous-graphe partiel.*

Par la proposition suivante, les graphes ayant $Ct_\gamma(G) = 1$ sont caractérisés.

Proposition 4 *[4] Pour un graphe G connexe, $Ct_\gamma(G) = 1$ si et seulement s'il existe un $\gamma(G)$-ensemble D qui n'est pas un stable.*

Il résulte de cette proposition le corollaire suivant:

Corollaire 5 *[4] Pour un graphe G connexe, si $\gamma(G)$=i(G), alors $Ct_\gamma(G)$>1.*

Les graphes G tels que $Ct_\gamma(G) = 2$ sont caractérisés par le résultat suivant:

Proposition 6 *[4] Pour un graphe G connexe, $Ct_\gamma(G) = 2$ si et seulement si tout $\gamma(G)$-ensemble est un stable et il existe un ensemble dominant D de cardinalité $\gamma + 1$ tel que $G[D]$ contient au moins deux arêtes.*

A partir du théorème 2, tous les graphes peuvent être classés en quatre catégories selon leurs nombre de domination par contraction $Ct_\gamma(G)$. On note par $\mathcal{C}_\gamma^i$ les graphes ayant $Ct_\gamma(G) = i$ pour $i = 0, 1, 2, 3$. Et on note par $\mathcal{P}_\gamma^j$ l'ensemble des graphes connexes satisfaisant la propriété j. Si $\mathcal{A}$ est une famille de graphes connexes, alors $\overline{\mathcal{A}}$ représente la famille de graphes connexes qui ne sont pas dans la famille $\mathcal{A}$.

Propriété 1. G admet une étoile comme sous graphe partiel.

Propriété 2. G admet un $\gamma(G)$-ensemble D qui n'est pas un stable.

Propriété 3. G admet un ensemble dominant D de cardinalité $\gamma + 1$ tel que $G[D]$ contient au moins deux arêtes.

Théorème 7 *[4]* $\mathcal{C}_\gamma^0 = \mathcal{P}_\gamma^1$, $\mathcal{C}_\gamma^1 = \mathcal{P}_\gamma^2$, $\mathcal{C}_\gamma^2 = \overline{\mathcal{P}_\gamma^2} \cap \mathcal{P}_\gamma^3$ *et* $\mathcal{C}_\gamma^3 = \overline{\mathcal{P}_\gamma^1} \cap \overline{\mathcal{P}_\gamma^3}$.

3 Caractérisation des arbres tels que $Ct\gamma(G) = 3$

On s'intérresse dans cette partie à donner une réponse au problème posé par Huang et Jun-Ming dans l'article [4] en caractérisant les arbres T ayant $Ct_\gamma(T) = 3$.

Soit $\mathcal{F}$ la famille des arbres T obtenus à partir d'une séquence d'arbres $T_1, T_2, ..., T_k$ avec $k \geq 1$ tel que $T_1 = P_6$ et $T = T_k$. Si $k \geq 2$, T_{i+1} est obtenu à partir de T_i par l'une des opérations suivantes. Soit $\{x, y\}$ l'ensemble des sommets support de l'arbre T_1 et on pose $A(T_1) = \{x, y\}$

Opération θ_1 : Ajouter des sommets pendants à un support z.
Poser $A(T_{i+1}) = A(T_i)$.
Opération θ_2 : Attacher un sommet z de $V(T_i) - A(T_i)$ à un sommet u de la chaîne uvw. Poser $A(T_{i+1}) = A(T_i) \cup \{v\}$.

Par la construction de $\mathcal{F}$, on a l'observation suivante:

Observation 8 *Si $T \in \mathcal{F}$ alors chaque sommet de $A(T)$ admet au moins deux sommets privés dans $V(T) - A(T)$.*

Nous donnons la définition d'un dominant parfait comme suit:

Définition 9 *Un ensemble S est dit un dominant parfait si S est un dominant et chaque sommet de $V - S$ admet exactement un seul voisin dans S.*

Théorème 10 (Bange, Barkaukas et Slater [5]) *Si G admet un dominant parfait alors tous les dominants minimaux ont la même taille $\gamma(G)$.*

Théorème 11 (Gunther, Hartnell, Markus et Rall [6]) .
Un arbre T d'ordre $n \geq 3$ admet un $\gamma(T)$-ensemble unique D si et seulement si chaque sommet de D admet au moins deux voisins privés dans $V - D$.

Lemme 12 *Si T est un arbre tel que $Ct_\gamma(T) = 3$, alors T admet un $\gamma(T)$- ensemble unique et un dominant parfait unique.*

Preuve. Soit S un $\gamma(T)$-ensemble et supposons que $Ct_\gamma(T) = 3$. Il est clair que $G[S]$ est un stable et donc $\gamma(T) = i(T)$. On suppose maintenant que l'ensemble S n'est pas un dominant parfait. Alors il existe un sommet $z \in V - S$ qui admet deux voisins x, y dans S. Notons par w le sommet obtenu à partir de la contraction des deux arétes xz et yz, et soit G' le graphe obtenu après la contraction. Alors l'ensemble $\{w\} \cup (S - \{x, y\})$ est un dominant de G' et $\gamma(G') < \gamma(G)$. Par conséquent $Ct_\gamma(T) = 2$, d'où la contradiction.

Pour l'unicité, soit D un $\gamma(T)$-ensemble. Comme T est un arbre, alors pour tout sommet $v \in D$ on a $N(v) \neq \emptyset$. Puisque $G[D]$ est un stable et si $N(v) = \{u\}$ alors $D' = (D - \{v\}) \cup \{u\}$ est un dominant de T qui n'est pas parfait. Donc chaque sommet $v \in D$ admet au moins deux voisins privés dans $V - D$. D'aprés le théorème 11, D est un dominant parfait unique. ■

Remarque: *Le résultat précédent n'est pas vrai pour tout graphe G, car si $G = C_6$ alors $Ct_\gamma(T) = 3$* et *le graphe G admet un dominant parfait qui n'est pas unique.*

Lemme 13 *Pour un arbre $T \neq k_{1,t}$, l'arbre T admet à la fois un unique $\gamma(T)$-ensemble et un unique dominant parfait si et seulement si $T \in \mathcal{F}$.*

Preuve. ($\Leftarrow$) Soit T un arbre de la famille $\mathcal{F}$. A partir de la construction $A(T)$ est un dominant parfait de T et d'aprés le théorème 10 $A(T)$ est un $\gamma(T)$-ensemble. Puisque chaque sommet de $A(T)$ possède au moins deux sommets privés alors d'aprés le thèorème 11 $A(T)$ est un unique $\gamma(T)$-ensemble. D'ou T admet à la fois un unique $\gamma(T)$-ensemble et un unique dominant parfait.

($\Rightarrow$) Soit D à la fois un unique $\gamma(T)$- ensemble et un unique dominant parfait de T. On utilise l'induction sur le nombre de sommets de T. L'unicité de l'ensemble D implique chaque sommet support est dans D. Aussi dire que D est dominant parfait signifie que la distance entre deux supports est au moins trois. Puisque T n'est pas une étoile alors $diam(T) \geq 5$. Il est clair que $T = P_6$ est le plus petit arbre admettant un dominant parfait unique et $T \in \mathcal{F}$. On suppose que chaque arbre T' d'ordre $n' < n$ admet un unique $\gamma(T')$-ensemble et un unique dominant parfait est dans $\mathcal{F}$.

Soit T un arbre d'ordre n. Si T admet un support z adjacent à au moins deux sommets pendants, alors considérons l'arbre T' obtenu à partir de T en supprimant un sommet pendant quelconque z' adjacent à z. Alors D reste un $\gamma(T')$-ensemble. Maintenant on suppose que D n'est pas unique et soit D' un seconde $\gamma(T')$-ensemble. Alors $z \notin D'$ et z possède un autre sommet pendant $z" \in D'$. Soit w un sommet non pendant adjacent à z dans T (un tel sommet existe toujours car $diam(T) \geq 5$). Alors D' contient au moins un sommet de $N[w] - \{z\}$ et cela pour dominer le sommet w, donc $\{z\} \cup (D' - \{z''\})$ est un $\gamma(T)$-ensemble qui n'est pas unique ou bien n'est pas parfait les deux donnent une contradiction. Par induction sur T' on a $T' \in \mathcal{F}$ par conséquent $T \in \mathcal{F}$ car il est obtenu à partir de T' par l'opération θ_1.

On suppose que chaque sommet support est adjacent à exactement un seule sommet pendant. Soit u' un sommet pendant à distance maximum r d'un sommet de degré plus de deux. Soient u, v, w les sommets parents des sommets u', u, v respectivements. Alors $u \in D$ et $w \notin D$.
Soit $T' = T - \{u', u, v\}$. Alors $D' = D - \{u\}$ est un unique $\gamma(T')$-ensemble et un unique dominant parfait de T'. Par induction sur T' on a $T' \in \mathcal{F}$ et par la suite $T \in \mathcal{F}$ car il est obtenu à partir de T' en utilisant l'opération θ_2. ■

Théorème 14 *Soit un arbre $T \neq k_{1,t}$ les assertions suivantes sont équivalentes :*
a)$-$ $Ct_\gamma(T) = 3$.
b)$-$ *T admet un unique $\gamma(T)$-ensemble et un unique dominant parfait .*
c)$-$ $T \in \mathcal{F}$.

Preuve. $(a) \Rightarrow (b)$: d'aprés le lemme 12
$(b) \Leftrightarrow (c)$: d'aprés le lemme 13.
Il suffit de montrer que (b) implique (a) : Supposons que $Ct_\gamma(T) = 1$, alors d'aprés la proposition 4, il existe un $\gamma(T)$-ensemble qui n'est pas un stable et comme T admet un dominant parfait unique alors $Ct_\gamma(T) \geq 2$. Supposons maintenant que $Ct_\gamma(T) = 2$, alors d'aprés la proposition 6 tout $\gamma(T)$-ensemble est un stable (ce qui est vérifie dans notre cas, car T admet un dominant parfait unique) et il existe un ensemble dominant $S \cup \{x\}$ de cardinalité $\gamma + 1$ contenant au moins deux arêtes ce qui est impossible, car dans ce cas x admet deux voisins dans S, contradiction. Par conséquent $Ct_\gamma(T) \geq 3$ et d'aprés le théorème 2 en déduit que $Ct_\gamma(T) = 3$. ■

4 Le nombre de domination stable par contraction $Cti(G)$

On commence par donner la définition du nombre domination stable par contraction.

Définition 15 *Soit $G = (V, E)$ un graphe connexe tel que $i(G) \geq 2$. On note par $Ct_i(G)$ le nombre minimum d'arête qu'il faut contracter successivement pour faire diminuer $i(G)$.*

Proposition 16 *Pour les chaînes P_n et les cycles C_n,$i(P_n)$=$i(C_n)$=$\lceil \frac{n}{3} \rceil$.*

Proposition 17 *Pour les chaînes P_n et les cycles C_n d'ordre $n \geq 4$, $Ct_i(P_n) = Ct_i(C_n) = i$ tel que $n = 3k + i$ et $1 \leq i \leq 3$.*

Nous donnons par les observations suivantes les conditions suffisantes pour que $Ct_i(G) = 1$.

Observation 18 *Soit $G = (V, E)$ un graphe connexe. S'il existe un i-ensemble S de G contenant au moins un sommet sans sommets privés, alors $Ct_i(G) = 1$.*

Proposition 19 *Soit $G = (V, E)$ un graphe connexe, s'il existe un $\gamma(G)$-ensemble S contenant exactement une arête, alors $Ct_i(G) = 1$.*

Preuve. Il est clair que $\gamma(G) \leq i(G)$ pour tout graphe G. Soit un $\gamma(G)$-ensemble S contenant une arête uv, alors $\gamma(G_{uv}) = \gamma(G) - 1 = i(G_{uv})$. Et par conséquent
$\gamma(G_{uv}) < \gamma(G)$. D'ou $\gamma(G_{uv}) < i(G)$ et par suite $i(G_{uv}) < i(G)$. Donc $Ct_i(G) = 1$. ■

Rappelons que le voisinage privé d'un sommet v par rapport à un ensemble S noté $pn[v, S]$ est l'ensemble des sommets du voisinage fermé de v qui n'ont pas d'autres voisins dans S, i-e: $pn[v, S] = \{u : N[u] \cap S = \{v\}\}$.

Proposition 20 *Soit $G = (V, E)$ un graphe connexe et soient S un $i(G)$-ensemble et $x \in S$.*
a)− Si $|pn[x, S]| = 0$, alors $Cti(G) = 1$.
b)− Si $|pn[x, S]| = d_G(x)$, alors $Cti(G) \leq 3$.
c)− Si $1 < |pn[x, S]| < d_G(x)$, alors $Cti(G) \leq \min_{x \in S}(d_G(x))$.

Preuve. Soient G un graphe connexe et S un $i(G)$-ensemble. Soit x un sommet quelconque de S. Posons $|pn[x,S]| = k$

$a)-$ Si $k = 0$, alors l'ensemble $N(x)$ est dominé par $S - \{x\}$. Dans ce cas la contraction d'une arête reliant x et un de ces voisins fait diminuer $i(G)$, par conséquent $Cti(G) = 1$.

$b)-$ $k = d_G(x)$. Puisque G est connexe alors, il existe une arête reliant un sommet $y \in pn[x,S]$ à un sommet $z \in V - (S \cup pn[x,S])$. Donc il existe un sommet $t \in S$ adjacent à z. Par conséquent la contraction des arêtes xy, yz et zt consécutivement fait diminuer $i(G)$, d'ou $Cti(G) \leq 3$.

$c)-$ Si $1 < k < d_G(x)$, alors il existe un sommet $z \in N(x)$ adjacent à $S - \{x\}$ et la contraction des k arêtes reliant x à $pn[x,S]$ ainsi que l'arête zx fait diminuer $i(G)$. Par conséquent $Cti(G) \leq k+1$ et puisque $k < d_G(x)$ alors $Cti(G) \leq d_G(x)$. ■

Contrairement au paramètre $Ct_\gamma(G)$, par l'observation suivante on montre qu'il existe des graphes où $Ct_i(G)$ peut être trés grand.

Observation 21 *Pour tout entier $k \geq 1$ il existe un arbre T_k tel que $Ct_i(T_k) \geq k$.*

Preuve. Soit T_k un arbre obtenu à partir d'une étoile $K_{1,k}$ avec $k \geq 2$ de centre y en attachant chaque sommet pendant x_j par $(k-1)$ sommets tel que $1 \leq j \leq k$. Il est clair que $S = \{x_1, x_2, ...x_k\}$ est un $i(T_k)$-ensemble tel que $d(x_j) = k$.On peut constater que la contraction d'une arête de type yx_j fait augmenter le nombre de domination stable de l'arbre résultant par rapport à $i(T_k)$. D'autre part la contraction d'une arête joignant un support à son sommet pendant ne fait pas changer $i(T_k)$. On conclut que au moins k arêtes sont necessaires pour diminuer le nombre de domination stable, d'ou $Ct_i(T_k) \geq k$. ■

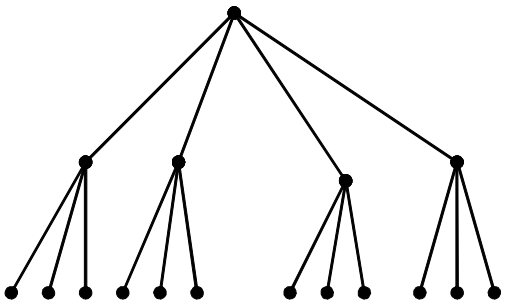


Fig.4.1: L'arbre T_k avec $k = 4$.

References

[1] C.Berge. Graphs, North holland, 1985.

[2] Haynes T.W, Hedetniemi S.T et Slater P.J, " Fundamentals of Domination in graphs", Marcel Dekker, New York, 1998.

[3] T.Burton, et D.P.Sumner, Domination dot-critical graphs. *Discrete Mathematics* 306 (2006) 11-18.

[4] Jia Huang et Jun- Ming Xu, Domination and total Domination contraction numbers of graphs, soumis.

[5] D.W.Bange, A.E.Barkaukas et P.J.Slater. Efficient dominating sets in graphs. In Applications of Discrete Mathématics, R.D. Ringeisen et F.S. Roberts, editors, SIAM, Philadelphia (1988) 189-199.

[6] G.Gunther, B.Hartnell, L.R.Markus et D.Rall, Graphs with unique minimun dominating sets. Congr. Numer. 101(1994), 55-63.

Bounds on the domination number in oriented graphs

[1]Mostafa Blidia and [2]Lyes Ould-Rabah
Lamda-RO, Department of Mathematics, University of Blida.
B.P. 270, Blida, Algeria.
E-mail: [1]m_blidia@yahoo.fr, [2]l.ouldrabah@yahoo.fr

March 19, 2010

Abstract

A dominating set of an oriented graph D is a set S of vertices of D such that every vertex not in S is a successor of some vertex of S. The minimum cardinality of a dominating set of D, denoted $\gamma(D)$, is the domination number of D. An irredundant set of an oriented graph D is a set S of vertices of D such that every vertex of S has a private successor, that is, for all $x \in S$, $|O[x] - O[S - x]| \geq 1$. The irredundance number of an oriented graph, denoted $ir(D)$, is the least number of vertices in a maximal irredundant set. We denote by $\beta_1(D)$ and $s(D)$, the number of edges in a maximum matching and support vertices of the underlyng graph of an oriented graph D, respectively. In this paper, we show that for every oriented graph D, $s(D) \leq ir(D) \leq \gamma(D) \leqslant n(D) - \beta_1(D)$. We also give characterizations of oriented trees satisfying $\gamma(T) = n(T) - \beta_1(T)$ and oriented graphs satisfying $\gamma(D) = s(D)$ and $s(D) = n(D) - \beta_1(D)$, respectively.

Keywords: locating-domination, critical graph.**2000**
Mathematics Subject Classification: 05C69, 05C15.

1 Introduction

An oriented graph (or digraph) D is a finite nonempty set of points called vertices together with a (possibly empty) set of ordered pairs of distinct vertices of D called arcs or oriented edges. An oriented graph D can be obtained from a simple graph G by assigning a direction (possibly both sense) to each edge of G. We say that G is the underlying graph of D and that D is an orientation of G. As with graphs, the vertex set of D is denoted by $V(D)$ and the arc set is denoted by $A(D)$. The oriented graph $D = (V, A)$ considered here has no loops and no multiple arcs (but pairs of opposite arcs are allowed). If $(x, y) \in A$, then the arc is oriented from x to y. The vertex x is called a predecessor of y and y is called a successor of x. If the reversal (y, x) of an arc (x, y) of D is also

present in D, we say that (x, y) is a reversible (symmetrical) arc. If $(x, y) \in A$ but $(y, x) \notin A$, then (x, y) is an asymmetrical arc.

The sets $O(u) = \{v : (u, v) \in A\}$ and $I(u) = \{v : (v, u) \in A\}$ are called the outset and inset of the vertex u. Likewise, $O[u] = O(u) \cup \{u\}$ and $I[u] = I(u) \cup \{u\}$. If $S \subseteq V$ then $O(S) = \bigcup_{s \in S} O(s)$ and $I(S) = \bigcup_{s \in S} I(s)$. Similarly $O[S] = \bigcup_{s \in S} O[s]$ and $I[S] = \bigcup_{s \in S} I[s]$. The indegree of a vertex u is given by $id(u) = |I(u)|$ and the outdegree of a vertex u is $od(u) = |O(u)|$. The maximum outdegree of a vertex in D is denoted by $\Delta_+(D)$

Let G be the underlying graph of a oriented graph D. If $e = uv$ is an edge of G, then u and v are adjacent vertices, while u and e are incident, as are v and e. Furthermore, if e_1 and e_2 are distinct edges of G incident with a common vertex, then e_1 and e_2 are adjacent edges. The *degree* of a vertex v of G is the number of vertices adjacent to v. A vertex of degree one is called a *leaf* and its neighbor is called a *support vertex*. If u is a support vertex, then L_u will denote the set of leaves attached at u. An edge incident with a leaf is called a *pendant edge*. A tree T is a *double star* if it contains exactly two vertices that are not leaves. A double star with p and q leaves attached at each support vertex, respectively, is denoted by $S_{p,q}$. Denote by T_x the subtree induced by a vertex x and its descendants in a rooted tree T. The *diameter* $\text{diam}(G)$ of a graph G is the maximum distance over all pairs of vertices of G. The *corona* $G \circ K_1$ of a graph G is obtained from G by adding a leaf at each of its vertices. For the underlying graph G of a oriented graph D, we denote by $n(D) = n(G), \ell(D) = l(G)$, $s(D) = s(G)$, $L(D) = L(G)$ and $S(D) = S(G)$ the number of vertices, leaves, support vertices and the set of leaves and support vertices of G, respectively.

A set of pairwise independent edges of G is called a matching in G. The number of edges in a maximum matching of G is the edge independence number $\beta_1(G)$ $(= \beta_1(D)$ if there is no ambiguity). If M is a specified matching in graph G, then every vertex of G is incident with at most one edge of M. A vertex that is incident with no edges of M is called an $\overline{M}$-vertex.

A set $S \subseteq V$ of an oriented graph D is independent if and only if for all $x, y \in S$, $x \notin O(y)$. The size of the largest independent set in D is denoted by $\beta(D)$.

A set $S \subseteq V$ of an oriented graph D is a dominating set of D if, for all $v \notin S$, v is a successor of some vertex $s \in S$ or $O[S] = V(D)$. We use the notation $\gamma(D)$ to represent the domination number of an oriented graph, i.e., the minimum cardinality of a set $S \subseteq V$ which is dominating. A set $S \subseteq V$ is irredundant if, for all $x \in S$, $|O[x] - O[S - x]| \geq 1$. If $y \in O[x] - O[S - x]$, then we say that y is a private successor of x with respect to S. Observe that x may be its own private successor. The irredundance number of an oriented graph, denoted $ir(D)$, is the least number of vertices in a maximal irredundant set. It is clear that $ir(D) \leq \gamma(D)$. A dominating set of D with minimum cardinality is called a $\gamma(D)$-set. For more details on domination in graphs, see the monographs by

Haynes, Hedetniemi, and Slater [4, 5].

In general, domination in oriented graphs has not been studied as intensively studied as that in graphs without orientation. In [3], Ghoshal, Lasker, and Pillone consider related topics in oriented graphs and suggest further avenues of study. Gallai-type results have been considered in [7]. In [1], Albertoon and al. characterize oriented trees satisfying $\gamma(D) + \Delta_+(D) = n$ and thus satisfying $ir(D) + \Delta_+(D) = n$.

In this paper, we show that for every oriented graph D, $s(D) \leq ir(D) \leq \gamma(D) \leqslant n(D) - \beta_1(D)$. We also give characterizations of oriented trees satisfying $\gamma(T) = n(T) - \beta_1(T)$ and oriented graphs satisfying $\gamma(D) = s(D)$ and $s(D) = n(D) - \beta_1(D)$, respectively.

2 Bounds

Before presenting our results, we recall some know bounds of a dominating number in oriented graphs.

Theorem 1 *[5] For any oriented graph D on n vertices,* $\dfrac{n(D)}{1 + \Delta_+(D)} \leq \gamma(D) \leqslant n(D) - \Delta_+(D)$.

Theorem 2 *[6] For a strongly connected oriented graph D on n vertices,* $\gamma(D) \leqslant \left\lceil \dfrac{n(D)}{2} \right\rceil$.

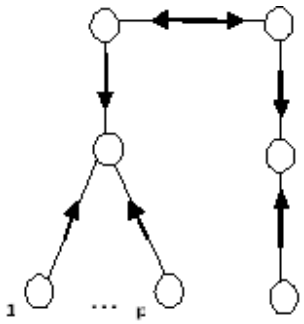
Observation 3 *Let D be an oriented graph.*

1. *Let x be a vertex of D such that $I(x) = \varnothing$. Then every $\gamma(D)$-set contains x.*
2. *Let v be a support vertex of D. Then every $\gamma(D)$-set contains at least one vertex of $L_v \cup \{v\}$.*

Recall that the number $\beta_1(D)$ can be computed for any graph in polynomial time [2]. Therefore, the following bounds can also be computed in polynomial time.

Theorem 4 *For any oriented graph D on n vertices, $s(D) \leq ir(D) \leq \gamma(D) \leqslant n(D) - \beta_1(D)$.*

Proof. Let S be a $ir(D)$-set of D. For every support vertex v such that $S \cap (L_v \cup \{v\}) = \varnothing$, correspond at least one vertex $z \in S$ with v its unique private successor (this is possible for otherwise $S \cup L_v$ is an irredundant set which contradicts the maximality of S). If z is a support vertex, then $L_z \in S$.



Indeed, all pendant edges attached at v are oriented from $y \in L_v$ to v (may be symmetrically). So, $ir(D) = |S| \geq s(D)$.

Let $M = \{x_i y_i : 1 \leq i \leq \beta_1\}$ be a set of edges of a maximum matching in the underlying graph G of D with Z_M the set of all $\overline{M}$-vertices of G (which are incident with no edges of M). Without loss of generality, we suppose that (x_i, y_i) is an arc of $D; 1 \leq i \leq \beta_1$. It is clear that $S = \{x_1, x_2, ...x_{\beta_1}\} \cup Z_M$ is a dominating set of D. So, $\gamma(D) \leq |S| = |\{x_1, x_2, ...x_{\beta_1}\}| + |Z_M| = \beta_1 + n - 2\beta_1 = n - \beta_1$, which implies the upper bound $\gamma(D) \leqslant n(D) - \beta_1(D)$. ■

Note that the difference between $\gamma(D)$ and $ir(D)$ can be arbitrarily large even for oriented trees. To see this, consider the oriented tree of Figure 1, where $\gamma(T) = p + 2$ and $ir(T) = 2 = s(D)$.

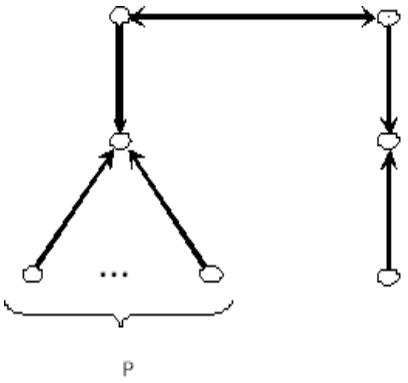


Figure 1

Next in Section 3 and 4, we present characterizations of special oriented graphs achieving equality in each bound of $s(D) \leq \gamma(D) \leqslant n(D) - \beta_1(D)$.

3 Characterization of directed trees achieving the upper bound

We begin by giving useful results:

Lemma 5 *Let D be a nontrivial oriented graph. If $\gamma(D) = n(D) - \beta_1(D)$, then every maximum matching $M = \{x_i y_i : 1 \leq i \leq \beta_1\}$ in the underlying graph G of D with corresponding arcs (x_i, y_i) ; $1 \leq i \leq \beta_1$ and Z_M the set of all $\overline{M}$-vertices of G, satisfies:*

1. *$\forall z \in Z_M$, $I(z) \cap \{x_1, ..., x_{\beta_1}\} = \varnothing$.*
2. *$\forall e = xy$ an edge of M and (x, y) a corresponding arc in D. If one end-vertex z of e satisfies $I(z) \cap ((\{x_1, ..., x_{\beta_1}\} - \{x\}) \cup Z_M) \neq \varnothing$, then the other end-vertex z' of e verifies $I(z') \cap ((\{x_1, ..., x_{\beta_1}\} - \{x\}) \cup Z_M) = \varnothing$.*

Proof. Let $M = \{x_i y_i : 1 \leq i \leq \beta_1\}$ be a maximum matching in the underlying graph G of D with corresponding arcs (x_i, y_i) ; $1 \leq i \leq \beta_1$ and Z_M the set of all $\overline{M}$-vertices of G. First, suppose that there exists $z \in Z_M$ such that $I(z) \cap \{x_1, ..., x_{\beta_1}\} \neq \varnothing$. It is clear that $S = \{x_1, ..., x_{\beta_1}\} \cup (Z_M - \{z\})$ is a dominating set of D and $|S| = |\{x_1, x_2, ...x_{\beta_1}\}| + |Z_M - \{z\}| = \beta_1 + n - 2\beta_1 - 1 = n - \beta_1 - 1$. Then S is a dominating set of D of size less than $n - \beta_1$, a contradiction. Now assume that there exists an edge $e = xy$ of M with a corresponding arc (x, y) in D, which do not satisfy Part 2 of Lemma 5. Without loss of generality, suppose that $I(y) \cap ((\{x_1, ..., x_{\beta_1}\} - \{x\}) \cup Z_M) \neq \varnothing$ and $I(x) \cap ((\{x_1, ..., x_{\beta_1}\} - \{x\}) \cup Z_M) \neq \varnothing$. Consider now $S = ((\{x_1, ..., x_{\beta_1}\} - \{x\}) \cup Z_M)$, it is clear that S is a dominating set of D of size less than $n - \beta_1$, a contradiction. ■

Observation 6 *Let T be a tree.*

1. *If T is a tree obtained from a tree T' by attaching a vertex to a support vertex of T', then $\beta_1(T) = \beta_1(T')$.*
2. *For every support vertex v of a nontrivial tree, there exits a maximum matching M which contains a pendant edge with end-vertex v.*
3. *If T is a tree obtained from a tree T' by attaching an end-vertex of P_2 to a vertex of T', then $\beta_1(T) = \beta_1(T') + 1$.*

We call the oriented graph of Figure 2 the obstruction (pairs of opposite arcs are allowed).

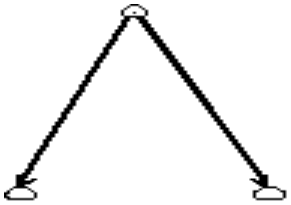


Figure 2: The obstruction

Let $\overrightarrow{K_{1,p}}$ be the oriented star (the underlying graph is a star) without the obstruction as a subdigraph, that is, the oriented star with center x such that $|O(x) \cap L_x| \leq 1$.

Observation 7 *Let T be a nontrivial oriented tree. If $\gamma(T) = n(T) - \beta_1(T)$, then for every support vertex x of T, the subdigraph induced by $L_x \cup \{x\}$ is a oriented star $\overrightarrow{K_{1,p}}$; $p \geq 1$.*

Proof. Assume that there exists a support vertex x of T such that $L_x \cup \{x\}$ is a oriented star $\overrightarrow{K_{1,p}}$; $p \geq 2$ with the obstruction as a subdigraph. By Part 2 of Observation 6, we consider a maximum matching M which contains a pendant edge with end-vertex x. Then Part 1 of Lemma 5 is not satisfied, so $\gamma(T) < n(T) - \beta_1(T)$, a contradiction. ■

We denote by $\overrightarrow{S_{p,q}}$ the oriented tree obtained from two oriented stars $\overrightarrow{K_{1,p}}$ and $\overrightarrow{K_{1,q}}$ by attaching the center x of $\overrightarrow{K_{1,p}}$ to the center y of $\overrightarrow{K_{1,q}}$ where the edge xy is arbitrary oriented.

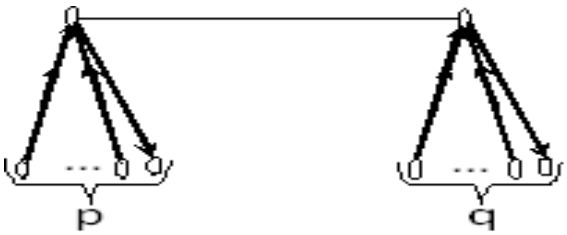


Figure 3: The oriented tree $\overrightarrow{S_{p,q}}$

We also denote by $\overrightarrow{P_2^1(x,y)}$ the oriented chain obtained from $P_2 = xy$ where the edge xy is asymmetrically oriented from y to x, that is, (x, y) is not present. And denote by $\overrightarrow{P_2^2(x,y)}$ the oriented chain obtained from $P_2 = xy$ where the edge xy is oriented from x to y, possibly the arc (y, x) is also present.

And denote by $H_k(z)$ the oriented tree obtained from oriented chains $\overrightarrow{P_2^2(x_i,y_i)}$; $1 \leq i \leq k$ and join every vertex x_i ; $1 \leq i \leq k$ by an edge to vertex z, where at least one edge x_iz is oriented from x_i to z (possibly symmetrically) and all others are arbitrary oriented. (For all these oriented graphs see Figure 4 and Figure 5.)

Figure 4: $\overrightarrow{P_2^1(x,y)}$ and $\overrightarrow{P_2^2(x,y)}$

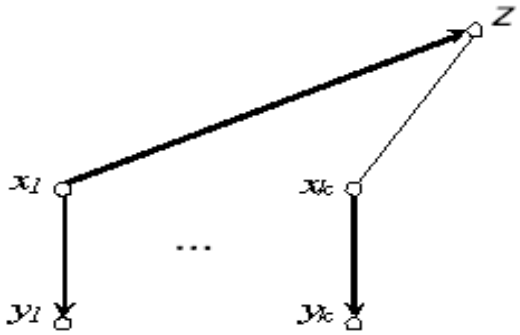


Figure 5: $H_k(z)$

In order to characterize the oriented trees with $\gamma(T) = n(T) - \beta_1(T)$, we introduce the family $\mathcal{F}$ of all trees T that can be obtained from a sequence T_1, T_2, ..., T_m ($m \geq 1$) of oriented trees, where T_1 is $\overrightarrow{P_2^1(x,y)}$, $\overrightarrow{P_2^2(x,y)}$, $T = T_m$, and, if $m \geq 2$, T_{i+1} is obtained recursively from T_i by one of the five operations defined below.

- **Operation** $\mathcal{O}_1$: Add a vertex y and join y by an edge to a support vertex x of T_i, where the edge xy is asymmetrically oriented from y to x.
- **Operation** $\mathcal{O}_2$: Add an oriented chain $\overrightarrow{P_2^1(x,y)}$ and join x by an edge to a vertex z of T_i, where the edge xz is arbitrary oriented.
- **Operation** $\mathcal{O}_3$: Add an oriented chain $\overrightarrow{P_2^2(x,y)}$ and join x by an edge to a support vertex z of T_i, where the edge xz is arbitrary oriented.
- **Operation** $\mathcal{O}_4$: Add oriented chains $\overrightarrow{P_2^2(x_i,y_i)}$; $i = 1, ...k$ and join every vertex x_i by an edge to a pendent vertex z of T_i, where the edge x_iz is asymmetrically oriented from z to x_i for $i = 1, ...k$.

- **Operation** $\mathcal{O}_5$: Add an oriented tree $H_k(z)$ and join z by an edge to a vertex w of T_i such that there exists a maximum matching M where w is a $\overline{M}$-vertex and where the edge zw is arbitrary oriented.

Lemma 8 *If a nontrivial oriented tree T is in $\mathcal{F}$, then $\gamma(T) = n(T) - \beta_1(T)$.*

Proof. Let T be a nontrivial oriented tree of $\mathcal{F}$. To show that $\gamma(T) = n(T) - \beta_1(T)$, we proceed by induction on m where $m-1$ is the number of operations performed to construct T from T_1. If $m = 1$, then $T = \overrightarrow{P_2^1(x,y)}$ or $\overrightarrow{P_2^2(x,y)}$ and since $\beta_1(T) = 1$, $\gamma(T) = 1$ and $n(T_1) = 2$, $\gamma(T) = n(T) - \beta_1(T)$. This establishes the basis case. Assume now that $m \geq 2$ and the result holds for all trees of $\mathcal{F}$ that can be constructed from a sequence of at most $m-2$ operations. Let $T = T_m$ be a nontrivial oriented tree of $\mathcal{F}$ constructed by $m-1$ operations, $T' = T_{m-1}$ and assume that T' has order $n(T'), \beta_1(T')$ and $\gamma(T')$. By induction hypothesis applied to T', we know that $\gamma(T') = n(T') - \beta_1(T')$. We consider five cases depending on whether T is obtained from T' by using $\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_4$ or $\mathcal{O}_5$.

Case 1. Suppose that T was obtained from T' by operation $\mathcal{O}_1$. Let S' be $\gamma(T')$-set. Then $S' \cup \{y\}$ is a dominating set of T, so $\gamma(T) \leq |S' \cup \{y\}| \leq \gamma(T')+1$. Let now S be a $\gamma(T)$-set of T. By Part 1 of Observation 3, S contains y. Without loss of generality since x is a support vertex of T', either x is contained in S or x is dominated by one vertex of $L_x - \{y\}$, so $S' = S - \{y\}$ is dominating set of T'. So, $\gamma(T') \leq |S'| = |S - \{y\}| = \gamma(T) - 1$. Thus, $\gamma(T) = \gamma(T') + 1$. By induction $\gamma(T') = n(T') - \beta_1(T')$ and by Part 1 Observation 6 $\beta_1(T) = \beta_1(T')$, so $\gamma(T) = n(T') - \beta_1(T') + 1 = n(T) - \beta_1(T)$.

Case 2. Suppose that T was obtained from T' by performing operation $\mathcal{O}_2$. Let S' be $\gamma(T')$-set. Then $S' \cup \{y\}$ is a dominating set of T, so $\gamma(T) \leq |S' \cup \{y\}| \leq \gamma(T') + 1$. Let now S be a $\gamma(T)$-set of T. By Part 1 of Observation 3, S contains y. Without loss of generality, we suppose that $x \notin S$ (otherwise replace x by z). So $S' = S - \{y\}$ is a dominating set of T'. So, $\gamma(T') \leq |S'| = |S - \{y\}| = \gamma(T) - 1$. Thus, $\gamma(T) = \gamma(T') + 1$. By induction $\gamma(T') = n(T') - \beta_1(T')$ and by Part 3 Observation 6, $\beta_1(T) = \beta_1(T') + 1$, so $\gamma(T) = n(T') - \beta_1(T) + 2 = n(T) - \beta_1(T)$.

Case 3. Suppose that T was obtained from T' by performing operation $\mathcal{O}_3$. Let S' be $\gamma(T')$-set. Then $S' \cup \{x\}$ is a dominating set of T, so $\gamma(T) \leq |S' \cup \{x\}| \leq \gamma(T')+1$. Let now S be a $\gamma(T)$-set of T. Without loss of generality, we suppose that $x \in S$ and $y \notin S$ and since z is a support vertex of T', either z is contained in S or z is dominated by one vertex of L_z, so $S' = S - \{x\}$ is a dominating set of T'. So, $\gamma(T) \leq |S'| = |S - \{x\}| = \gamma(T) - 1$. Thus, $\gamma(T) = \gamma(T') + 1$. By induction $\gamma(T') = n(T') - \beta_1(T')$ and by Part 3 of Observation 6, $\beta_1(T) = \beta_1(T')+1$, so $\gamma(T) = n(T') - \beta_1(T) + 2 = n(T) - \beta_1(T)$.

Case 4. Suppose that T was obtained from T' by performing operation $\mathcal{O}_4$. Let S' be $\gamma(T')$-set. Then $S' \cup \{x_1, ..., x_k\}$ is a dominating set of T, so $\gamma(T) \leq |S' \cup \{x_1, ..., x_k\}| \leq \gamma(T') + k$. Let now S be a $\gamma(T)$-set of T. Without loss of

generality, we suppose that $x_i \in S$ and $y_i \notin S$ for $i = 1, ..., k$ and since every edge $x_i z$ is asymmetrically oriented from z to x_i for $i = 1, ..., k$, $S' = S - \{x_1, ..., x_k\}$ is a dominating set of T'. So, $\gamma(T') \leq |S'| = |S - \{x_1, ..., x_k\}| = \gamma(T) - k$.Thus, $\gamma(T) = \gamma(T') + k$. By induction $\gamma(T') = n(T') - \beta_1(T')$ and by Part 3 of Observation 6, $\beta_1(T) = \beta_1(T') + k$, so $\gamma(T) = n(T') - \beta_1(T) + 2k = n(T) - \beta_1(T)$.

Case 5. Suppose that T was obtained from T' by performing operation $\mathcal{O}_5$. Let S' be a $\gamma(T')$-set. Since there exists at least one edge $x_i z$ which is oriented from x_i to z, $S' \cup \{x_1, ..., x_k\}$ is a dominating set of T, so $\gamma(T) \leq |S' \cup \{x_1, ..., x_k\}| \leq \gamma(T') + k$. Let now S be a $\gamma(T)$-set of T. Without loss of generality, we suppose that $x_i \in S$ and $y_i \notin S$ for $i = 1, ..., k$ and $z \notin S$ (otherwise replace w by z). So $S' = S - \{x_1, ..., x_k\}$ is a dominating set of T'. So, $\gamma(T') \leq |S'| = |S - \{x_1, ..., x_k\}| = \gamma(T) - k$. Thus, $\gamma(T) = \gamma(T') + k$. By induction $\gamma(T') = n(T') - \beta_1(T')$ and since there exists a maximum matching M with w is a $\overline{M}$-vertex, it is clear that $\beta_1(T) = \beta_1(T') + k + 1$, so $\gamma(T) = n(T') - \beta_1(T) + 2k + 1 = n(T) - \beta_1(T)$. ■

Theorem 9 *If T is a nontrivial oriented tree of order $n(T)$, then $\gamma(T) = n(T) - \beta_1(T)$ if and only if $T \in \mathcal{F}$.*

Proof. If $T \in \mathcal{F}$, then by Lemma 8, $\gamma(T) = n(T) - \beta_1(T)$. To prove that if T is a nontrivial oriented tree of order $n \geq 2$, then $\gamma(T) = n(T) - \beta_1(T)$ only if $T \in \mathcal{F}$, we process by induction on the order of T. If $\text{diam}(T) = 1$ (the diameter of the underlying tree of the oriented tree), then $T = \overrightarrow{P_2^1(x, y)}$ or $\overrightarrow{P_2^2(x, y)}$ which belongs to $\mathcal{F}$. If $\text{diam}(T) = 2$, then $T = \overrightarrow{K_{1,p}}$ (see Observation 7) which is obtained from $\overrightarrow{P_2^1(x, y)}$ or $\overrightarrow{P_2^2(x, y)}$ by applying $p - 2$ times $\mathcal{O}_1$. If $\text{diam}(T) = 3$, then $T = \overrightarrow{S_{p,q}}$ which is obtained by applying operations $\mathcal{O}_2$ or $\mathcal{O}_3$ followed by zero or more repetitions of Operation $\mathcal{O}_1$. This establishes the basis cases.

So we suppose that $\text{diam}(T) \geq 4$, and that every nontrivial oriented tree T' of order less than n satisfying $\gamma(T') = n(T') - \beta_1(T')$ is in $\mathcal{F}$. Let T be a nontrivial oriented tree of order n satisfying $\gamma(T) = n(T) - \beta_1(T)$. Consider a $\gamma(T)$-set S of T. We consider the underlying tree of the oriented tree and we root T at a vertex r of maximum eccentricity. Let x be a support vertex at maximum distance from r in the rooted tree. Let T_u denote the subtree induced by a vertex u and its descendants in the rooted tree T. We consider three cases.

Case 1. x is a support vertex with $|L_x| \geq 2$. By Observation 7, the subdigraph induced by $L_x \cup \{x\}$ is a oriented star $\overrightarrow{K_{1,p}}$; $p \geq 1$ without the obstruction as a subdigraph. So, there exists y attached to x with the edge xy asymmetrically oriented from y to x. Let $T' = T - \{y\}$. Then $n(T') = n(T) - 1$ and by Part 1 of Observation 6, $\beta_1(T) = \beta_1(T')$. By Part 1 of Observation 3, S contains y, and since x is a support, without loss of generality $S' = S - \{y\}$ is a dominating set of T' (x is dominated by a leaf of $L_x - \{y\}$ or $x \in S$). So, $\gamma(T) - 1 \leq \gamma(T') \leq |S'| = |S - \{y\}| = \gamma(T) - 1$. Thus $\gamma(T') = \gamma(T) - 1 = n(T) - \beta_1(T) - 1 = n(T') - \beta_1(T')$. By induction on T', we have $T' \in \mathcal{F}$, implying that $T \in \mathcal{F}$ because T is obtained by using Operation $\mathcal{O}_1$.

From now on we may assume that $|L_x| = 1$. Let $L_x = \{y\}$. Let z be the parent of x in the rooted tree, since $\text{diam}(T) \geq 4$, z exists.

Case 2. The edge xy is asymmetrically oriented from y to x, that is; (x, y) is not present. Let $T' = T - \{x, y\}$. Then $n(T') = n(T) - 2$ and by Part 3 of Observation 6, $\beta_1(T) = \beta_1(T') + 1$. Also, by Part 1 of Observation 3, S contains y. Without loss of generality, we suppose that $x \notin S$ (otherwise replace x by z). So $S' = S - \{y\}$ is dominating set of T'. Thus, $\gamma(T) - 1 \leq \gamma(T') \leq |S'| = |S - \{y\}| = \gamma(T) - 1$ which implies that $\gamma(T') = \gamma(T) - 1 = n(T) - \beta_1(T) - 1 = n(T) - \beta_1(T') - 2 = n(T') - \beta_1(T')$. By induction on T', we have $T' \in \mathcal{F}$, implying that $T \in \mathcal{F}$ because T is obtained by using Operation $\mathcal{O}_2$.

Case 3. The edge xy is oriented from x to y, possibly the arc (x, y) is symmetrical. Let us examine the following subcases:

Case 3.1. z is a support vertex in T. Let $T' = T - \{x, y\}$. Then $n(T') = n(T) - 2$ and by Part 3 of Observation 6, $\beta_1(T) = \beta_1(T') + 1$. Without loss of generality, we suppose that $x \in S$ and $y \notin S$ (otherwise replace y by x) and since z is a support vertex of T', either z is contained in S or z is dominated by one vertex of L_z, so $S' = S - \{x\}$ is dominating set of T'. Thus, $\gamma(T) - 1 \leq \gamma(T') \leq |S'| = |S - \{y\}| = \gamma(T) - 1$ which implies that $\gamma(T') = \gamma(T) - 1 = n(T) - \beta_1(T) - 1 = n(T) - \beta_1(T') - 2 = n(T') - \beta_1(T')$. By induction on T', we have $T' \in \mathcal{F}$, implying that $T \in \mathcal{F}$ because T is obtained by using Operation $\mathcal{O}_3$.

Case 3.2. z is not a support vertex in T. We can suppose that every child x of z in the rooted tree is a weak support with $L_x = \{y\}$ in the underlying tree and is a predecessor of y (otherwise we can apply **Case 2**). So, let $\overrightarrow{P_2^2(x_i, y_i)}$; $i = 1, ..., k$ $(k \geq 1)$ be oriented chains where every x_i is joined to the vertex z in T.

- If the edge $x_i z$ is asymmetrically oriented from z to x_i for $i = 1, ..., k$, then consider $T' = T - \bigcup_{i=1}^{k} \{x_i, y_i\}$. Since T has a diameter at least four, T' is nontrivial oriented tree and z is a pendant vertex in T'. Since $n(T') = n(T) - 2k$ and by Part 3 of Observation 6, $\beta_1(T) = \beta_1(T') + k$ and it is a routine matter to check $\gamma(T') = \gamma(T) - k$. Hence $\gamma(T') = \gamma(T) - k = n(T) - \beta_1(T) - k = n(T) - \beta_1(T') - 2k = n(T') - \beta_1(T')$. Applying the inductive hypothesis to T', we have $T' \in \mathcal{F}$. Since T is obtained from T' by using Operation $\mathcal{O}_4$, $T \in \mathcal{F}$.

- If there exist an edge $x_i z$ which is oriented from x_i to z (possibly symmetrically), then since $\text{diam}(T) \geq 4$, let w be the parent of z in the rooted tree. Let $T' = T - (\bigcup_{i=1}^{k} \{x_i, y_i\} \cup \{z\})$, $n(T') = n(T) - 2k - 1$. Also, Since T has a diameter at least four, T' is nontrivial oriented tree. It is a routine matter to check $\gamma(T') = \gamma(T) - k$. If for every maximum matching M of T', w is incident with at most one edge of M, then $\beta_1(T) = \beta_1(T') + k$. So, $\gamma(T) = \gamma(T') + k \leq n(T') - \beta_1(T') + k = n(T') - \beta_1(T) + 2k = n(T) - 1 - \beta_1(T) < n(T) - \beta_1(T)$, a contradiction. However, there exists a maximum matching M with w as a $\overline{M}$-vertex. Hence, $\beta_1(T) = \beta_1(T') + k + 1$ and $\gamma(T') = \gamma(T) - k = n(T) - \beta_1(T) - k = n(T) - \beta_1(T') - 2k - 1 = n(T') - \beta_1(T')$. Applying the inductive hypothesis to T', we have $T' \in \mathcal{F}$. Since T is obtained from T' by using Operation $\mathcal{O}_5$, $T \in$

$\mathcal{F}$. This achieves the proof. ■

4 Characterization of digraphs achieving the lower bound

Theorem 10 *Let D be a oriented graph. Then $\gamma(D) = s(D)$ if and only if the oriented graph D verifies :*

1. *For every vertex z of $V(D) - (S(D) \cup L(D))$, $I(z) \cap S(D) \neq \varnothing$.*
2. *For every vertex $x \in S(D)$ with $|L_x| \geq 2$, $O(x) \cap L_x = L_x$.*
3. *Let $L' = \{y \in L \ / \ I(y) \cap S(D) = \varnothing\}$, for every $z \in V(D) - (S(D) \cup L(D))$, $(I(z) \setminus O(L')) \cap S(D) \neq \varnothing$.*

Proof. We first prove the part "only if", suppose that one of the conditions is not satisfied. Then in all cases, $\gamma(D) > s(D)$, a contradiction.

We prove the part "if", by Theorem 4, $\gamma(D) \geq s(D)$. We construct the dominating set S' as follow, set every support vertex with at least two leaves in S'. If x is a support vertex with one leaf and $O(x) \cap L_x = \varnothing$, then set the leaf in S', if not set x in S'. By construction, $|S'| = s(D)$ and S' dominates all vertices of $S(D) \cup L(D)$. Suppose there exists a vertex z of $V(D) - (S(D) \cup L(D))$ which is not dominated by S'. By Part $1^\circ /$ of Theorem 10, $I(z) \cap S(D) \neq \varnothing$. Let $S'' = I(z) \cap S(D)$, by construction of S' the leaves attached to support vertices of S'' are in S'. Therefore, for every vertex x of S'' $O(x) \cap L_x = \varnothing$, a contradiction with Part $3^\circ /$ of Theorem 10. So S' is a dominating set. $|S'| = s(D) \geq \gamma(D)$, which implies that $\gamma(D) = s(D)$. ■

Theorem 11 *Let D be a oriented graph. Then $\gamma(D) = s(D) = n(D) - \beta_1(D)$ if and only if the underlying graph G of D is a corona.*

To prove Theorem 11, we use the following result due to Xu [8].

Theorem 12 *[8]Let G be a graph. Then $\beta(G) + \beta_1(G) \leq n(G)$.*

Proof of Theorem 11. We first prove the part "if". If the underlying graph of D is a corona, then G has a perfect matching, $\beta_1(D) = \dfrac{n(D)}{2} = s(D)$. By Theorem 4, $\dfrac{n(D)}{2} = s(D) \leq \gamma(D) \leq n(D) - \beta_1(D) = n(D) - \dfrac{n(D)}{2} = \dfrac{n(D)}{2}$. Thus $\gamma(D) = \dfrac{n(D)}{2} = s(D)$.

We prove the part "only if", by Theorem 12, $s(D) = n(D) - \beta_1(D) \geq \beta(D)$ and $s(D) \leq l(D) \leq \beta(D)$. So, $s(D) = l(D) = \beta(D)$ which implies that $V(D) - (S(D) \cup L(D)) = \varnothing$. It follows that the underlying G of D is a corona. This complete the proof ■

References

[1] J. Albertson, A. Harris, L. Langley, and S. Merz, "Domination parameters and Gallai-type theorems for directed trees." Ars Combin. 81 (2006) 201–207.

[2] J. Edmonds, Paths, trees and flowers. Canad. J. Math. 17 (1965) 449–467.

[3] J. Ghoshal, R. Laskar and D. Pillone, "Topics on domination in directed graphs." In *Domination in Graphs: Advanced Topics*, T. W. Haynes, S. T. Hedetniemi, and P. J. Slater (eds), , Marcel Dekker, New York, 1998, 401-437.

[4] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, *Fundamentals of Domination in Graphs*, Marcel Dekker, New York, 1998.

[5] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater (eds), *Domination in Graphs: Advanced Topics*, Marcel Dekker, New York, 1998.

[6] Changwoo Lee, On the domination number of a digraph. Ph.D. Dissertation, Michigan State University (1994).

[7] S.K. Merz and D.J. Stewart, "Gallai-type theorems and domination in digraphs and tournaments." Cong. Numer., 154 (2002) 31–41.

[8] S. Xu, Relations between parameters of graphs. Discrete Math. 89 (1991), 65–88.

Note on b-colorings in Harary graphs

Zoham Zemir[†], **Noureddine Ikhlef Eschouf**[‡]
and Mostafa Blidia[†]

[†]LAMDA-RO, Department of Mathematics,
University of Blida, B.P. 270, Blida, Algeria.
E-mail: zohaze@yahoo.fr, mblidia@hotmail.com
[‡]University Yahia Farès of Medéa.
nour_echouf@yahoo.fr,

Abstract

A b-coloring is a coloring of the vertices of a graph such that each color class contains a vertex that has a neighbor in all other color classes. The b-chromatic number $b(G)$ is the largest integer k such that G admits a b-coloring with k colors. In this note, according to the values taken by the order n of a graph, we determine exact values or bounds for the b-chromatic number of $H_{2m,n}$ which is the Harary graph $H_{k,n}$ when k is even. Therefore our result improves the result concerning the b-chromatic of p-th power graphs of cycles and give a negative answer to the open problem of Effantin and Kheddouci.

Keywords: Coloration, b-coloring, b-chromatic number.
AMS Subject Classification: 05C69.

1 Introduction

A proper coloring of a graph $G = (V, E)$ is a mapping c from V to the set of positive integers (colors) such that any two adjacent vertices are mapped to different colors. Each set of vertices colored with one color is a stable set of vertices or color class of G, so a coloring is a partition of V into stable sets. The smallest number k for which G admits a coloring with k colors is the chromatic number $\chi(G)$ of G.

A *b-coloring* is a proper coloring such that every color class i contains at least one vertex that has a neighbor in all the other classes. Any such vertex will be called a *b-dominating vertex* of color i. The *b-chromatic number* $b(G)$ is the largest integer k such that G admits a b-coloring with k colors.

The motivation of this special coloring is as follow. Let c be an arbitrary proper coloring of G and suppose we want to decrease the number of colors by recoloring all the vertices of a given color class X with other colors that is by

putting the vertices of X in other color class. Then this is possible if and only if no vertex of X is a b-dominating vertex. In other words, one color can be recuperated by recoloring each vertex of some fixed color class if and only if the coloring c is not a b-coloring.

The open neighborhood of a vertex $v \in V$ is $N(v) = \{u \in V \mid uv \in E\}$, i.e, the set of all vertices adjacent with v. The closed neighborhoods of v is $N[v] = N(v) \cup \{v\}$. The degree of a vertex v of G is $d(v) = |N(v)|$. By $\Delta(G)$ we denote the maximum degree of G. Let $\Delta(G)$ be the maximum degree in G, and let $m(G)$ be the largest integer k such that G has k vertices of degree at least $k-1$. It is easy to see that every graph G satisfies

$$b(G) \leq m(G) \leq \Delta(G) + 1$$

(the first inequality follows from the fact that if G has any b-coloring with k colors then it has k vertices of degree at least $k-1$; the second inequality follows from the definition of $m(G)$). Irving and Manlove [10, 18] proved that every tree T has b-chromatic number $b(T)$ equal to either $m(T)$ or $m(T) - 1$, and their proof is a polynomial-time algorithm that computes the value of $b(T)$. On the other hand, Kratochvíl, Tuza and Voigt [17] proved that it is NP-complete to decide if $b(G) = m(G)$, even when restricted to the class of connected bipartite graphs such that $m(G) = \Delta(G)+1$. These NP-completeness results have incited searchers to establish bounds on the b-chromatic number in general or to find exact or approximate values for subclasses of graphs (see: [2, 3, 4, 6, 5, 7, 8, 9, 11, 12, 13, 14, 15, 17, 16]).

For $2 \leq k < n$, the Harary graph $H_{k,n}$ on n vertices is defined by West [19] as follows: Place n vertices around a circle, equally spaced. If k is even, $H_{k,n}$ is formed by making each vertex adjacent to the nearest $\frac{k}{2}$ vertices in each direction around the circle. If k is odd and n is even, $H_{k,n}$ is formed by making each vertex adjacent to the nearest $\frac{(k-1)}{2}$ vertices in each direction around the circle and to the diametrically opposite vertex. In both cases, $H_{k,n}$ is k-regular. If both k and n are odd, $H_{k,n}$ is constructed as follows. It has vertex $v_0, v_1, ..., v_{n-1}$ and is constructed from $H_{k-1,n}$ by adding edges joining vertex v_i to vertex $v_{i+\frac{(n-1)}{2}}$ for $0 \leq i \leq \frac{(n-1)}{2}$.

We denote by $dist_G(x, y)$ the distance between vertices x and y in G. The p-th power graph G^p with $p \geq 1$ is a graph obtained from G by adding an edge between every pair of vertices x and y with $dist_G(x, y) \leq p$, in particular $G^1 = G$. The p-th power graph of a cycle C_n with $p \geq 1$ which is C_n^p is the the Harary graph $H_{k,n}$ with $k = 2p$. In [5], Effantin and Kheddouci investigate the b-chromatic number of the p-th power graph, so, they determine exact values and bounds for b-chromatic number of the p-th power graph of paths and the p-th power graph of cycles.

In this note, according to the values taken by the order n of a graph, we determine exact values or bounds for the b-chromatic number of $H_{2m,n}$ which is the Harary graph $\mathrm{H}_{k,n}$ when k is even. Therefore our result improves the result

in [5], concerning the b-chromatic of p-th power graphs of cycles. Also we give a negative answer to the open problem of Effantin and Kheddouci.

2 Main result

Theorem 1 *Let $H_{2m,n}$ be the Harary graph. Then*

$$b(H_{2m,n}) = \begin{cases} 2m+1 & \text{if } n = 2m+1 \text{ or } n \geq 4m+1 \\ 2m - \left\lfloor \dfrac{4m-n}{3} \right\rfloor & \text{if } \left\lceil \dfrac{5m+3}{2} \right\rceil \leq n \leq 4m \\ \geq n-m-1 & \text{if } 2m+2 \leq n < \left\lceil \dfrac{5m+3}{2} \right\rceil \end{cases}$$

Proof. We distinguish between four cases according to each value of the order of $H_{2m,n}$.

Case 1: $n = 2m+1$. Then $H_{2m,n}$ is a clique of order $2m+1$ and clearly $b(H_{2m,n}) = \chi(H_{2m,n}) = 2m+1$.

Case 2: $n \geq 4m+1$. Since $\Delta(H_{2m,n}) = 2m$, $b(H_{2m,n}) \leq \Delta(H_{2m,n}) + 1 = 2m+1$. To prove equality, we construct a b-coloring with $2m+1$ colors $0, 1, 2, ..., 2m$ as follow. Let $v_0, v_1, ..., v_{n-1}$ be vertices of $H_{2m,n}$ in this order around the circle. First, assign color 0 to v_0. Since $n \geq 4m+1$, we begin by coloring the nearest $4m$ vertices to v_0; $2m$ vertices in each direction around the circle according to the ordering of vertices. Assign color i to v_i; $i = 1, ..., 2m$ and color $i-(n-2m-1)$ to v_i; $i = n-2m, ..., n-1$. The vertices v_i and v_j are adjacent if $i-m \leq j \leq i+m$ where addition is taken modulo n . A vertex v_i and a vertex v_j have the same color if $i = j-(n-2m-1)$ for $i \in \{1, ..., 2m\}$ and $j \in \{n-2m, ..., n-1\}$, so $i-2m-1 \geq j = i+n-2m-1 \geq i+4m+1-2m-1 = i+2m$. Hence two vertices with a same color are not adjacent, which implies that the partial coloring is proper. Also, we can see easily that the vertices $v_i; i = 1, .., m$ and the vertices $v_i; i = n-m, ..., n-1$ with v_0 are b-dominating vertices for this partial proper coloring. Finally, extend this partial proper coloring to a proper coloring of $H_{2m,n}$ as follow. Color the remaining vertices in the whole graph in arbitrary order, assigning to each vertex a color from $\{0, 1, ..., 2m\}$ different from the colors already assigned to its neighbors which is in fact an extension by a standard greedy coloring algorithm. We obtain a b-coloring with $2m+1$ colors in which the vertices $v_0, v_1, ..., v_m, v_{m-n}, ...v_{n-1}$ are b-dominating vertices.

Case 3: $\left\lceil \dfrac{5m+3}{2} \right\rceil \leq n \leq 4m$.

First, we show that $b(H_{2m,n}) \leq 2m - \left\lfloor \dfrac{4m-n}{3} \right\rfloor$. Suppose to the contrary that $H_{2m,n}$ admits a b-coloring with k colors, $k \geq 2m - \left\lfloor \dfrac{4m-n}{3} \right\rfloor + 1$.

Claim 1 *There exists at least one color class with one vertex.*

Proof of Claim 1: Otherwise every color class has at least two vertices, so $n \geq 2k \geq 4m - 2\left\lfloor \dfrac{4m-n}{3} \right\rfloor + 2$ and since $\left\lfloor \dfrac{4m-n}{3} \right\rfloor \leq \dfrac{4m-n}{3}$, $n \geq 4m+6$, a

contradiction.□

Let $0, ..., k-1$ be the colors used by a b-coloring of $H_{2m,n}$. Without loss of generality let v_0 be the only vertex with color 0. So, v_0 is a b-dominating vertex of color 0 and there are at least $k-1$ other b-dominating vertices with distinct colors adjacent to v_0.

Let $X = \{x_1, x_2, ..., x_m\}$ and $Y = \{y_1, y_2, ..., y_m\}$ be the neighborhood of v_0 in each direction around the circle in right and left direction of v_0 respectively.

Let x_i (resp. y_j) be the lastest b-dominating vertex in X (resp. Y). Set $A = \{x_k \in X : k \leq i\}$ and $B = \{y_k \in X : k \leq j\}$. Let $Z = V \setminus (\{v_0\} \cup X \cup Y)$ be the set of the non neighborhood of v_0. Let V_{ij} (resp. $\overline{V_{ij}}$) be the set of vertices between x_i and y_j in left (resp. right) direction of x_i around the circle, that is $v_0 \in V_{ij}$ and $v_0 \notin \overline{V_{ij}}$.

If $3m+1 \leq n \leq 4m$, then $|Z| = n - (2m+1) \geq m$; so $\left|\overline{V_{ij}} \cup \{x_i, y_j\}\right| \geq |Z| + 2 \geq m+2$. Also we have

$$\begin{aligned} |A| + |B| + 1 \geq k &\geq 2m - \left\lfloor \frac{4m-n}{3} \right\rfloor + 1 \geq 2m - \frac{4m-n}{3} + 1 \\ &\geq \frac{2m+n}{3} + 1 \geq \frac{2m+3m+4}{3} = \frac{5m+4}{3} = m + \frac{2m+4}{3} \\ &\geq m+2, \end{aligned}$$

then $|V_{ij} \cup \{x_i, y_j\}| \geq m+2$. Hence x_i is not adjacent to y_j.

The lastest b-dominating vertex x_i in A needs at least $k-m$ colors which are assigning to some b-dominating vertices at the end of B, so we need at least $k-m$ distinct vertices with this colors which belong to $V(H_{2m,n}) - (\{v_0\} \cup A \cup B)$ and which are adjacent to x_i. Let A' be the set of this vertices required by x_i. Similarly the lastest b-dominating vertex y_j in B needs at least $k-m$ colors which are assigning to b-dominating vertices at the end of A, so we need at least $k-m$ distinct vertices with this colors which belong to $V(H_{2m,n}) - \{v_0\} \cup A \cup B$ and which are adjacent to y_j. Let B' be the set of vertices required by y_j. Since the colors needed by x_i are in the neighborhood of y_j and the colors needed by y_j are in the neighborhood of x_i, this colors are different, so A' and B' are disjoint. Thus

$$\begin{aligned} n &\geq |A| + |B| + 1 + |A'| + |B'| \geq k + 2(k-m) = 3k - 2m \\ &\geq 3(2m - \left\lfloor \frac{4m-n}{3} \right\rfloor + 1) - 2m = 4m - 3\left\lfloor \frac{4m-n}{3} \right\rfloor + 3 \\ &\geq 4m - 4m + n + 3 = n + 3, \end{aligned}$$

a contradiction.

Now we suppose that $\left\lceil \frac{5m+3}{2} \right\rceil \leq n \leq 3m$.

Claim 2 *Each set X and Y contains at least $\frac{m+2}{2}$ b-dominating vertices.*

Proof of Claim 2: To see this, assume that X or Y contains at most $\frac{m}{2}$ b-dominating vertices. Then

$$2m - \left\lfloor \frac{4m-n}{3} \right\rfloor + 1 \leq k \leq \frac{3m}{2} + 1$$

which implies that $n \leq \frac{5m}{2}$, a contradiction.□

Claim 3 *All the vertices of $A \cup B$ are b-dominating.*

■

Proof. Proof of Claim 3: First we prove that x_1 is a b-dominating vertex, Suppose that x_1 with the color c_1 is not b-dominating, so in the neighborhood of x_1 there exists some missed color c_1', which implies that $Y \setminus \{y_m\}$ does not contain colors c_1' and c_1. Since v_0 is the only b-dominating vertex with his color, the color of y_m must be c_1'. Hence $X \cup Y$ does not contain a b-dominating vertex with the color c_1, a contradiction. Similarly we can prove that y_1 is a b-dominating vertex. Now we suppose that A contains a non b-dominating vertex x_l with the color c_l. Let x_p and x_q; $p < l < q$ the nearest b-dominating vertices in each direction around the circle; in right and left direction of x_l respectively. We denote by F the set of non b-dominating vertices between x_p and x_q; which contains at least x_l. By Claim 2, it is clear that $|F| \leq \frac{m-2}{2}$. As x_l is a non b-dominating vertex, so in the neighborhood of x_l there exists some missed color c_l', which implies that in V there is only one vertex of color c_l', because the color c_l' does not exist in $N[x_l]$, so it bellow to $M = V \setminus N[x_l]$. Since

$$|M| = |V \setminus N[x_l]| = |V| - |N[x_l]| = n - 2m - 1 \leq 3m - 2m - 1 = m - 1,$$

the subgraph $G[M]$ induced by M is a clique. Therefore there is one vertex y_h of color c_l' in $G[M]$ and y_h is a b-dominating vertex, so $y_h \in B$. However x_p and x_q are adjacent to y_h. Then

$$n = |V_{ph}| + \left|\overline{V_{qh}}\right| + |\{x_p, x_q\}| + |F| \leq 2m + 1 + |F| \leq 2m + 1 + \frac{m-2}{2} = \frac{5m}{2},$$

a contradiction.□

Let B' be the set of b-dominating vertices in B such that no color in B' is repeated in A. Let y_t be the last vertex of Y, whose color does not appear in A. y_t exists, otherwise $k = 1 + |A| \leq m + 1$, a contradiction. So y_t is a b-dominating vertex and $y_t \in B'$ and we have

$$|A| + |B'| + |\{v_0\}| \geq k \geq 2m - \left\lfloor \frac{4m-n}{3} \right\rfloor + 1 \geq \frac{2m+n}{3} + 1 \geq \frac{3m}{2} + 2.$$

Claim 4 *x_i is not adjacent to y_t.*

Proof of Claim 4: If x_i is adjacent to y_t, then two cases arise: Assume that $V_{it} \cup \{x_i, y_t\}$ induce a clique, thus $|V_{it} \cup \{x_i, y_t\}| \leq m+1$ (the cardinality maximum of a clique in $H_{2m,n}$ is $m+1$). Since $|V_{it} \cup \{x_i, y_t\}| \geq |A| + |B'| + |\{v_0\}| \geq k$ and $k \geq \frac{3m}{2} + 2$, $|V_{it} \cup \{x_i, y_t\}| \geq \frac{3m}{2} + 2$, a contradiction. Thus $V_{it} \cup \{x_i, y_t\}$ does not induce a clique, so $\overline{V_{it}} \cup \{x_i, y_t\}$ induce a clique. In this case since every vertex of A is b-domnating, y_t is adjacent to all vertices of A (otherwise it can not have the color of y_t). Hence $H_{2m,n}$ is a clique which contradicts hypothesis. $\square$

The lastest b-dominating vertex x_i in A needs at least $k-m$ colors which are assigning to some b-dominating vertices at the end of B', so we need at least $k-m$ distinct vertices with this colors which belong to $V(H_{2m,n}) - (\{v_0\} \cup A \cup B')$ and which are adjacent to x_i. Let A' be the set of this vertices required by x_i. Similarly the lastest b-dominating vertex y_t in B' needs at least $k-m$ colors which are assigning to b-dominating vertices at the end of A, so we need at least $k-m$ distinct vertices with this colors which belong to $V(H_{2m,n}) - \{v_0\} \cup A \cup B'$ and which are adjacent to y_t. Let B'_1 be the set of vertices required by y_t. Since the colors needed by x_i are in the neighborhood of y_t and the colors needed by y_t are in the neighborhood of x_i, this colors are different, so A' and B'_1 are disjoint. Thus

$$\begin{aligned} n &\geq |A| + |B'| + |A'| + |B'_1| + 1 \geq k + 2(k-m) = 3k - 2m \\ &\geq 3(2m - \left\lfloor \frac{4m-n}{3} \right\rfloor + 1) - 2m = 4m - 3\left\lfloor \frac{4m-n}{3} \right\rfloor + 3 \\ &\geq 4m - 4m + n + 3 = n + 3, \end{aligned}$$

a contradiction. So in all case, if $\left\lceil \frac{5m+3}{2} \right\rceil \leq n \leq 4m$, then $b(H_{2m,n}) \leq 2m - \left\lfloor \frac{4m-n}{3} \right\rfloor$.

Now, we give a b-coloring of $H_{2m,n}$ with $2m - \left\lfloor \frac{4m-n}{3} \right\rfloor$, when $\left\lceil \frac{5m+3}{2} \right\rceil \leq n \leq 4m$. Let $v_1, v_2, ..., v_n$ be vertices of $H_{2m,n}$ in this order around the circle. Set $k = 2m - \left\lfloor \frac{4m-n}{3} \right\rfloor$, then $n \leq 2k$, otherwise $n > 2k$ implies that $n > 4m$, a contradiction. Since $n \leq 2k$, we can color all vertices of $H_{2m,n}$ by the following b-coloring, assign color i to v_i; $i = 1, ..., k$ and color $i-(n-k)$ to v_i; $i = k+1, ..., n$, according to the ordering of vertices. The vertices v_i and v_j are adjacent if $i - m \leq j \leq i + m$ where addition is taken modulo $n+1$. A vertex v_i and a vertex v_j have the same color if $i = j - (n-k)$ for $i \in \{1, ..., k\}$ and

$j \in \{k+1, ..., n\}$. Since

$$|j-i| = n-k = n-2m+\left\lfloor \frac{4m-n}{3} \right\rfloor > n-2m+\frac{4m-n}{3}-1$$
$$= \frac{3n-6m+4m-n-3}{3} = \frac{2n-2m-3}{3}$$
$$\geq \frac{2\frac{5m+3}{2}-2m-3}{3} = m,$$

two vertices with a same color are not adjacent, which implies that the coloring is proper. Also, we can see easily that the vertices $v_i; i = 1, .., m+1$ and the vertices $v_i; i = n-k+m+2, ..., n$; with $k \leq m+2$, are b-dominating vertices for this proper coloring.

Case 4: $2m+2 \leq n < \left\lceil \frac{5m+3}{2} \right\rceil$.

To show that $b(H_{2m,n}) \geq n-m-1$, we construct a b-coloring with $n-m-1$ colors as follow. Let $v_1, v_2, ..., v_n$ be vertices of $H_{2m,n}$ in this order around the circle. Set $k = n-m-1$, then $n \leq 2k$, otherwise $n > 2k$ implies that $n < 2m+2$, a contradiction. Since $n \leq 2k$, we can color all vertices of $H_{2m,n}$ by the following b-coloring, assign color i to v_i; $i = 1, ..., k$ and color $i-(n-k)$ to v_i; $i = k+1, ..., n$, according to the ordering of vertices. The vertices v_i and v_j are adjacent if $i-m \leq j \leq i+m$ where addition is taken modulo $n+1$. A vertex v_i and a vertex v_j have the same color if $i = j-(n-k)$ for $i \in \{1, ..., k\}$ and $j \in \{k+1, ..., n\}$. Since

$$|j-i| = n-k = n-n+m+1 = m+1,$$

two vertices with a same color are not adjacent, which implies that the coloring is proper. Also, we can see that the vertices $v_i; i = 1, .., m+1$ and the vertices $v_i; i = n-k+m+2, ..., n$; with $k \leq m+2$, are b-dominating vertices for this proper coloring, which completes the proof of Theorem 1. ■

Proposition 2 *Let $H_{2m,2m+3}$ be the Harary graph. Then*

$$n-m-1 \leq b(H_{2m,2m+3}) \leq \left\lfloor \frac{6m+9}{5} \right\rfloor$$

And this bounds are sharp.

Proof. Let c be an arbitrary b-coloring of $H_{2m,2m+3}$. The first inequality leads from Theorem 1. Let $v_0, v_1, ..., v_{2m+2}$ be vertices of $H_{2m,2m+3}$ in this order around the circle. Now we prove the second inequality. Since $|Z| = |V \setminus (\{v_0\} \cup X \cup Y)| = 2$, each color is repeated at most twice. Let k_1 (resp. k_2) be the number of color classes with one vertex (resp. two vertices). By 1-class (resp. 2-class) we denote the color class with one vertex (resp. two vertices). Then $n = k_1 + 2k_2$ and $b = k_1 + k_2 = n - k_2 = 2m + 3 - k_2$.

If $k_1 = 1$, then $n-1 = 2m+2 = 2k_2$ which implies that $k_2 = m+1$. So $b = n-m-1 = m+2$.

Let $k_1 \geq 3$, (k_1 is odd integer since the order of $H_{2m,2m+3}$ is odd and $2m+3 = k_1 + 2k_2$).

We prove that the two nearest neighbors around the circle of a b-dominating vertex which belongs to an 1-class are b-dominating vertices and everyone is contained in an 2-class. Let v_0 be the vertex which belongs to an 1-class, v_1 and v_{n-1} its nearest neighbors around the circle and v_{m+1}, v_{m+2} its non neighbors with $c(v_{m+1}) = a$ and $c(v_{m+2}) = b$. We must have $c(v_1) = b$ and $c(v_{n-1}) = a$ with v_1 and v_{n-1} b-dominating vertices, because the vertices v_{m+1} and v_{m+2} can not be adjacent to the color of v_0. Therefore two b-dominating vertices where each one is in an 1-class are not consecutive around the circle. Also we prove that between two b-dominating vertices where each one belongs to an 1-class, there exists at least two b-dominating vertices where each one belongs to an 2-class. Assume to the contrary that there exists one exactly b-dominating vertex which belongs to an 2-class. Without loss of generality, let v_0 and v_2 be the b-dominating vertices where each one belongs to an 1-class, so v_1 is a vertex which belongs to an 2-class. It is obvious to verify that this b-coloring is impossible. Hence $k_2 \geq 2k_1$ and since $n = k_1 + 2k_2$, $k_2 \geq \dfrac{2n}{5}$. Consequently $b = 2m + 3 - k_2 \leq \left\lfloor \dfrac{3n}{5} \right\rfloor = \left\lfloor \dfrac{6m+9}{5} \right\rfloor$.

Let c be a b-coloring with $b(H_{2m,2m+3})$ colors (a mapping from V to the set of positive integers (colors)). We give examples which show that the bounds of Proposition 2 are sharp.

For each value of m we have checked the b-coloring given. (In each case the b-dominating vertices are marked by *).

1. $m = 1, n = 5, b(H_{2,5}) = n - m - 1 = \left\lfloor \dfrac{6m+9}{5} \right\rfloor = 3$

vertices	v_0^*	v_1^*	v_2^*	v_3	v_4
b-coloring	0	1	2	1	2

2. $m = 6, n = 15, b(H_{12,15}) = n - m = \left\lfloor \dfrac{6m+9}{5} \right\rfloor = 9$

vertices	v_0^*	v_1^*	v_2	v_3	v_4^*	v_5^*	v_6^*	v_7	v_8	v_9^*	v_{10}^*	v_{11}^*
b-coloring	0	1	5	7	2	3	4	8	1	5	6	7

v_{12}	v_{13}	v_{14}^*
2	4	8

3. $m = 11, n = 25, b(H_{22,25}) = n - m + 1 = \left\lfloor \dfrac{6m+9}{5} \right\rfloor = 15$

vertices	v_0^*	v_1^*	v_2	v_3	v_4^*	v_5^*	v_6^*	v_7	v_8	v_9^*	v_{10}^*	v_{11}^*
b-coloring	0	1	8	10	2	3	4	11	13	5	6	7

v_{12}	v_{13}	v_{14}^*	v_{15}^*	v_{16}^*	v_{17}	v_{18}	v_{19}^*	v_{20}^*	v_{21}^*	v_{22}	v_{23}	v_{24}^*
14	1	8	9	10	2	4	11	12	13	5	7	14

4. $m = 16, n = 35, b(H_{32,35}) = n - m + 2 = \left\lfloor \frac{6m+9}{5} \right\rfloor = 21$

vertices	v_0^*	v_1^*	v_2	v_3	v_4^*	v_5^*	v_6^*	v_7	v_8	v_9^*	v_{10}^*	v_{11}^*
b-coloring	0	1	11	13	2	3	4	14	16	5	6	7

v_{12}	v_{13}	v_{14}^*	v_{15}^*	v_{16}^*	v_{17}	v_{18}	v_{19}^*	v_{20}^*	v_{21}^*	v_{22}	v_{23}	v_{24}^*
17	19	8	9	10	20	1	11	12	13	2	4	14

v_{25}^*	v_{26}^*	v_{27}	v_{28}	v_{29}^*	v_{30}^*	v_{31}^*	v_{32}	v_{33}	v_{34}^*
15	16	5	7	17	18	19	8	10	20

5. $m = 21, n = 45, b(H_{42,45}) = n - m + 3 = \left\lfloor \frac{6m+9}{5} \right\rfloor = 27$

vertices	v_0^*	v_1^*	v_2	v_3	v_4^*	v_5^*	v_6^*	v_7	v_8	v_9^*	v_{10}^*	v_{11}^*
b-coloring	0	1	14	16	2	3	4	17	19	5	6	7

v_{12}	v_{13}	v_{14}^*	v_{15}^*	v_{16}^*	v_{17}	v_{18}	v_{19}^*	v_{20}^*	v_{21}^*	v_{22}	v_{23}	v_{24}^*
20	22	8	9	10	23	25	11	12	13	26	1	14

v_{25}^*	v_{26}^*	v_{27}	v_{28}	v_{29}^*	v_{30}^*	v_{31}^*	v_{32}	v_{33}	v_{34}^*	v_{35}^*	v_{36}^*	v_{37}
15	16	2	4	17	18	19	5	7	20	21	22	8

v_{38}	v_{39}^*	v_{40}^*	v_{41}^*	v_{42}	v_{43}	v_{44}^*
10	23	24	25	11	13	26

6. $m = 26, n = 55, b(H_{52,55}) = n - m + 4 = \left\lfloor \frac{6m+9}{5} \right\rfloor = 33.$

■

By looking into the disposition of the colors assigned to a b-coloring done on the previous examples, it is easy to generalize these examples, it suffices for this to take $m = 5k + 1; k \in IN^*$, then we have $n = 2m + 3 = 10k + 5$ and $b(G) = \frac{6m+9}{5} = (n - m - 1) + \frac{m-1}{5} = 6k + 3$.

The examples 2-6 given before in the proof of Proposition 2 provide counterexamples to the open problem of Effantin and Kheddouci [5].

References

[1] C. Berge. *Graphs*. North Holland, 1985.

[2] M. Blidia, F. Maffray, Z. Zemir. On b-colorings in regular graphs. *Disc. Appl. Math.* 157 (2009) 1787–1793.

[3] S. Corteel, M. Valencia-Pabon, J.-C. Vera. On approximating the b-chromatic number. *Disc. Appl. Math.* 146 (2005) 106–110.

[4] B. Effantin. The b-chromatic number of power graphs of complete caterpillars, *J. Discrete Math. Sc. Cryptogr.* 8 (2005) 483–502.

[5] B. Effantin, H. Kheddouci. The b-chromatic number of some power graphs. *Discrete Mathematics and Theoretical Computer Science* 6 (2003) 45–54.

[6] B. Effantin, H. Kheddouci. Exact values for the b-chromatic number of a power complete k-ary tree, *J. Discrete Math. Sc. Cryptogr. 8* (2005) 117–129.

[7] A. El-Sahili, M. Kouider. About b-colourings of regular graphs. *Res. Rep.* 1432*, LRI, Univ. Orsay, France*, 2006.

[8] T. Faik. La b-continuité des b-colorations: complexité, propriétés structurelles et algorithmes. *PhD thesis, Univ. Orsay, France*, 2005.

[9] C.T. Hoàng, M. Kouider. On the b-dominating coloring of graphs. *Disc. Appl. Math.* 152 (2005) 176–186.

[10] R.W. Irving, D.F. Manlove. The b-chromatic number of graphs. *Discrete Appl. Math.* 91 (1999) 127–141.

[11] R. Javadi, B. Omoomi. On b-coloring of Kneser graphs, *Disc. Math.* 306 (2009).

[12] R. Javadi, B. Omoomi. On b-coloring of cartesian product of graphs, *Ars Combinatoria, to appear.*

[13] M. Kouider. b-chromatic number of a graph, subgraphs and degrees, *Res. Rep.* 1392*, LRI, Univ. Orsay, France*, 2004.

[14] M. Kouider, M. Mahéo. Some bounds for the b-chromatic number of a graph, *Disc. Math.* 256 (2002) 267–277.

[15] M. Kouider, M. Mahéo. The b-chromatic number of the cartesien product of the graphs, *Studia Sci. Math. Hungar* 14 (2007) 49–55.

[16] M. Kouider, M. Zaker. Bounds for the b-chromatic number of some families of graphs. *Disc. Math.* 306 (2006) 617–623.

[17] J. Kratochvíl, Zs. Tuza, M. Voigt. On the b-chromatic number of graphs. *Lecture Notes in Computer Science* (Graph-Theoretic Concepts in Computer Science: 28th International Workshop, WG 2002) 2573 (2002), 310–320.

[18] D.F. Manlove. Minimaximal and maximinimal optimisation problems: a partial order-based approach. *PhD thesis. Tech. Rep. 27, Comp. Sci. Dept., Univ. Glasgow, Scotland,* 1998.

[19] D.B. West. Introduction to Graph Theory, second edition, Prentice-Hall Upper Saddle River, NJ, 2001.

Double domination edge removal critical graphs

[1]Mostafa Blidia, [1]Mustapha Chellali, [2]Soufiane Khelifi and [3]Frédéric Maffray
[1]LAMDA-RO, Department of Mathematics Université de Blida Algérie
[2]Laboratoire LMP2M, Université de Médéa Ain D'heb 26000 Médéa, Algérie.
[3]C.N.R.S., Laboratoire G-SCOP, UJF, 46 Avenue Félix Viallet, 38031 Grenoble Cedex, France.

Abstract

Let G be a graph without isolated vertices. A set $S \subseteq V(G)$ is a double dominating set if every vertex in $V(G)$ is adjacent to at least two vertices in S. G is said edge removal critical graph with respect to double domination, if the removal of any edge increases the double domination number. In this paper, we first give a necessary and sufficient conditions for $\gamma_{\times 2}$-critical graphs. Then we provide a constructive characterization of $\gamma_{\times 2}-$critical trees.

1 Introduction

In a graph $G = (V(G), E(G))$, the *open neighborhood* of a vertex $v \in V(G)$ is $N_G(v) = N(v) = \{u \in V \mid uv \in E(G)\}$, the *closed neighborhood* is $N_G[v] = N[v] = N(v) \cup \{v\}$ and the *degree* of v, denoted by $\deg_G(v)$, is the size of its open neighborhood. A vertex with degree one in a graph G is called a *pendent vertex* or a *leaf*, and its neighbor is called its *support.* An edge incident to a leaf in a graph G is called a *pendent edge*. We let $S(G), L(G)$ be the set of support vertices and leaves of G, respectively. If $A \subseteq V(G)$, then $G[A]$ is the graph induced by the vertex set A. The *diameter* $\mathrm{diam}(G)$ of a graph G is the maximum distance over all pairs of vertices of G. We denote by K_n the *complete graph* of order n, and by $K_{m,n}$ the *complete bipartite graph* with partite sets X and Y such that $|X| = m$ and $|Y| = n$. A star of order $n+1$ is $K_{1,n}$. A subdivided star $K^*_{1,n}$ is the graph obtained by subdividing each edge of a star $K_{1,n}$ once. A graph is k-regular if all its vertices have degree k. The path and the cycle on n vertices are denoted by P_n and C_n, respectively.

A subset S of vertices of $V(G)$ is a *dominating set* of G if every vertex in $V(G) - S$ is adjacent to a vertex in S, and $S \subseteq V$ is a *double dominating set* of G, abbreviated *DDS*, if every vertex in $V - S$ has at least two neighbors in S and every vertex of S has a neighbor in S. The *double domination number* $\gamma_{\times 2}(G)$ is the minimum cardinality of a double dominating set of G. A double dominating set of G with minimum cardinality is called a $\gamma_{\times 2}(G)$-set. Double domination

was introduced by Harary and Haynes [4] and is studied for example in [1, 3, 4]. For a comprehensive survey of domination in graphs and its variations, see [5, 6].

Given a graph, a new graph can be obtained by removing or adding an edge. The study of the effects of such modifictions have been considered for several domination parameters. Note that Sumner and Blitch [7] were the first introducing *edge removal critical graphs* for the domination number. For a survey we cite [5] (Chapter 5). In this paper we study the effects on increasing double domination number when an edge is deleted.

2 Preliminary results

We begin by giving a straightforward property of double dominating sets.

Remark 1 *Every DDS of a graph contains all its leaves and support vertices.*

Next we show that the removal of a non-pendent edge of a graph G can increase the double domination number of G by at most two.

Theorem 1 *Let G be a graph without isolated vertices. Then $\gamma_{\times 2}(G) \leq \gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)+2$ for every non-pendent edge $e \in E(G)$.*

Proof. Let $e = xy$ be a non-pendent edge. Clearly every $\gamma_{\times 2}(G-e)$-set is a DDS of G and so $\gamma_{\times 2}(G) \leq \gamma_{\times 2}(G-e)$. Now let S be a $\gamma_{\times 2}(G)$-set. If $S \cap \{x,y\} = \emptyset$, then S is a DDS of $G-e$ and hence $\gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)$. Assume now, without loss of generality, that $S \cap \{x,y\} = \{y\}$. Then since x has two neighbors in S, $S \cup \{x\}$ is a DDS of $G-e$ implying that $\gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)+1$. Finally assume that $\{x,y\} \subseteq S$. We examine three cases.

If each x and y has degree at least two in $G[S]$, then since e is a non pendent edge, S remains a DDS of $G-e$ and so $\gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)$. Assume that both x and y are pendent vertices in $G[S]$. Since $e = xy$ is a non-pendent edge each of x and y has a neighbor in $V-S$. Let $x', y' \in V-S$ be a the neighbors of x and y, respectively. Then $S \cup \{x', y'\}$ is a DDS of $G-e$ and so $\gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)+2$. Finally, assume without loss of generality, that x is a vertex of degree one in $G[S]$ and y has degree at least two in $G[S]$. Since xy is a non-pendent edge, let $x' \in V-S$ be any neighbor of x. Then $S \cup \{x'\}$ is a DDS of $G-e$ and so $\gamma_{\times 2}(G-e) \leq \gamma_{\times 2}(G)+1$. ■

A graph G is said to be edge removal critical (ER-critical) with respect to double domination or just $\gamma_{\times 2}$-critical, if for every edge $e \in E(G)$, $\gamma_{\times 2}(G-e) > \gamma_{\times 2}(G)$. If $G-e$ contains isolated vertices, then we set that $\gamma_{\times 2}(G-e) = +\infty$. Thus nontrivial stars are $\gamma_{\times 2}$-critical. Let $X_G \subset E(G)$ be the set of non-pendent edges in G. Clearly if $X_G = \emptyset$, then $\gamma_{\times 2}(G-e) = +\infty$ and so G is a $\gamma_{\times 2}$-critical graph.

The following Properties are straightforward.

Remark 2 *If G is a $\gamma_{\times 2}$-critical graph, then no two support vertices are adjacent.*

Proposition 1 *Let G be a graph obtained from a subdivided star $K^*_{1,r}$ $(r \geq 2)$ of center y by adding an edge from y to any vertex x of a nontrivial graph G'. Then G is not a $\gamma_{\times 2}$-critical graph.*

Proof. Assume that G is $\gamma_{\times 2}$-critical. Let u_i for $1 \leq i \leq r$ be the support vertices of the subdivided star $K^*_{1,r}$ with center y and let S be any $\gamma_{\times 2}(G)$-set. By Remark 1 each u_i belongs to S. If $y \in S$, then removing any edge yu_i does not increase the double domination number. Thus $y \notin S$ and hence S is a DDS of $G - xy$ implying that $\gamma_{\times 2}(G - e) \leq \gamma_{\times 2}(G)$, a contradiction. It follows that G is not a $\gamma_{\times 2}$-critical graph. ■

Next we give a necessary and a sufficient condition for a graph to be $\gamma_{\times 2}$-critical.

Theorem 2 *G is a $\gamma_{\times 2}$-critical graph if and only if for every $\gamma_{\times 2}(G)$-set S the following conditions hold.*

i) Each component in $G[S]$ is a star.
ii) $V - S$ is an independent set.
iii) Every vertex of $V - S$ has degree two.

Proof. Assume that G is a $\gamma_{\times 2}$-critical graph and let S be any $\gamma_{\times 2}(G)$-set. Observe that $G[S]$ contains no cycle for otherwise removing any edge on the cycle does not increase the double domination number, a contradiction. Thus $G[S]$ is a forest. If $G[S]$ contains a component with diamter at least three, then there exists an edge on the diametrical path of such a component whose removal does not increase the double domination number, a contradiction too. Since $G[S]$ does not contains isolated vertices, every component of $G[S]$ has diameter at most two, that is a star. Now assume that $V - S$ contains two adjacent vertices x, y. Then S remains a DDS for $G - xy$ and so $\gamma_{\times 2}(G - xy) \leq \gamma_{\times 2}(G)$, a contradiction. It follows that $V - S$ is an independent set. Finally assume that a vertex $x \in V - S$ has degree at least three. By item (ii) $N(x) \subset S$, and hence removing any edge incident to x does not increase the double domination number, a contradiction.

Conversely, suppose that for every $\gamma_{\times 2}(G)$-set conditions (i), (ii) and (iii) are satisfied. Assume that G is not $\gamma_{\times 2}$-critical and let uv be an edge of X_G for which $\gamma_{\times 2}(G - uv) = \gamma_{\times 2}(G)$. Let D be a $\gamma_{\times 2}(G - uv)$-set. Clearly D is a DDS of G and since $\gamma_{\times 2}(G - uv) = \gamma_{\times 2}(G)$, D is also $\gamma_{\times 2}(G)$-set. If $\{u, v\}$ $\cap D = \emptyset$, then D is a $\gamma_{\times 2}(G)$-set and $V - D$ is not an independent set in G. Thus D contains at least one of u or v. Assume that $\{u, v\} \subset D$. Then u has a neighbor, say $x \neq v$, in D and likewise, v has a neighbor, say $y \neq u$, in D,

with possibly $x = y$. Then D is a $\gamma_{\times 2}(G)$-set such that $\{u, v, x, y\}$ induces in $G[D]$ either a cycle C_3, C_4 or a path P_4, a contradiction. Thus, without loss of generality, assume that $u \in D$ and $v \notin D$. Then v is dominated at least twice by D in $G - uv$ but then condition (iii) does not hold for D in G since v would have at least three neighbors. In any case D is a $\gamma_{\times 2}(G)$-set for which conditions (i), (ii) and (iii) are not all satisfied. It follows that G is $\gamma_{\times 2}$-critical. ■

As immediate consequence to Theorem 2 we have the following two corollaries.

Corollary 1 *If G is a $\gamma_{\times 2}$-critical graph, then every $\gamma_{\times 2}(G)$-set contains all vertices of degree at least three.*

Corollary 2 *If G is a graph with minimum degree at least three, then G is not $\gamma_{\times 2}$-critical.*

The following observation will be useful for the proof of the next result.

Remark 3 *1)If $n \geq 3$, then $\gamma_{\times 2}(C_n) = \left\lceil \frac{2n}{3} \right\rceil$.*

2)If $n \geq 2$, then $\gamma_{\times 2}(P_n) = \begin{cases} 2n/3 + 1 & \text{if } n \equiv 0 (\mathrm{mod}\, 3) \\ 2\lceil n/3 \rceil & \text{otherwise} \end{cases}$

Proposition 2 *The only $\gamma_{\times 2}$-critical k-regular graphs with $k \geq 2$ are the cycles C_n with $n \equiv 0, 1 \,(\mathrm{mod}\, 3)$.*

Proof. Assume that G is a k-regular $\gamma_{\times 2}$-critical graph. By Corollary 2 $k \leq 2$ and it follows that $k = 2$, that G is a cycle. Using Remark 3 it is a simple exercice to see that the order of G must satisfy $n \equiv 0, 1 \,(\mathrm{mod}\, 3)$. ■

3 $\gamma_{\times 2}$-critical trees

For ease of presentation, we next consider rooted trees. For a vertex v in a (rooted) tree T, we let $C(v)$ and $D(v)$ denote the set of children and descendants, respectively, of v, and we define $D[v] = D(v) \cup \{v\}$. The maximal subtree at v is the subtree of T induced by $D[v]$, and is denoted by T_v. Also, a vertex of degree at least three in T is called a *branch vertex*, and we denote by $B(T)$ the set of such vertices.

Remark 4 *If T is the tree obtained from a tree T' by attaching a vertex to a support vertex, then $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$.*

Lemma 1 *Let T be a tree obtained from a nontrivial tree T' by adding k ($k \geq 1$) paths $P_3 = a_i b_i c_i$ attached by edges $c_i x$ for every i, at a vertex x of T' which belongs to some $\gamma_{\times 2}(T')$-set, then $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 2k$.*

Proof. Let S' be a $\gamma_{\times 2}(T')$-set that contains x. S' can be extended to a DDS of T by adding the vertices a_i, b_i for every i, and so $\gamma_{\times 2}(T) \leqslant \gamma_{\times 2}(T') + 2k$. Now let D be a $\gamma_{\times 2}(T)$-set. By Remark 1, D contains a_i, b_i for every i. If D contains three vertices from $\{c_1, c_2, ..., c_k\}$, say c_1, c_2, c_3, then $x \notin D$ and so $\{x\} \cup D - \{c_1, c_2\}$ is a DDS smaller than D, a contradiction. Thus every $\gamma_{\times 2}(T)$-set contains at most two vertices from $\{c_1, c_2, ..., c_k\}$. Now, without loss of generality, we can assume that $D \cap \{c_1, c_2, ..., c_k\} = \emptyset$ (else we replace such vertices by x or/and a neighbor of x in T'). Hence $x \in D$ to double dominate every c_i, implying that $\gamma_{\times 2}(T') \leqslant \gamma_{\times 2}(T) - 2k$. It follows that $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 2k$. ■

Remark 5 *If T is a tree obtained from a tree T' by attaching a new vertex x to a pendent vertex u whose support vertex v is adjacent to at least one pendent path of order three, then $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$.*

Proof. Let $x_i y_i z_i$ with $1 \leq i \leq k$ be k pendent paths P_3 attached to v by the vertices x_i and S any $\gamma_{\times 2}(T)$-set. Since every $\gamma_{\times 2}(T')$-set can be extended to a DDS of T by adding the set $\{x\}$, $\gamma_{\times 2}(T) \leq \gamma_{\times 2}(T') + 1$. On the other hand, without loss of generality, we may assume that $v \in S$ (if $v \notin S$, then by minimality, $k = 1$ and $x_1 \in S$, and so we can replace x_1 by v in S), hence the set $S' = S \cap T'$ is a DDS of T' and so $\gamma_{\times 2}(T') \leq \gamma_{\times 2}(T) - 1$. It follows that $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$. ■

In order to characterize $\gamma_{\times 2}$-critical trees, we define the family of all trees $\mathcal{F}$ that can be obtained from a sequence $T_1, T_2, ..., T_j$ ($j \geqslant 1$) of trees such that T_1 is a star $K_{1,r}$ with $r \geq 1$, $T = T_j$, and if $j \geqslant 2$, T_{i+1} can be obtained recursively from T_i by one of the operations listed below.

- **Operation $\mathcal{O}_1$:** Add a new vertex and join it by an edge to any support vertex of T_i.
- **Operation $\mathcal{O}_2$:** Add a path P_3 and join by an edge a leaf of P_3 to a support vertex of T_i.
- **Operation $\mathcal{O}_3$:** Add k ($k \geq 1$) paths P_3 and join by edges a leaf of each path P_3 to the same leaf of T_i.
- **Operation $\mathcal{O}_4$:** Add a new vertex u and join it by an edge to a leaf v of T_i whose support neighbor x has degree $k + 2$ and is adjacent to $k \geq 1$ pendent paths P_3 such that every vertex in $N(x) - \{v\}$ has degree two and does not belong to any $\gamma_{\times 2}(T_i)$-set.

Note that we can determine in polynomial time the vertices that are in no minimum double dominating set of a tree [2].

Now we are ready to characterize $\gamma_{\times 2}$-critical trees.

Theorem 3 *A nontrivial tree T is $\gamma_{\times 2}$-critical if and only if $T \in \mathcal{F}$.*

Proof. We proceed by induction on the order of T. Since stars are $\gamma_{\times 2}$-critical, and by Remark 2, double stars are not $\gamma_{\times 2}$-critical since they have two adjacent support vertices. Hence, assume that T has diameter at least four. The smallest tree of diameter four is the path P_5 and it can be obtained from a star $K_{1,1}$ by Operation $\mathcal{O}_3$, and so T belongs to $\mathcal{F}$. Assume now that $\text{diam}(T) = 4$ and $T \neq P_5$. Let x_1-x_2-x_3-x_4-x_5 be the longest path of T. Clearly x_1 and x_5 are leaves and so x_2 and x_4 are their support vertices, respectively. If $\deg_T(x_3) = 2$, then at least one of x_2 and x_4 is a strong support. Then $T \in \mathcal{F}$ and is obtained from P_5 by using Operation $\mathcal{O}_1$. Now we assume that $\deg_T(x_3) \geq 3$. By Remark 2, x_3 cannot be a support vertex. Thus every neighbor of x_3 is a support vertex but then $\gamma_{\times 2}(T - x_2x_3) \leq \gamma_{\times 2}(T)$, contradicting the fact that T is $\gamma_{\times 2}$-critical. Thus assume that $\text{diam}(T) \geq 5$. The smallest tree of diameter five is the path P_6, which can be obtained from a star $K_{1,2}$ by operation $\mathcal{O}_3$, and so it belongs to $\mathcal{F}$.

Let $n \geq 7$ and assume that every $\gamma_{\times 2}$-critical tree T' of order $n' < n$ is in $\mathcal{F}$. Let T be a $\gamma_{\times 2}$-critical tree of order n and let S be any $\gamma_{\times 2}(T)$-set. If any support vertex, say y, of T is adjacent to two or more leaves, then let T' be the tree obtained from T by removing a leaf adjacent to y. By Remark 4, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$. Clearly $X_{T'} = X_T$ and T' is $\gamma_{\times 2}$-critical. By the inductive hypothesis on T', we have $T' \in \mathcal{F}$. It follows that $T \in \mathcal{F}$ because it is obtained from T' by using Operation $\mathcal{O}_1$. Thus we may assume for the next that every support vertex is adjacent to exactly one leaf.

We now root T at leaf x of a longest path. Let u be a vertex at distance $\text{diam}(T) - 1$ from x on a longest path starting at x, and let r be the child of u on this path. Let w_1, v be the parents of u and w_1, respectively. Since u is a support vertex, $\deg_T(u) = 2$ and so by Remark 2, w_1 cannot be a support vertex. On the other hand if $\deg_T(w_1) \geq 3$, then T_{w_1} is a subdivided star. Thus T is a tree obtained from a tree T' and the subdivided star T_{w_1} of center w_1 by adding the edge w_1v, where $v \in V(T')$. But by Proposition 1 T is not $\gamma_{\times 2}$-critical, a contradiction. Thus $\deg_T(w_1) = 2$. We consider the following two cases.

Case 1. v is a support vertex. Let $T' = T - \{r, u, w_1\}$. Then $X_{T'} = X_T - \{vw_1, w_1u\}$ and by Lemma 1, $\gamma_{\times 2}(T') = \gamma_{\times 2}(T) - 2$. Now let e be any edge of $X_{T'}$. Since T is $\gamma_{\times 2}$-critical, the removal of e produces two trees T_1 and T_2 such that $\gamma_{\times 2}(T-e) = \gamma_{\times 2}(T_1)+\gamma_{\times 2}(T_2) > \gamma_{\times 2}(T)$. Without loss of generality, we can assume that $\{r, u, w_1\} \in T_1$. Thus by Lemma 1, $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 2$ (since $e \in E(T')$, the deletion of the edge e in T' provides on one hand T_1' and on the other hand the tree T_2). It follows that $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T_1) - 2 + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T - e) - 2 > \gamma_{\times 2}(T) - 2 = \gamma_{\times 2}(T')$. Consequently the deletion of every edge of $X_{T'}$ increases the double domination number of T' and so T' is $\gamma_{\times 2}$-critical. By the inductive induction, $T' \in \mathcal{F}$ and so $T \in \mathcal{F}$ since it is obtained from T' by Operation $\mathcal{O}_2$.

Case 2: v is not a support vertex. Let $C(v) = \{w_1, w_2, ..., w_k\}$ with $k \geq 1$

. We first assume that $C(v)$ contains no support vertex. If $\deg(v) = 2$, then, without loss of generality, we may assume that $v \in S$ (else replace w_1 by v) and if $\deg(v) \geq 3$, then, by Corollary 1, $v \in S$. Also since every w_i for $i \neq 1$ plays the same role as w_1, every w_i has degree two. Now if $w_1 \in S$, then $\{r, u, w_1, v\}$ induces a path P_4 in S, a contradiction with Theorem 2. Thus S contains no w_i. Now let $T' = T - \underset{1 \leq i \leq k}{\cup} T_{w_i}$. Thus $X_{T'} = X_T - \{vw_i, w_ic(w_i), yv \text{ for } 1 \leq i \leq k\}$, where y is the parent of v and $c(w_i)$ is the unique child of w_i. Proceeding like in Case 1, it can be seen that T' is $\gamma_{\times 2}$-critical. By our inductive hypothesis, $T' \in \mathcal{F}$ and so $T \in \mathcal{F}$ because it is obtained from T' by Operation $\mathcal{O}_3$.

Now assume that v has a child, say w which is a support vertex. Then such a vertex w is unique for otherwise if $w'' \in C(v)$ is a second support vertex, then S is a DDS of $T - vw''$, implying that $\gamma_{\times 2}(T - vw'') \leq \gamma_{\times 2}(T)$, a contradiction with the fact that T is a $\gamma_{\times 2}$-critical tree. Since $\deg_T(v) \geq 3$, then by Corollary 1, $v \in S$. Let w' be the leaf neighbor of w. By item (i) of Theorem 2, every component of $G[S]$ is a star and so $y \notin S$ and no vertex of $C(v)$ different to w is in S. It follows that $\deg_T(y) = 2$ for otherwise by Corollary 1, y belongs to S. Let $C(v) = \{w, w_1, w_2, ..., w_k\}$ with $k \geq 1$. As mentioned above $\deg_T(w_i) = 2$ for every i. Now let T' be the tree obtained from T by removing the leaf w'. By Remark 5, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T^{'}) + 1$. We shall show now that y does not belong to any $\gamma_{\times 2}(T')$-set. Suppose to the contrary that there is a $\gamma_{\times 2}(T')$-set S' that contains y, and since $v, w \in S'$, the set $S = S' \cup \{w'\}$ is a $\gamma_{\times 2}(T)$-set containing y, v, w and w' and so $G[S]$ contains an induced P_4, a contradiction. Further, it is clear that there is no $\gamma_{\times 2}(T')$-set that contains any vertex w_i for every i.

Clearly, $X_{T'} = X_T - \{vw\}$ and to prove that T' is $\gamma_{\times 2}$-critical, consider any edge e of $X_{T'}$. Since T is $\gamma_{\times 2}$-critical, the removal of e produces two trees T_1 and T_2 such that $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) > \gamma_{\times 2}(T)$.

Subcase 1. $e = vw_i$, for $1 \leq i \leq k$. Then $T_2 = T_{w_i}$ and T_1 contains v. Since $e \in E(T')$, the deletion of the edge e in T' provides on one hand T_1' and on the other hand a tree T_2. We need first to show that $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 1$ and we begin by the case $k = 1$. Since every $\gamma_{\times 2}(T_1')$-set can be extended to a DDS of T_1 by adding w', $\gamma_{\times 2}(T_1) \leq \gamma_{\times 2}(T_1') + 1$ implying that $\gamma_{\times 2}(T_1') \geq \gamma_{\times 2}(T_1) - 1$. Suppose now that $\gamma_{\times 2}(T_1') > \gamma_{\times 2}(T_1) - 1$. This implies that v does not belong to any $\gamma_{\times 2}(T_1)$-set. Assume to the contrary that there exists a $\gamma_{\times 2}(T_1)$-set S that contains v. Then $S_1 = S \cap T_1'$ is a DDS of T_1' and so $\gamma_{\times 2}(T_1') \leq \gamma_{\times 2}(T_1) - 1$, a contradiction. Further we show that in this case, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T_1) + 3$. Since every $\gamma_{\times 2}(T_1)$-set can be extended to a DDS of T by adding the set $\{w_1, u, r\}$, it follows that $\gamma_{\times 2}(T) \leq \gamma_{\times 2}(T_1) + 3$. Suppose now that $\gamma_{\times 2}(T) < \gamma_{\times 2}(T) + 3$ and let D be any $\gamma_{\times 2}(T)$-set. By corollary 1 and Remark 1, $v, w, w' \in D$, and so $D_1 = D \cap T_1$ is a DDS of T_1 implying that $\gamma_{\times 2}(T_1) \leq \gamma_{\times 2}(T) - 2$ since $w_1 \notin D$ and $u, r \in D$ and so $\gamma_{\times 2}(T) \geq \gamma_{\times 2}(T_1) + 2$. It follows that $\gamma_{\times 2}(T) = \gamma_{\times 2}(T_1) + 2$, and so D_1 is a $\gamma_{\times 2}(T_1)$-set that contains v, a contradiction. Thus $\gamma_{\times 2}(T) = \gamma_{\times 2}(T_1) + 3$. Now let X_1 be any $\gamma_{\times 2}(T_1)$-set. Since $v \notin X_1$ and $\deg_{T_1}(y) = 2$, $y \in X_1$ and so the set $X_2 = X_1 \cup \{w_1, u, r\}$ is a $\gamma_{\times 2}(T)$-set and the set $(X_2 - w_1) \cup \{v\}$ is a $\gamma_{\times 2}(T)$-set that contains a path $P_4 = w', w, v, y$, a contradiction with the fact that T

is $\gamma_{\times 2}$-critical. Consequently $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 1$. For the case $k \geq 2$, by Remark 5, $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 1$. It follows that for any edge $e = vw_i$ we have $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T_1) - 1 + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T - e) - 1 > \gamma_{\times 2}(T) - 1 = \gamma_{\times 2}(T')$.

Subcase 2. $e = w_i c(w_i)$, for $1 \leq i \leq k$. Then $T_2 = T_{c(w_i)}$ and T_1 contains v. Since $e \in E(T')$, the deletion of the edge e in T' provides on one hand T_1' and on the other hand the tree T_2. Clearly, $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 1$ and it follows that $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T_1) - 1 + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T - e) - 1 > \gamma_{\times 2}(T) - 1 = \gamma_{\times 2}(T')$.

Subcase 3. $e \neq vw_i$ and $e \neq w_i c(w_i)$. Without loss of generality, suppose that T_1 contains v. Since $e \in E(T')$, the deletion of the edge e in T' provides on one hand T_1' and on the other hand a tree T_2. By Remark 5 $\gamma_{\times 2}(T_1') = \gamma_{\times 2}(T_1) - 1$. Hence $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T_1) - 1 + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T - e) - 1 > \gamma_{\times 2}(T) - 1 = \gamma_{\times 2}(T')$.

Consequently the deletion of every edge of $X_{T'}$ increases the double domination number of T' and so T' is $\gamma_{\times 2}$-critical. By the inductive hypothesis, $T' \in \mathcal{F}$ and so $T \in \mathcal{F}$ since it is obtained from T' by Operation $\mathcal{O}_4$.

Conversely, let $T \in \mathcal{F}$. Then T can be obtained from a sequence $T_1, T_2, ..., T_j$ ($j \geqslant 1$) of trees such that T_1 is a star $K_{1,r}$ with $r \geq 1$ and $T = T_j$, and if $j \geq 2$, then T_{i+1} is obtained from T_i by one of the four operations defined above. Proceed by induction on the length j of the sequence of trees needed to construct T. Suppose $j = 1$. Then T is a star $K_{1,r}$ with $k \geq 1$. So T is $\gamma_{\times 2}$-critical.

Assume that the result holds for all trees in T of length less than j in $\mathcal{F}$, where $j \geq 2$. Let T be a tree of length j in $\mathcal{F}$. Thus, $T \in \mathcal{F}$ can be obtained from a sequence $T_1, T_2, ..., T_j$ of T trees. We denote T_{j-1} simply by T'. Applying the inductive hypothesis, $T' \in \mathcal{F}$ is $\gamma_{\times 2}$-critical. We now consider four possibilities depending on whether T is obtained from T' by operation $\mathcal{O}_1$, $\mathcal{O}_2$, $\mathcal{O}_3$ or $\mathcal{O}_4$.

Case 1: T is obtained from T' by operation $\mathcal{O}_1$.

Let v be a support vertex in T' and let u be the vertex attached to v to obtain the tree T. Clearly, $X_T = X_{T'}$ and by Remark 4, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$. We shall show that T is $\gamma_{\times 2}$-critical. Since T' is $\gamma_{\times 2}$-critical, then the removal of any edge $e \in X_{T'}$ produces two trees T_1' and T_2' with $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2') > \gamma_{\times 2}(T')$. On the other hand, $T - e = T_1 \cup T_2'$ where $T_1 = T_1' \cup \{u\}$, and by Remark 4, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T_1') + 1$. Thus, $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2') = \gamma_{\times 2}(T_1') + 1 + \gamma_{\times 2}(T_2') > \gamma_{\times 2}(T') + 1 = \gamma_{\times 2}(T)$. Hence, T is $\gamma_{\times 2}$-critical.

Case 2: T is obtained from T' by operation $\mathcal{O}_2$.

Let v be a support vertex in T' and let xyz be the path attached to v by x to obtain the tree T. Then by Lemma 1, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 2$. To show that T is $\gamma_{\times 2}$-critical, we consider any edge e of the set $X_T = X_{T'} \cup \{vx, xy\}$.

- If e is any edge of $X_{T'}$, then removing e from T' provides the trees T_1' and T_2' such that $v \in T_1'$ with $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T_1') + \gamma_{\times 2}(T_2') > \gamma_{\times 2}(T')$, and from T the trees T_1 and T_2' where $T_1 = T_1' \cup \{x, y, z\}$, and by Lemma 1, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T_1') + 2$. Hence, $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2') = \gamma_{\times 2}(T_1') + 2 + \gamma_{\times 2}(T_2') > \gamma_{\times 2}(T') + 2 = \gamma_{\times 2}(T)$.

- If $e = vx$, then deleting e from T produces the trees T_1 and $T_2 = xyz$ with $\gamma_{\times 2}(T_2) = 3$ and $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2$. Hence, $\gamma_{\times 2}(T - vx) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T) + 1 > \gamma_{\times 2}(T)$.
- Finally, if $e = xy$, then $T - e = T_1 \cup T_2$ where $T_2 = yz$. Clearly $\gamma_{\times 2}(T_2) = 2$ and $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2 + 1 = \gamma_{\times 2}(T) - 1$. Hence, $\gamma_{\times 2}(T - xy) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T) + 1 > \gamma_{\times 2}(T)$.

Consequently, for any edge e of X_T, $\gamma_{\times 2}(T - e) > \gamma_{\times 2}(T)$. So T is $\gamma_{\times 2}$-critical.

Case 3: T is obtained from T' by operation $\mathcal{O}_3$.

Let v be a leaf of T' and let $x_i y_i z_i$ with $1 \leq i \leq k$ be the paths attached to v by the vertices x_i to obtain the tree T. By Lemma 1, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 2k$. First we prove that the support vertex u of v is in every $\gamma_{\times 2}(T)$-set. If u remains a support vertex in T, then u belongs to every $\gamma_{\times 2}(T)$-set. Hence we may assume that u is not a support vertex in T and Let D be a $\gamma_{\times 2}(T)$-set not containing u. Then there is a neighbor of u say w in D with its neighbor w', and v is in D with one vertex x_i for $1 \leq i \leq k$. Without loss of generality, assume that $x_1 \in D$. Then the set $D_1 = (D - \{x_1\}) \cup \{u\}$ is a $\gamma_{\times 2}(T)$-set and so $D' = D_1 \cap T'$ is a $\gamma_{\times 2}(T')$-set containing an induced path $P_4 = w'wuv$, a contradiction. Thus, u belongs to every $\gamma_{\times 2}(T)$-set.

We now consider any edge of the set $X_T = X_{T'} \cup \{uv, vx_i, x_i y_i\}$ with $1 \leq i \leq k$. Note that considering the edges vx_i or $x_i y_i$ is similar to consider vx_1 or $x_1 y_1$.

- If e is an edge of $X_{T'}$, then the removal of e in T' provides the trees T'_1 and T'_2 such that $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T'_1) + \gamma_{\times 2}(T'_2) > \gamma_{\times 2}(T')$. Deleting e from T gives the trees T_1 and T'_2 where $T_1 = T'_1 \cup \{x_i, y_i, z_i\}$ for $1 \leq i \leq k$, and by Lemma 1, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T'_1) + 2k$. Hence, $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T'_2) = \gamma_{\times 2}(T'_1) + 2k + \gamma_{\times 2}(T'_2) > \gamma_{\times 2}(T') + 2k = \gamma_{\times 2}(T)$.
- If $e = uv$, then $T - e$ gives the trees T_1 and T_2 such that T_2 is a tree obtained from a star $K_{1,k}$ where each edge is subdivided twice. Clearly, the set $D_2 = \{v, x_1, y_i, z_i\}$ for $1 \leq i \leq k$ is a $\gamma_{\times 2}(T_2)$-set and so $\gamma_{\times 2}(T_2) = 2k + 2$. Since any $\gamma_{\times 2}(T_1)$-set can be extended to a $\gamma_{\times 2}(T)$-set by adding the set D_2, then $\gamma_{\times 2}(T_1) \geq \gamma_{\times 2}(T) - 2k - 2$. Suppose that D_1 is a $\gamma_{\times 2}(T_1)$-set of cardinality $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2k - 2$. But then $D = D_1 \cup D_2$ is a $\gamma_{\times 2}(T)$-set such that $\{v, x_1, y_1, z_1\} \subset D$. If $u \in D_1$, then the set $D - \{x_1\}$ would be a DDS of cardinality less than $\gamma_{\times 2}(T)$, a contradiction, and if $u \notin S_1$, then D is a $\gamma_{\times 2}(T)$ that does not contain u, a contradiction too. Hence $\gamma_{\times 2}(T_1) \geq \gamma_{\times 2}(T) - 2k - 1$, and so $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) \geq \gamma_{\times 2}(T) - 2k - 1 + 2k + 2 = \gamma_{\times 2}(T) + 1 > \gamma_{\times 2}(T)$.
- If $e = vx_1$, then $T - e$ is formed by the trees T_1 and $T_2 = x_1 y_1 z_1$ with $\gamma_{\times 2}(T_2) = 3$. If v is pendent in T_1 ($k = 1$), then by Lemma 1, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2$. Now if $k \geq 2$, then we can simply see that there exists some $\gamma_{\times 2}(T_1)$-sets that contain v and so by Lemma 1, $_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2$. Hence, $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T) + 1 > \gamma_{\times 2}(T)$.
- If $e = x_1 y_1$, then $T - e$ is formed by the trees T_1 and $T_2 = y_1 z_1$ such that $\gamma_{\times 2}(T_2) = 2$, and since any $\gamma_{\times 2}(T_1)$-set can be extended to a $\gamma_{\times 2}(T)$-

set by adding the set $\{y_1, z_1\}$ then $\gamma_{\times 2}(T_1) \geq \gamma_{\times 2}(T) - 2$. Suppose that $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T) - 2$, and let S_1 be a $\gamma_{\times 2}(T_1)$-set. By Remark 1, $\{v, x_1\} \subset S_1$, and so $S = S_1 \cup \{y_1, z_1\}$ is a $\gamma_{\times 2}(T)$-set. Now if $u \in S_1$, then the set $S - \{x_1\}$ would be a DDS of cardinality less than $\gamma_{\times 2}(T)$, a contradiction, and if $u \notin S_1$, then S is a $\gamma_{\times 2}(T)$ that does not contain u, a contradiction too. Thus $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) \geq \gamma_{\times 2}(T) - 1 + 2 = \gamma_{\times 2}(T) + 1 > \gamma_{\times 2}(T)$.

Hence, for any edge e of X_T, $\gamma_{\times 2}(T - e) > \gamma_{\times 2}(T)$ and T is $\gamma_{\times 2}$-critical.

Case 4: T is obtained from T' by operation $\mathcal{O}_4$.

Let x be a support vertex of T' and v its leaf-neighbor such that $\deg_{T'}(x) = k + 2$ and let $v_i u_i z_i$ with $1 \leq i \leq k$ be the paths attached to x by the vertices x_i, and y a neighbor of x of degree two such that every vertex in $N_{T'}(x) - \{v\}$ does not belong to any $\gamma_{\times 2}(T')$. We attach to v a new vertex u to obtain the tree T. By Remark 5, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$.

Consider now any edge of the set $X_T = X_{T'} \cup \{xv\}$.

- If $e \in X_{T'} - \{xv_i\}$ with $1 \leq i \leq k$ then deleting e in T' produces two trees T'_1 and T'_2 such that $x \in T'_1$, and so $\gamma_{\times 2}(T' - e) = \gamma_{\times 2}(T'_1) + \gamma_{\times 2}(T'_2) > \gamma_{\times 2}(T')$. Deleting e in T gives the trees T_1 and T'_2 where $T_1 = T'_1 \cup \{u\}$. By Remark 5, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T'_1) + 1$, and hence $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T'_2) = \gamma_{\times 2}(T'_1) + 1 + \gamma_{\times 2}(T'_2) > \gamma_{\times 2}(T') + 1 = \gamma_{\times 2}(T)$.
- If $e = xv_i$ with $1 \leq i \leq k$, then let $T' - e = T'_1 \cup T'_2$ such that T'_1 contains x. For the case $k \geq 2$, by Remark 5, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T'_1) + 1$ with $T_1 = T'_1 \cup \{u\}$. Now if $k = 1$, then clearly $\gamma_{\times 2}(T_1) \leq \gamma_{\times 2}(T'_1) + 1$. Suppose that $\gamma_{\times 2}(T_1) < \gamma_{\times 2}(T'_1) + 1$. With the same approach like in subcase 1, it results that x does not belong to any $\gamma_{\times 2}(T_1)$-set and that $\gamma_{\times 2}(T) = \gamma_{\times 2}(T_1) + 3$. Let S_1 be any $\gamma_{\times 2}(T_1)$-set. By Remark 1, $u, v \in S_1$ and since $x \notin S_1$, its neighbor y belongs to S_1 with its neighbor say z, and so the set $S = S_1 \cup \{x, u_1, z_1\}$ is a $\gamma_{\times 2}(T)$-set. Now by Remark 5, $\gamma_{\times 2}(T) = \gamma_{\times 2}(T') + 1$ and since $u, v \in S$, the set $S' = S - \{u\}$ is a $\gamma_{\times 2}(T')$-set that contains the vertices v, x, y and z, and so $G[S']$ induces a path P_4, a contradiction with the fact that T' is $\gamma_{\times 2}$-critical. Consequently, for $k \geq 1$, $\gamma_{\times 2}(T_1) = \gamma_{\times 2}(T'_1) + 1$, and so $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T'_2) = \gamma_{\times 2}(T'_1) + 1 + \gamma_{\times 2}(T'_2) > \gamma_{\times 2}(T') + 1 = \gamma_{\times 2}(T)$.
- If $e = xv$, then let $T - e = T_1 \cup T_2$ such that $T_2 = uv$. Suppose that $\gamma_{\times 2}(T - e) = \gamma_{\times 2}(T_1) + \gamma_{\times 2}(T_2) = \gamma_{\times 2}(T_1) + 2 = \gamma_{\times 2}(T)$ and let S_1 be any $\gamma_{\times 2}(T_1)$-set. Without loss of generality, we may assume that x belongs to S_1 with a vertex from $N_{T_1}(x)$. Then the set $S = S_1 \cup \{v, u\}$ is a $\gamma_{\times 2}(T)$-set, and so the set $S' = S - \{u\}$ is a $\gamma_{\times 2}(T')$-set that contains a neighbor of x, a contradiction. Hence $\gamma_{\times 2}(T - e) > \gamma_{\times 2}(T)$.

Consequently, for any edge e of X_T, $\gamma_{\times 2}(T - e) > \gamma_{\times 2}(T)$. So T is $\gamma_{\times 2}$-critical.

This completes the proof of Theorem 3. ■

References

[1] M. Blidia, M. Chellali and T.W. Haynes, Characterizations of trees with equal paired and double domination numbers. *Discrete Mathematics*, 306 (2006) 1840-1845.

[2] Blidia M., Chellali M. and Khelifi S., "Vertices belonging to all or to no minimum double domination sets of trees", AKCE, 2, No. 1 (2005), pp. 1-9.

[3] M. Chellali and T.W. Haynes, On paired and double domination in graphs. *Utilitas Mathematica*, 67 (2005) 161-171.

[4] F. Harary and T. W. Haynes, Double domination in graphs. *Ars Combin.* 55 (2000) 201-213.

[5] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, *Fundamentals of Domination in Graphs*, Marcel Dekker, New York, 1998.

[6] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater (eds), *Domination in Graphs: Advanced Topics*, Marcel Dekker, New York, 1998.

[7] D. P. Sumner and P. Blitch, Domination critical graphs. *J. Combin. Theory Ser.* B 34 (1983) 65–76.

[8] H. B. Walikar and B. D. Acharya, Domination critical graphs. *Nat. Acad Sci. Lett.* 2 (1979) 70-72.

Requêtes non standards

OptAssist : outil d'assistance pour l'optimisation des entrepôts de données relationnels

Kamel Boukhalfa[1], Ladjel Bellatreche[2], and Zaia Alimazighi[1]

[1] USTHB University - Algiers - Algeria
boukhalk@ensma.fr, alimazighi@wissal.dz
[2] LISI/ENSMA - Poitiers University - France
bellatreche@ensma.fr

Résumé Pour optimiser son entrepôt, l'administrateur est amené à effectuer une conception physique. Durant cette phase, il doit effectuer de multiples choix comme les techniques d'optimisation à sélectionner, leurs algorithmes de sélection et les valeurs des paramètres de ces algorithmes ainsi que les attributs et les tables utilisés par certaines de ces techniques. Nous montrons dans cet article la nature des difficultés rencontrées par l'administrateur durant la conception physique. Nous présentons par la suite un outil d'aide qui permet à l'administrateur d'effectuer les bons choix d'optimisation. Nous montrons l'utilisation interactive de cet outil sur un entrepôt de données issu du Benchmark APB-1.

Key words: Optimisation, Entrepôts de données, Conception physique, Fragmentation horizontale, Index de jointure binaires

1 Introduction

Les principales caractéristiques des entrepôts de données sont leur grande taille et la complexité des requêtes décisionnelles dues aux opérations de sélection, de jointure et d'agrégration. Ces caractéristiques ont rendu la tâche d'administration de plus en plus complexe. Traditionnellement, dans les applications de gestion de bases de données de type OLTP (On-Line Transaction Processing), la tâche d'un administrateur était principalement concentrée sur la gestion des utilisateurs et l'utilisation d'un nombre restreint de techniques d'optimisation comme les index et les vues.

L'optimisation du temps d'exécution de requêtes constitue une exigence primordiale des utilisateurs de l'entrepôt. Pour satisfaire cette exigence, l'administrateur de l'entrepôt de données (AED) doit effectuer une conception physique qui devient cruciale pour garantir une bonne performance. La conception physique doit déterminer comment une requête doit être exécutée efficacement sur l'entrepôt de données. Pour cela, l'AED dispose d'un ensemble de techniques d'optimisation comme la fragmentation verticale, horizontale, les index, etc. Il pourra utiliser une seule technique ou en combiner plusieurs afin d'avoir une meilleure performance. Plusieurs algorithmes de sélection sont disponibles pour chaque technique choisie, chacun caractérisé par un ensemble de paramètres à

régler. Pour certaines techniques, plusieurs objets de l'entrepôt sont candidats (généralement des tables et des attributs). Pour bien mener sa conception physique, l'AED est confronté à effectuer plusieurs choix liés à : (1) les techniques d'optimisation, (2) la nature de sélection, (3) les algorithmes de sélection et leurs paramètres et (4) les tables et attributs candidats (voir figure 1).

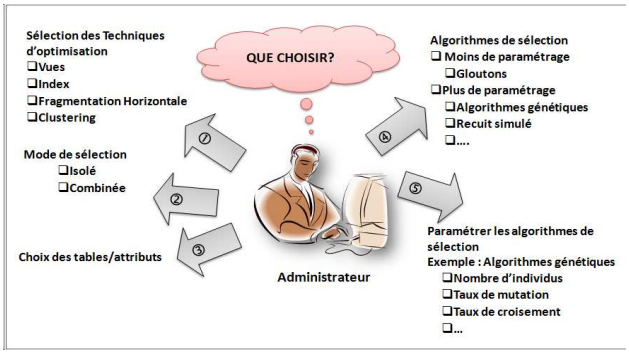


Figure 1. Choix effectués par l'administrateur

Choix des techniques d'optimisation : Si nous explorons la littérature et les éditeurs de gestion de bases de données commerciaux, nous trouvons une large panoplie de techniques d'optimisation qui peuvent être redondantes ou non. Les *techniques redondantes* nécessitent un espace de stockage et un coût de maintenance (les vues matérialisées, les index avancés, la fragmentation verticale, etc.) par contre les *techniques non redondantes* ne nécessitant ni espace de stockage ni coût de maintenance (la fragmentation horizontale, le traitement parallèle, etc.). Pour optimiser les requêtes définies sur l'entrepôt, l'AED peut choisir une ou plusieurs techniques parmi ces deux catégories. Ce choix est souvent difficile dû au fait que certaines techniques sont bénéfiques pour certaines requêtes et non pas pour d'autres.

Le choix du mode de sélection : En présence de plusieurs techniques d'optimisation, l'AED dispose de deux modes de sélection : la *sélection isolée* et la *sélection multiple*. Dans la sélection isolée, il choisit une seule technique qui pourra être redondante s'il dispose d'assez d'espace et peu de mises à jour par exemple, sinon il pourra choisir une technique non redondante. La sélection isolée a été largement étudiée [1,2,3,4,5], mais elle est souvent insuffisante pour une meilleure optimisation de l'entrepôt. La sélection multiple permet de sélectionner plusieurs techniques à la fois. Elle est principalement motivée par les fortes similarités entre les techniques d'optimisation. Les travaux majeurs dans cette catégorie sont principalement concentrés sur la sélection des vues matérialisées et des index [6,7,8].

Choix et paramétrage des algorithmes de sélection : Une fois que les techniques d'optimisation utilisées sont choisies, l'AED est confronté au problème de choix de leurs algorithmes de sélection. Pour chaque mode de sélection, isolée ou multiple, un large choix d'algorithmes est possible. Ces algorithmes sont de natures diverses qui vont des simples algorithmes comme les algorithmes gloutons aux algorithmes plus complexes comme les algorithmes basés sur la programmation linéaire, les algorithmes génétiques, les colonies de fourmis, etc. Certains algorithmes possèdent peu de paramètres comme les algorithmes gloutons. Par contre d'autres possèdent plusieurs paramètres qu'il faut régler et configurer pour avoir une bonne performance.

Choix des attributs et tables candidats : Les entrepôts de données relationnels sont généralement modélisés par un schéma en étoile composé d'une table de faits et un ensemble de tables de dimension. Pour certaines techniques comme la fragmentation horizontale (FH), verticale et les index, plusieurs tables et attributs sont candidats pour être utilisés par ces techniques. L'AED est confronté dans certains cas à choisir un sous-ensemble de tables et d'attributs candidats parmi l'ensemble initial. Ce choix est souvent effectué pour réduire la complexité du problème de sélection.

Comme nous venons de voir, la tâche de l'AED devient de plus en plus complexe vu le nombre important de choix à effectuer, d'où la nécessité de développement d'outils d'aide. Ces outils doivent assister l'AED pour effectuer les bons choix d'administration. Le présent article est organisé en 5 sections. La section 2 présente un état de l'art sur les outils d'aide développés pour assister l'administrateur dans sa tâche d'optimisation. La section 3 présente l'architecture générale de l'outil que nous proposons. La section 4 est consacrée à la présentation des fonctionnalités de l'outil appliquées sur un entrepôt de données. Enfin, la section 5 conclut le papier et présente quelques perspectives.

2 Etat de l'art

Pour assister l'AED dans sa tâche d'optimisation de la couche physique, certains outils ont été développés. La plupart des outils existant ont été proposés par les éditeurs des grands SGBD commerciaux dans le cadre de l'auto-administration des bases de données. Parmi ces outils, nous pouvons citer *Oracle SQL Acces Advisor* [9], *DB2 Design Advisor* [10] et *Microsoft Database Tuning Advisor* [11].

[9] propose l'outil *SQL Access Advisor* qui offre un ensemble complet de conseils sur la manière d'optimiser la conception d'un schéma pour maximiser les performances d'une application. Cet outil est un assistant automatisant certains aspects de la conception physique et de tuning [3] pratiquées manuellement sur les bases de données Oracle. L'outil analyse la charge de requêtes et propose des recommandations pour créer de nouveaux index si nécessaire, de supprimer

3. Le tuning est un ensemble d'activités utilisées pour optimiser les performances d'une base ou entrepôt de données suite à des évolutions de ces derniers

les index inutilisés, de créer de nouvelles vues matérialisées, etc. Les recommandations générées sont accompagnées par une évaluation quantifiée des gains de performance attendus ainsi que des scripts nécessaires pour les implémenter. [10] propose l'outil *DB2 Design Advisor* qui fait partie de *DB2 V8.2* et constitue une amélioration de l'outil *DB2 Index Advisor Tool* [12] qui permet de sélectionner un ensemble d'index. L'outil permet d'optimiser un ensemble de requêtes en proposant un ensemble de recommandations. Ces recommandations concernent quatre techniques d'optimisation : les index, les vues matérialisées, la FH et le clustering. L'outil *Microsoft Database Tuning Advisor* (DTA) est issu du projet *Microsoft Research AutoAdmin*. *DTA* peut fournir des recommandations intégrées pour les index définis sur les tables de base, les vues, les index définis sur les vues et le partitionnement horizontal. Il prend comme entrées un ensemble de BD sur un serveur, une charge de requêtes, les techniques d'optimisation à sélectionner ainsi qu'un ensemble de contraintes, comme l'espace de stockage des techniques redondantes. Il donne en sortie un ensemble de recommandations sur les index, les vues ainsi que la FH.

La plupart des outils que nous venons de présenter ont été proposés dans le cadre de l'auto-administration des bases de données et sont généralement spécifiques à un SGBD donné. Ils sont caractérisés aussi par l'utilisation de l'optimiseur de requêtes pour évaluer la qualité des techniques sélectionnées. Cela représente une tâche supplémentaire de l'optimiseur et pourra engendrer une détérioration du temps de réponses des requêtes.

En voulant automatiser l'administration et le tuning des bases et entrepôts de données, les auteurs de ces outils visaient à décharger l'administrateur et à l'éloigner de ces deux tâches. [13] montre qu'une conception physique élaborée sans l'intervention de l'administrateur pose un problème de robustesse. Les techniques d'optimisation générées peuvent détériorer les performances au lieu de les améliorer. Les algorithmes utilisés par ces outils pour la sélection des techniques d'optimisation sont figés et non accessibles pour l'administrateur. Il est intéressant d'enrichir cette panoplie d'outils par d'autres outils d'aide permettant plus d'interactivité avec l'administrateur. Ces outils doivent donner la possibilité à l'administrateur de personnaliser sa conception physique et d'utiliser son expérience afin d'améliorer la qualité des techniques d'optimisation sélectionnées.

L'outil *OptAssist* que nous proposons permet d'aider l'administrateur dans sa tâche d'optimisation d'entrepôts de données. Il lui permet de choisir les techniques d'optimisation, le mode de leur sélection, les algorithmes utilisés, les paramètres relatifs à chaque algorithme ainsi que les tables et les attributs pris en compte pour la génération des recommandations. *OptAssist* permet de recommander une fragmentation primaire et dérivée[4], au contraire de la plupart des outils qui proposent uniquement la fragmentation primaire. Il permet aussi une sélection multiple de la FH et des index de jointure binaires (IJB) pour mieux optimiser l'entrepôt. L'outil utilise un modèle de coût que nous avons proposé

4. La FH primaire consiste à fragmenter une table en utilisant les attributs de cette table. La FH dérivée consiste à fragmenter une table selon les fragments d'une autre table

dans [14]. Il supporte plusieurs SGBD grâce à l'exploitation de la méta-base pour collecter toutes les informations et statistiques nécessaires pour l'optimisation.

3 Architecture de l'outil

OptAssist accepte comme entrée un schéma d'entrepôt, une charge de requêtes Q et un ensemble de contraintes (le nombre de fragments maximum W et le quota d'espace réservé aux index S), il permet en sortie de fragmenter horizontalement l'entrepôt, de l'indexer ou les deux en même temps. Le choix d'utiliser la FH et les IJB est motivée par plusieurs similarités que nous avons identifiées entre ces deux techniques [14]. L'outil est composé d'un ensemble de modules permettant d'aider l'AED à effectuer ses différents choix d'optimisation de l'entrepôt (voir figure 2) : (1) module d'interrogation de la méta-base, (2) module de gestion des requêtes, (3) module de sélection d'un schéma de FH, (4) module de sélection de configuration d'IJB, (5) module de FH, (6) module d'indexation et (7) module de réécriture des requêtes.

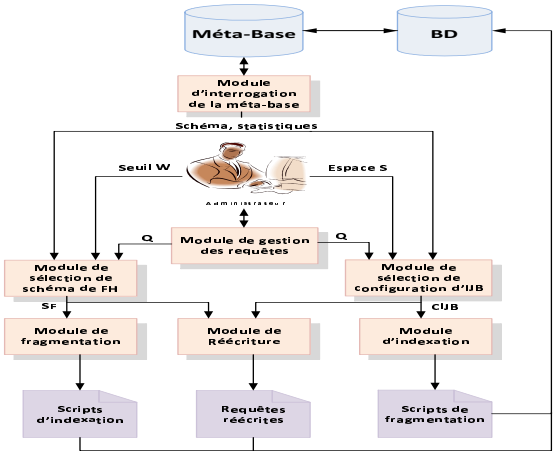


Figure 2. Architecture de l'outil

3.1 Module d'interrogation de la méta-base

Le module d'interrogation de la méta-base est un module très important qui permet à l'outil de fonctionner avec n'importe quel type de SGBD. A partir d'un type de SGBD, nom d'utilisateur et un mot de passe ce module permet de se connecter au compte correspondant et collecter un certain nombre d'informations à partir de la méta-base. Ces informations touchent deux niveaux, logique

et physique. Les informations du niveau logique regroupent la liste des tables du schéma logique que l'utilisateur gère ainsi que les attributs appartenant à ces tables. Les informations du niveau physique représentent les techniques d'optimisation utilisées ainsi qu'un ensemble de statistiques sur les tables et attributs de l'entrepôt.

3.2 Module de gestion des requêtes

Ce module permet d'aider l'AED à définir la charge de requêtes les plus fréquentes (Q) sur laquelle l'optimisation est basée. Le module permet une édition manuelle des requêtes ou une importation à partir de fichiers externes. Il permet aussi de gérer la charge en donnant la possibilité d'ajouter, de supprimer ou de modifier des requêtes. Le module intègre un *parseur* qui permet d'identifier les erreurs syntaxiques ainsi que les tables et attributs utilisés par chaque requête.

3.3 Module de sélection d'un schéma de fragmentation horizontale (MSSFH)

Le MSSFH nécessite en entrée un schéma de l'entrepôt, une charge de requêtes et un seuil W représentant le nombre de fragments maximum que l'administrateur souhaite avoir. A partir de ces données, ce module sélectionne un schéma de fragmentation (SF) permettant de minimiser le coût d'exécution des requêtes et générant un nombre de fragments ne dépassant pas W. Dans [15], nous avons effectué une étude de complexité du problème de FH dans le cadre des entrepôts de données relationnels et nous avons prouvé qu'il est NP-Complet. Pour cela, nous avons proposé trois algorithmes heuristiques pour le résoudre, un Algorithme Génétique (AG), un algorithme de Recuit Simulé (RS) et un algorithme de Hill Climbing (HC) [1,15]. Ces trois algorithmes sont supportés dans le MSSFH.

3.4 Module de sélection de configuration d'IJB

Ce module nécessite en entrée un schéma de l'entrepôt, une charge de requêtes (Q) et un espace de stockage S que l'administrateur réserve pour les index. Il sélectionne une configuration d'IJB ($CIJB$) permettant de minimiser le temps d'exécution des requêtes en entrée et occupant un espace de stockage ne dépassant pas S. Le module supporte deux algorithmes gloutons que nous avons proposé dans [14] et un algorithme basé sur une technique de data-mining (*Recherche des motifs fréquents fermés*) proposé par Aouiche et al. [6].

3.5 Module de fragmentation horizontale (MFH)

Le MFH est responsable de fragmenter physiquement l'entrepôt de données selon le schéma de FH obtenu à partir de module de sélection. A partir du schéma de FH, ce module détermine les tables de dimension à fragmenter par

la fragmentation primaire ainsi que les attributs utilisés pour effectuer cette fragmentation. Le module permet ensuite de fragmenter physiquement la table des faits par une fragmentation dérivée en utilisant les fragments des tables de dimension. Dans [14] nous avons identifié deux problèmes : (1) la plupart des SGBD ne supportent pas la FH primaire sur trois attributs ou plus et (2) la FH dérivée n'est pas supportée en cas de deux tables de dimension ou plus. Nous avons proposé une technique permettant de résoudre ces deux problèmes. Cette technique est supportée par le MFH. Ce dernier génère tous les scripts qui permettent de fragmenter les tables de dimension ainsi que la table de faits selon le schéma de fragmentation SF en entrée.

3.6 Module d'indexation

Le module d'indexation est responsable de la création des IJB sélectionnés par le module de sélection des index. Ce module génère les requêtes de création des index sur l'entrepôt de données.

3.7 Module de réécriture des requêtes

Une fois les structures d'optimisation créées physiquement sur l'entrepôt (FH et/ou IJB), une étape de réécriture des requêtes est nécessaire. La réécriture pour les IJB consiste à ajouter un $Hint$ dans la clause SELECT pour forcer l'utilisation des index Créés[5]. La réécriture pour la FH consiste à identifier les fragments valides pour chaque requête, de réécrire la requête sur chacun de ces fragments et enfin faire l'union des résultats obtenus.

4 Fonctionnalités de l'outil

Nous présentons dans cette section les principales fonctionnalités de l'outil à travers son utilisation sur un entrepôt de données réel issu du Benchmark Apb-1 [16]. Le schéma en étoile que nous avons dégagé à partir de ce banc d'essais est constitué d'une table de faits *Actvars* (24 786 000 n-uplets) et de quatre tables de dimension, *Prodlevel* (9 000 n-uplets), *Custlevel* (900 n-uplets), *Timelevel* (24 n-uplets) et *Chanlevel* (9 n-uplets).

Pour aider l'AED dans l'optimisation de l'entrepôt de données, l'outil assure quatre principales fonctionnalités : visualisation de l'état de l'entrepôt, la préparation de l'optimisation, la fragmentation de l'entrepôt et l'indexation de l'entrepôt (fragmenté ou non).

4.1 Visualisation de l'état de l'entrepôt

La visualisation de l'état de l'entrepôt permet à l'AED de connaître le schéma de l'entrepôt de données, les tables de dimension, la table des faits ainsi que

5. Le hint *INDEX* dans une requête permet de forcer l'utilisation d'un ou plusieurs index dans le plan d'exécution d'une requête.

certaines statistiques sur ces tables. La visualisation permet aussi d'afficher les techniques d'optimisation déjà créées sur l'entrepôt de données. Toutes ces informations sont collectées grâce au module de d'interrogation de la méta-base. Cette visualisation permet à l'administrateur d'avoir une vue globale sur son entrepôt avant de commencer un processus d'optimisation. La figure 3(a) montre un exemple de visualisation où les tables, les attributs ainsi que les techniques d'optimisation créées sont affichés. La figure 3(b) montre un ensemble de statistiques collectées sur les objets de l'entrepôt.

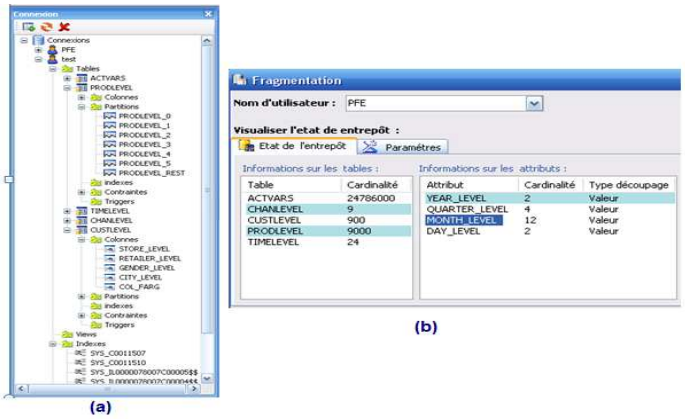


Figure 3. (a) Visualisation des objets de l'entrepôt, (b) visualisation des statistiques

4.2 Préparation de l'optimisation

La préparation de l'optimisation vise à collecter les informations nécessaires pour effectuer cette optimisation. Elle concerne la préparation de la charge de requêtes Q utilisée pour effectuer l'optimisation, le choix du mode de sélection et la définition des paramètres physique. La figure 4 montre l'interface permettant de gérer la charge des requêtes. L'outil permet aussi d'ajouter, modifier et supprimer une requête et de vérifier sa syntaxe. L'outil supporte deux modes de sélection : *isolé* et *multiple*. Dans le mode isolé l'AED peut utiliser la fragmentation seule (FHSEULE) ou l'indexation seule (IJBSEULS) pour optimiser son entrepôt. La sélection multiple consiste à utiliser les deux techniques en fragmentant l'entrepôt en un ensemble de fragments ensuite indexer ces derniers. L'outil permet à l'AED de fixer certains paramètres physiques comme la taille du buffer et la taille de la page système.

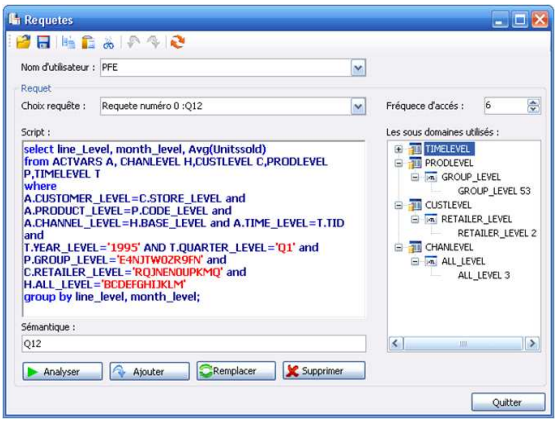


Figure 4. Interface de gestion de la charge de requêtes

4.3 Le partitionnement de l'entrepôt

Le partitionnement de l'entrepôt consiste à fragmenter les tables de dimension par la FH primaire ensuite la table des faits par la FH dérivée. L'AED commence par choisir le nombre de fragments maximum (W) ensuite il choisit s'il veut une fragmentation personnalisée ou non. S'il choisit la fragmentation non personnalisée alors *OptAssist* partitionne l'entrepôt de données en utilisant tous les attributs candidats et un algorithme de fragmentation par défaut. La fragmentation personnalisée, *offre plus de liberté* à l'AED dans le processus de sélection. Il peut choisir les tables et les attributs de dimension participant au processus de fragmentation. Il doit choisir l'algorithme de partitionnement (AG, RS ou HC) et établir ses paramètres. La figure 5 représente la zone de choix des algorithmes et de leurs paramètres. Pour chaque algorithme sélectionné, *OptAssist* active les paramètres correspondants et donne la possibilité à l'AED de les modifier. Pour illustrer la fragmentation personnalisée nous considérons le cas où l'AED choisit d'éliminer certains attributs et tables du processus de fragmentation. La figure 7 représente l'interface de personnalisation de la fragmentation. Si l'AED choisit de personnaliser sa fragmentation, alors *OptAssist* lui donne la possibilité de choisir les attributs et les tables de dimension candidats pour le processus de fragmentation. Dans cette figure, l'AED a éliminé la table *CustLevel*, un attribut de la table *TimeLevel* et trois attributs de la table *ProdLevel* du processus de fragmentation. Après avoir choisi les tables et les attributs, l'AED choisit l'algorithme de recuit simulé, fixe W à 100 et lance l'exécution. Le schéma de fragmentation obtenu avec cette personnalisation génère 72 fragments et un coût d'exécution de requêtes représentant un gain de l'ordre de 8,8% par rapport à la fragmentation non personnalisée.

Une fois le schéma de l'entrepôt fragmenté, l'AED peut visualiser une recommandation proposée par *OptAssist*. Elle contient le nombre de fragments générés, les attributs utilisés par la FH, les tables de dimension fragmentées, une estimation du nombre d'entrées-sorties nécessaires pour exécuter la charge de requêtes, le nombre de fragments de chaque table de dimension, le gain de performance obtenu par cette fragmentation (par rapport à un schéma non fragmenté), etc. La figure 8 montre l'interface affichant les attributs utilisés pour fragmenter l'entrepôt (quatre attributs parmi douze ont été utilisés : *Line_level, Year_level, Month_level* et *All_level*). Si l'AED n'est pas satisfait de cette recommandation, il peut revenir à l'étape précédente et changer les différents paramètres (rechoisir les attributs et les tables ou l'algorithme, ses paramètres, etc.). Ce retour en arrière est primordial dans la phase de conception physique. Une fois satisfait, l'AED demande à l'outil de générer les scripts de fragmentation. Ceux-ci pourront par la suite être appliqués sur l'entrepôt de données.

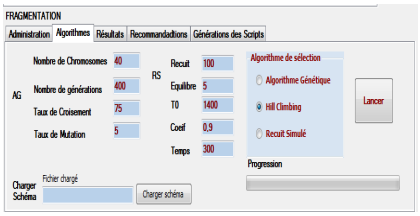


Figure 5. Choix et configuration des algorithmes

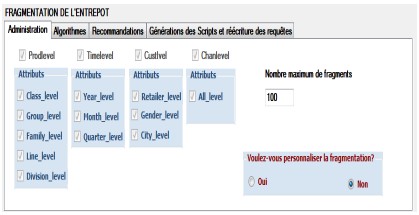


Figure 6. Fragmentation non personnalisée

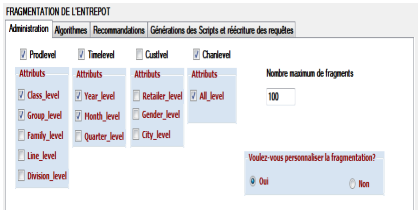


Figure 7. Personnalisation de la fragmentation

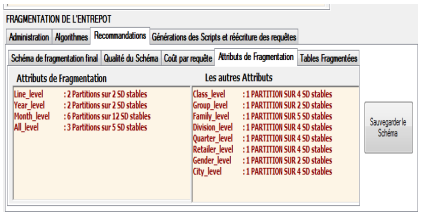


Figure 8. Attributs de fragmentation

Dans le cas où l'AED est satisfait du schéma sélectionné, il demande la génération des scripts et la réécriture des requêtes sur ce schéma. Pour fragmenter physiquement l'entrepôt, l'AED exécute les scripts générés sur l'entrepôt d'origine, ce dernier sera remplacé par l'entrepôt fragmenté.

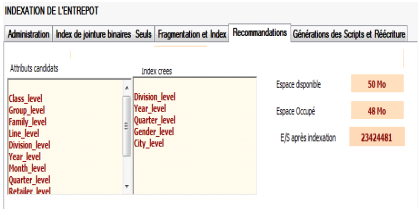


Figure 9. Recommandations d'indexation dans IJBSEULS

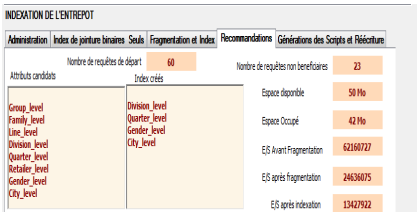


Figure 10. Recommandation d'indexation dans HP&IJB

4.4 L'indexation de l'entrepôt

La tâche d'indexation permet de sélectionner un ensemble d'IJB sur l'entrepôt de données des IJB. L'outil supporte deux manières d'indexation : isolée (IJBSEULS) et multiple (FH&IJB). Dans le cas d'une indexation de type IJBSEULS, l'AED doit d'abord choisir les attributs indexables candidats et l'espace de stockage S. Comme pour la FH, deux types d'indexation sont possibles : *l'indexation non personnalisée* et *l'indexation personnalisée*. Les recommandations générées contiennent des informations concernant le gain apporté par les index, les attributs indexés, le coût de stockage exigé par les index sélectionnés, etc. Pour illustrer ce cas, nous considérons que l'AED choisit de faire une indexation non personnalisée avec un seuil de 50 Mo. La figure 9 présente l'interface dédiée à aux recommandations générées sur les index créés (les attributs indexés, l'espace occupé, l'espace disponible, etc.). Parmi les douze attributs indexables, cinq attributs ont été utilisés pour créer cinq IJB occupant 48 Mo d'espace disque.

L'indexation dans FH&IJB consiste à indexer l'entrepôt après son fragmentation. La différence entre l'indexation dans IJBSEULS et FH&IJB réside dans le choix des attributs candidats. Au lieu de considérer les attributs candidats à partir de la configuration initiale de l'entrepôt de données, l'AED doit identifier ces attributs parmi les attributs d'indexation non utilisés par la fragmentation. Pour illustrer cette indexation, considérons que l'AED cherche à indexer l'entrepôt de données fragmenté. Il choisit le mode de sélection multiple, l'outil désactive automatiquement les attributs utilisés pour fragmenter l'entrepôt, puisqu'ils ne sont pas utilisés pour indexer l'entrepôt. L'AED choisit alors les attributs candidats à l'indexation et un seuil d'espace de stockage de 50 Mo, puis lance l'algorithme de sélection. La figure 10 montre les recommandations d'indexation après la sélection d'une configuration d'index. Nous trouvons plusieurs informations, comme le nombre de requêtes non bénéficiaires de la fragmentation, les attributs indexés, l'espace de stockage des index sélectionnés, le coût d'exécution avant et après indexation, etc. De la même façon que pour la fragmentation, s'il est satisfait des recommandations, il lance la génération des scripts de création des index binaires, sinon il peut revenir en arrière pour modifier ses choix.

5 Conclusion

La tâche d'optimisation de la couche physique dans les entrepôts de données est devenue un enjeu majeur. Cela est dû aux caractéristiques des entrepôts : la volumétrie, la complexité des requêtes, les exigences de temps de réponse raisonnable et la gestion de l'évolution de l'entrepôt. Dans cet environnent, nous avons mis en évidence les difficultés qu'un administrateur pourrait rencontrer durant l'optimisation de la couche physique. Ces difficultés sont multiples, car elles concernent plusieurs niveaux de conception : le choix des techniques d'optimisation pertinentes pour l'ensemble de requêtes à optimiser, le choix de la nature de sélection des techniques d'optimisation et le choix des algorithmes et leur paramètres. Vu ces difficultés, nous avons identifié le besoin de développer un outil d'assistance de l'administrateur qui permet de répondre aux besoins en termes de choix possibles. Nous avons proposé l'outil, *OptAssist*, offrant trois techniques d'optimisation : la FH primaire, FH dérivée et les IJB. Il permet à l'administrateur de choisir les différents algorithmes et leurs paramètres. Il peut alors utiliser ces techniques d'une manière isolée ou multiple. Une autre particularité de *OptAssist* est le fait de proposer des sélections personnalisées et non personnalisées des structures d'optimisation. Une des perspectives de cet outil est son extension en considérant d'autres techniques d'optimisation comme les vues matérialisées, la fragmentation verticale, traitement parallèle.

Références

1. Bellatreche, L., Boukhalfa, K., Abdalla, H.I. : Saga : A combination of genetic and simulated annealing algorithms for physical data warehouse design. in 23rd British National Conference on Databases (212-219) (July 2006)
2. Bellatreche, L., Missaoui, R., Necir, H., Drias, H. : A data mining approach for selecting bitmap join indices. Journal of Computing Science and Engineering **2**(1) (January 2008) 206–223
3. Chaudhuri, S. : Index selection for databases : A hardness study and a principled heuristic solution. IEEE Transactions on Knowledge and Data Engineering **16**(11) (November 2004) 1313–1323
4. Johnson, T. : Performance measurements of compressed bitmap indices. Proceedings of the International Conference on Very Large Databases (1999)
5. Chee-Yong, C. : Indexing techniques in decision support systems. Phd. thesis, University of Wisconsin - Madison (1999)
6. Aouiche, K., Darmont, J., Boussaid, O., Bentayeb, F. : Automatic Selection of Bitmap Join Indexes in Data Warehouses. 7th International Conference on Data Warehousing and Knowledge Discovery (DAWAK 05) (August 2005)
7. Sanjay, A., Surajit, C., Narasayya, V.R. : Automated selection of materialized views and indexes in microsoft sql server. Proceedings of the International Conference on Very Large Databases (September 2000) 496–505
8. Talebi, Z.A., Chirkova, R., Fathi, Y., Stallmann, M. : Exact and inexact methods for selecting views and indexes for olap performance improvement. 11th International Conference on Extending Database Technology (EDBT'08) (Mars 2008)

9. Agrawal, S. : Automatic sql tuning in oracle 10g. In Proceedings of the 30th International Conference on Very Large Databases (VLDB) (2004)
10. Zilio, D.C., Rao, J., Lightstone, S., Lohman, G.M., Storm, A., Garcia-Arellano, C., Fadden, S. : Db2 design advisor : Integrated automatic physical database design. Proceedings of the International Conference on Very Large Databases (August 2004) 1087–1097
11. Agrawal, S. : Database tuning advisor for microsoft sql server 2005. In Proceedings of the 30th International Conference on Very Large Databases (VLDB) (2004)
12. Valentin, G., Zuliani, M., Zilio, D., Lohman, G., Skelley, A. : Db2 advisor : An optimizer smart enough to recommend its own indexes. In : 16th International Conference on Data Engineering (ICDE 00), San Diego, USA. (2000) 101–110
13. Gebaly, K.E., Aboulnaga, A. : Robustness in automatic physical database design. in 11th International Conference on Extending Database Technology (EDBT'08), March (2008)
14. Boukhalfa, K. : De la conception physique aux outils d'administration et de tuning des entrepôts de données. Ph.d. thesis, Ecole Nationale Supérieure de Mécanique et d'aéronautique Poitiers et Université de Poitiers (July 2009)
15. Bellatreche, L., Boukhalfa, K., Richard, P. : Horizontal partitioning in data warehouse : Hardness study, selection algorithms and validation on oracle10g. in 10th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2008) (September 2008) 87–96
16. Council, O. : Apb-1 olap benchmark, release ii. http ://www.olapcouncil.org/ research/ resrchly.htm (1998)

Evaluation de Requêtes Flexibles dans un Contexte Non-Centralisé : Une Approche Basée sur les Résumés Distribués

Abdelkader Alem[1], Allel Hadjali[2]

[1] Département d'Informatique, Université Ibn Khaldoun BP 78, 14000 Tiaret, Algérie
alemainhadid@yahoo.fr
[2] ENSSAT/IRISA, Université Rennes 1, 6 rue de kérampont - BP 80518,
22305 Lannion Cedex, France
hadjali@enssat.fr

Résumé. L'étude présentée dans cet article traite le problème d'évaluation des requêtes flexibles dans un contexte distribué, illustré par un système P2P. L'idée suggérée est de n'envoyer la requête qu'aux sources de données qui sont susceptibles de fournir les meilleures réponses à la requête, ce qui nécessite d'évaluer la pertinence de chaque source au moyen de son résumé. L'approche proposée consiste en deux étapes. La première étape concerne la construction d'un index de routage global (et distribué), qui décrit les données des différentes sources, afin de trouver l'ensemble des pairs pertinents pour la requête posée. La seconde étape a trait à la recherche des réponses satisfaisant au mieux cette requête.

Mots-clés: Requête flexible, système P2P, index de routage, résumé de donnée.

1 Introduction

Au cours de cette dernière décennie, le paradigme des systèmes distribués, en particulier les systèmes dits P2P, est devenu très populaire en permettant le partage des ressources et l'échange d'information entre des millions d'utilisateurs. De plus, les P2P offrent de nombreux autres avantages comme l'auto-organisation, l'autonomie et le "passage à l'échelle". Parmi les systèmes P2P les plus connus, on peut citer Gnutella [10], Napster [11]. Cependant, un des problèmes majeurs dont souffrent ces systèmes est la localisation des sources (ou pairs) pertinentes (c.-à-d., celles contenant les réponses qui satisfont au mieux les besoins de l'utilisateur). Ce problème a fait l'objet de plusieurs études et de nombreuses techniques efficaces ont été proposées pour la localisation de données pertinentes dans les systèmes P2P [6].

Les index de routage [6] font partie des techniques proposées dans la littérature pour une évaluation efficace (en évitant la stratégie d'inondation) des requêtes dans les réseaux P2P. Rappelons qu'un index de routage est une structure de données (avec un ensemble d'algorithmes) qui, étant donnée une requête sur un pair, retourne la liste de pairs voisins ordonnés selon leur pertinence à la requête considérée. Ces index

permettent donc de ne renvoyer la requête qu'aux pairs qui sont les plus probables à fournir de réponses. On distingue deux catégories d'index : les index locaux et les index globaux. Dans Gnutella [10], chaque nœud maintient un index local sur les données qu'il possède (un nœud diffuse la requête à tous ses voisins dans le réseau). Un index global peut être centralisé ou distribué, par exemple, Napster [11] stocke un index global de toutes les données sur un pair central (ou super pair), alors que Chord [16] distribue un index global sur tous les nœuds du réseau.

Par ailleurs, les requêtes à préférences est un thème de recherche qui a aussi suscité un intérêt croissant ces dernières années, voir par exemple [2][5][7]. Les requêtes dites Skyline [2] ont reçu une attention particulière de la part d'un grand nombre de chercheurs dans la communauté des bases de données. Cependant, la plupart des travaux proposés ont été réalisés dans un environnement centralisé. Relativement, peu d'études existent dans un cadre décentralisé [9][12][13][18][19]. Mentionnons, par exemple, les travaux de Hose et al. [9] et de Zinn [19] qui ont proposé des approches pour le traitement des requêtes skyline dans des systèmes P2P. Dans ces travaux, les auteurs utilisent des index de routage appelés DDS (Distributed Data Summaries). Ces index sont basés sur une structure d'arbre particulière, appelée QArbre (traduction de l'anglais de QTree). Cette structure est une combinaison d'histogramme et d'une autre structure d'arbre (dite R-Tree). Dans chaque pair p, on maintient un résumé de toutes les données qui peuvent être accessibles par envoi d'une requête à tous les voisins de p.

Dans cet article, le problème considéré concerne l'évaluation des requêtes flexibles [3] dans un système distribué. Ces requêtes permettent aussi d'exprimer des préférences au moyen de prédicats flous (comme "jeune", "cher", etc.) dont la satisfaction est graduelle. Dans ce cadre, le résultat d’une requête n’est plus un ensemble "plat" d’éléments mais un ensemble où chaque élément est associé avec un degré de satisfaction.

Pour autant que nous le sachions, il n'existe pas de travaux sur cette problématique dans la littérature, excepté ceux proposés dans [4][8] où l'aspect évaluation des requêtes n'a pas été suffisamment traité. L'étude que nous suggérons combine l'approche décrite dans [19] pour la construction d'index de routage et le modèle de résumé introduit dans [15] pour la définition du contenu de l'index. En particulier, une stratégie d'évaluation de requêtes flexibles dans un environnement P2P est proposée. Elle permet de ne renvoyer que les réponses qui satisfont au mieux la requête posée.

Le reste de l'article est organisé comme suit. La section 2 introduit un exemple de référence servant à illustrer notre approche. La section 3 présente l'approche basée sur les QArbres pour la construction d'index de routage. En section 4, on décrit le modèle de résumé SaintEtiq. La section 5 est consacrée à l'approche proposée pour la construction d'index global et distribué. La section 6 discute l'évaluation de requêtes flexibles dans un contexte P2P. Un exemple illustratif est décrit dans la section 7. La section 8 conclut l'article et esquisse quelques directions pour de futurs travaux.

2 Exemple de Référence

L'exemple suivant est utilisé pour illustrer notre approche. Il s'agit de 3 magasins de vente (de cameras) répartis sur trois sites formant un réseau P2P (avec les sites 1 et 3 comme nœuds feuilles et le site 2 comme nœud central). Chaque site maintient sa propre base de données. Chaque tuple (caméra) est décrit par deux attributs *(prix, qualité)*, voir Tableau 1.

Tableau 1. Ensemble de caméras réparties sur trois sites.

Site 1			**Site 2**			**Site 3**		
Modèle	Prix	Qualité	Modèle	Prix	Qualité	Modèle	Prix	Qualité
S1	165	7,2	C1	90	5	X1	295	9,3
S2	275	8,1	C2	260	6,5	X2	310	12,8
S3	270	8,2	C3	375	7,9	X3	540	14,1
S4	369	13,6	C4	55	4,6	X4	330	8,1
S5	635	10,1	C5	410	14	X5	720	16,1
S6	350	10,1	C6	560	15,3	X6	100	7,9
S7	290	9,5	C7	521	15,1	X7	160	7,1
S8	395	8,3	C8	730	16,5	X8	360	10,3
S9	412	14	C9	820	17	X9	290	8
S10	537	14,7	C10	610	16,5	X10	270	7,8
S11	300	12,3				X11	340	8,9
S12	149	5,5						

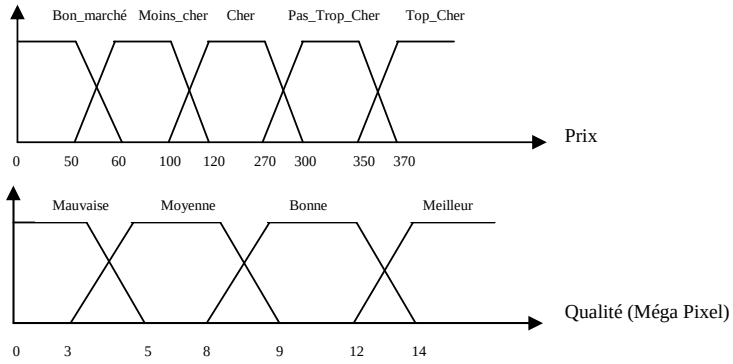


Fig. 1. Variables linguistiques décrivant les attributs "*Prix*" et "*Qualité*"

Dans le Tableau 1 la colonne "Modèle" est utilisée juste pour l'identification des tuples (le prix est exprimé en euros et la qualité concerne la résolution d'une caméra, elle est mesurée en termes de nombre de pixels). On suppose aussi disponible des partitions floues sur les attributs *"Prix"* et *"Qualité"*, voir Figure 1. Par exemple, les

prédicats flous "*Pas_Trop_Cher*" et "*Bonne*" sont représentés par les f.a.t[1] (300, 350, 30, 20) et (9, 12, 1, 2) respectivement. Voir [3] pour plus de détails.

3 Index de Routage : L'Approche de Zinn

Le problème traité par Zinn dans [19] concerne l'évaluation des requêtes Skyline dans un environnement P2P. La solution proposée comprend deux phases essentielles : (i) Construction d'un index de routage basé sur une structure hiérarchique dite QArbre; (ii) Evaluation des requêtes en exploitant l'index de routage construit. Dans ce qui suite, on décrit brièvement la première étape en présentant tout d'abord la structure de QArbre.

3.1 Structure de QArbre

Une structure de QArbre [19] est un arbre où chaque nœud correspond à une boîte multidimensionnelle rectangulaire (appelée MBB pour Minimum Bounding Box). Un MBB est le plus petit rectangle contenant de l'information sur les données se trouvant dans le sous-arbre (resp. nœud) si le nœud n'est pas une feuille mais un nœud interne (resp. si le nœud est une feuille). Cette information est de nature statistique comme, par exemple, le nombre de données accessibles via le nœud. Les feuilles sont appelées des "paniers" (buckets). Plus la taille d'un panier est réduite, plus il est plus facile de décider si ses données sont pertinentes pour répondre à une requête posée ou non.

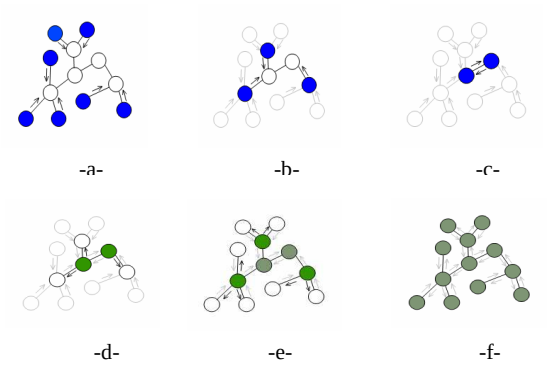


Fig. 2. Processus de construction de l'index

3.2 Principe de construction de l'index

Le QArbre est utilisé comme structure de base pour la compression des données du réseau. Le réseau est constitué d'un ensemble de pairs où chaque pair est relié à un petit ensemble de voisins et peut envoyer des messages à tous ses voisins. Un

[1] f.a.t signifie fonction d'appartenance trapézoïdale. Elle est représentée par un quadruplet (A, B, a, b) où [A, B] (resp. [A-a, B+b]) exprime le noyau (resp. le support).

algorithme distribué pour la construction de l'index est proposé dans [19]. La Figure 2 illustre les différentes étapes de cette construction dans un réseau P2P (avec 13 pairs). La Figure (Fig. 2-a) montre la phase d'initialisation : tous les nœuds feuilles (colorés en bleu) envoient de l'information sur leurs données locales (résumées au moyen de "buckets"). Puis les pairs ayant reçu des messages de tous leurs voisins sauf un (appelé N) résument leurs données locales avec toutes les données de leurs voisins (sauf évidemment N) dans des structures QArbres pour les envoyer ensuite au pair N (Fig. 2-b). Cette procédure est exécutée itérativement jusqu'à ce que les nœuds centraux reçoivent l'information de tous leurs voisins (Fig. 2-c). La dernière étape de l'algorithme consiste à faire l'opération inverse (l'information globale est diffusée dans le même chemin mais de haut en bas). Quand le dernier nœud feuille reçoit cette information de routage, l'algorithme se termine et ainsi tous les pairs possèdent des index de routage dits corrects (Fig. 2-d, e et f).

4 Résumé de Données : Le modèle SaintEtiQ

4.1 Construction du Résumé

SaintEtiq [15] est un modèle structuré pour les résumés de données. Il prend en entrée deux types d'informations : les données à résumer et les données relatives au domaine (appelées aussi connaissances sur le domaine). Ces connaissances sont constituées essentiellement de variables linguistiques définies sur les domaines d'attribut de la relation considérée (voir Figure 1).

Un résumé *z* peut être représenté par une paire *(Iz, Rz)* où *Iz* (resp. *Rz*) est appelé l'intension (resp. extension) de z. L'extension d'un résumé *z* est le sous ensemble des tuples impliqués dans *z*, alors que l'intension est la description linguistique de ces tuples. Par exemple, dans les Tableaux 2 et 3, l'intension du résumé Z_{f1} est I_{Zf1}= *{Cher, Moyenne}* et son extension R_{Zf1}= $\{Ct_1, Ct_{2,1}, Ct_{3,1}, Ct_{12,2}\}$.

Le modèle SaintEtiQ comprend deux grandes phases : i) la réécriture; et ii) l'organisation du résumé.

Tableau 2. Réécritures des Tuples du site 1 (Ch : Cher, Moy : Moyen, P_T_C : Pas très Cher)

Mod	Prix	Qualité	Tuples candidats
S1	16500	7,2	Ct1 (1.0/Ch, 1.0/Moy)
S2	27500	8,1	Ct2,1 (0.7/Ch, 1.0/Moy);
			Ct2,2 (0.3/P_T_C, 1.0/Moy)
S3	27000	8,2	Ct3,1 (1.0/Ch, 0.8/Moy);
			Ct3,2 (1.0/P_T_C, 0.2/Bon)
S4	36900	13,6	Ct4,1 (0.1/P_T_C, 1.0/Meil);
			Ct4,2 (0.9/T_C, 1.0/Meil)
…	…	…	…

4.1.1 Etape de réécriture

Cette étape permet au système de réécrire les tuples de la base de données avant que ceux-ci ne soient exploités par le service du résumé (un tuple réécrit est appelé tuple candidat, *Ct*). Elle donne naissance à un ou plusieurs tuples candidats qui peuvent être considérés comme des représentations linguistiques d'un tuple de la base.

Par exemple dans le Tableau 2, le tuple S4 est réécrit en deux tuples candidats Ct4,1 (0.1/P_T_Ch, 1.0/Meil) et Ct4,2 (0.9/T_Ch, 1.0/Meil). Le Tableau 2[2] donne quelques réécritures des tuples du site 1 des caméras.

4.1.2 Organisation du résumé

Cette étape consiste à organiser les résumés au sein d'une hiérarchie de manière à ce que le résumé le plus général soit placé au sommet de la hiérarchie et les résumés les plus spécifiques au niveau des feuilles. Le résumé racine décrit ainsi l'intégralité du jeu de données tandis que les feuilles ne résument qu'une partie plus limitée de la base. Voir Tableau 3 et Figure 3 (pour la hiérarchie de résumé associée au site 1).

Tableau 3. Classification des tuples candidats pour le site 1

Résumé Z	Intension	Tuples couverts par Z
Z_{f1}	(1.0/Ch, 1.0/Moy)	Ct_1, $Ct_{2,1}$, $Ct_{3,1}$, $Ct_{12,2}$
Z_{f2}	(1.0/P_T_C, 1.0/Moy)	$Ct_{2,2}$, $Ct_{11,1}$
Z_{f3}	(1.0/Ch, 1.0/Bon)	$Ct_{3,2}$, $Ct_{7,1}$
…	…	…
Z_{f7}	(1.0/P_T_C, 1.0/Bon)	Ct_6, $Ct_{7,2}$, $Ct_{11,2}$
…	…	…
Z_{13}	(1.0/P_T_C, 1.0/T_C, 1.0/Meil)	Z_{f4}, Z_{f5}
Z'_1	(1.0/Ch, 1.0/P_T_C, 1.0/T_C, 0,1/Mauv , 1.0/Moy, 1.0/Bon,)	Z_{11} , Z_{12} , Z_{f9}
Z_1	(1.0/Ch, 1.0/P_T_C, 1.0/T_C, 0,1/Mauv, 1.0/Moy, 1.0/Bon, 1.0/Meil)	Z_{13} , Z'_1

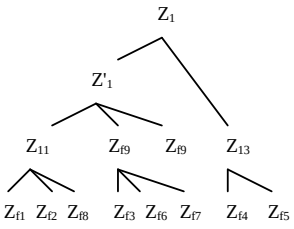


Fig. 3. Hiérarchie de résumé associée au site 1

4.2 Principe d'interrogation

L'interrogation d'une hiérarchie de résumé SaintEtiq a été abordée dans [17]. L'idée de base consiste à parcourir l'arbre des résumés afin de trouver les réponses qui satisfont la requête posée. Le résultat est donc un ensemble de résumés. Dans cette section, on rappelle les éléments essentiels de cette démarche.

Reprenons l'exemple des caméras (voir Section 2) et soit la requête flexible Q = *"Pas_Trop_Cher* $\wedge$ *Bonne"*. L'ensemble des étiquettes linguistiques apparaissant dans une requête sur l'attribut A_i sont dénotées par C_{Ai}. Les différents C_{Ai} sont regroupés dans l'ensemble C appelé *caractérisation initiale* liée à la requête posée. Par exemple,

[2] Faute de place, nous ne pouvons donner ici tout le contenu du Tableau 2 (resp. 3). Ces deux tableaux sont complètement décrits dans [1].

C = {Pas_Trop_Cher, Bonne} pour la requête *Q*. La sélection des résumés est réalisée par la fonction *Recherche* de l'algorithme 1. Cette fonction est récursive, elle prend en entrée un résumé *z* et la caractérisation initiale *C* de la requête. Elle retourne en sortie une liste de résumés L_{res}.

Dans l'algorithme 1 le test de correspondance, Corr(z, C), entre les caractères requis C_{Ai} des attributs de la requête et les descripteurs $L_{Ai}(z)$ extraits de l'intension de z, peut être de trois types :

Correspondance nulle : il existe au moins un attribut A_i pour lequel *z* ne montre aucun des caractères requis pour A_i.

Correspondance exacte : pour tous les attributs de la requête, le résumé *z* présente uniquement les caractéristiques recherchées, il est considéré comme résultat si *z* est une feuille.

Correspondance par excès : cette situation se présente lorsque le résumé *z* possède (sur un ou plusieurs attributs) plus de descripteurs que ceux de la requête, l'exploration du sous-arbre de racine *z* est donc nécessaire.

Algorithme 1. Fonction Recherche(z, C) [17]

```
Entrée : Résumé z et la caractérisation C
1. Lres := ϕ /* la liste est vide */
2. Si Corr(z, C) = excès Alors
3.  Pour chaque nœud fils zfils de z Faire
4.        Lres := Lres ∪ Recherche(zfils, C)
5.  Fin pour
6. Sinon
7.   Si Corr(z, C) = exacte Alors
8.        Si z est une feuille Alors
9.              ajouter (z, Lres)
10.       Sinon
11.             Lres := Lres ∪ Recherche (z, C)
12.       Fin si
13.  Fin si
14. Fin si
Résultat : Lres
```

5 Gestion des Résumés dans un Système P2P

Cette section est consacrée à la démarche proposée pour la construction d'un index global permettant le routage dans le réseau P2P. Il a été montré dans [17] qu'un résumé SaintEtiQ peut être considéré comme un index multidimensionnel. Dans cette étude, l'idée suggérée ici est de construire une hiérarchie de résumé (résumé global) qui décrit toutes les données du réseau P2P. L'approche proposée combine (i) l'algorithme décrit dans [19] pour la construction d'index de routage (cet algorithme présente l'avantage, d'une part, d'avoir une complexité temporelle acceptable et, d'autre part, de garantir une maintenance à moindre coût) ; et (ii) le modèle de résumé

de données SaintEtiq pour la définition du contenu de l'index (ce modèle conduit à un résumé exprimé en termes linguistiques et donc proche du langage de l'utilisateur).

Dans cette étude, le modèle P2P considéré est un système pair à pair décentralisé non structuré où la recherche se fait à l'aide des index de routage. Les hypothèses de base suivantes sont supposées vérifier :

- Les données sont arbitrairement distribuées sur les pairs.
- Chaque pair stocke et partage sa base de données locale.
- Tous les attributs impliqués dans la requête figurent dans les différentes bases de données distribuées.
- Chaque pair maintient un résumé local de sa propre base. Tous les pairs coopèrent pour construire un résumé global décrivant l'ensemble des données du réseau. On suppose aussi que les mêmes connaissances du domaine sont utilisées pour résumer les données de toutes les sources.
- Le problème d'hétérogénéité (et l'intégration de schémas) n'est pas considéré ici. Un schéma global est supposé.

5.1 Enrichissement des résumés

On considère ici qu'une hiérarchie de résumé sert comme un index multidimensionnel où un terme supplémentaire, P_z, est ajouté à la définition d'un résumé. Ce terme, dit Peer-extent [8], fournit l'ensemble des pairs qui ont des données décrites par le résumé *z*. On suppose aussi que dans la phase de réécriture, chaque tuple candidat contient un identificateur de son tuple original.

5.2 Procédure de construction de l'index

Cette procédure est largement inspirée de [19], la seule différence réside dans la manière de résumer les données. Dans notre cas, les données locales pour chaque pair sont résumées à l'aide du modèle SaintEtiQ. Ainsi, l'index global décrivant l'ensemble des données du réseau P2P est construit en suivant les étapes suivantes:

1. Une phase d'initialisation est faite sur chaque pair. L'ensemble des données locales est résumé à l'aide du modèle SaintEtiQ et un index local est créé pour chaque pair.
2. Tous les nœuds feuilles envoient une copie des informations sur leurs données (synthétisées sous forme d'une hiérarchie de résumés) vers ses voisins dans le réseau de P2P (Fig.2-a).
3. Tous les pairs ayant reçu des messages depuis leurs voisins sauf un (appelé N), fusionnent leurs informations locales (disponibles sous forme de résumé) avec toutes les informations reçues des différents voisins à l'exception du nœud N. Puis, ils envoient le résumé fusionné à N (Fig.2-b). L'opération de fusion suit exactement le modèle SaintEtiQ comme expliqué en Section 4.1 et elle est illustrée plus loin en Section 7.

4. L'étape 3 est répétée itérativement jusqu'à ce que le nœud central reçoive des informations de tous leurs voisins (Fig.2-c).
5. Le nœud central produit un résumé global (après la fusion de tous les résumés de ses voisins avec son résumé local). Ce résumé décrit toutes les données disponibles dans le réseau P2P, il sert comme un index sur les données du réseau (Fig.2-d).
6. La dernière étape consiste à faire l'opération inverse (le résumé global est diffusé dans le même chemin mais de haut en bas), et l'algorithme se termine quand les nœuds feuilles reçoivent l'index de routage global (Fig.3- e et f).

6 Stratégie d'Evaluation

Dans cette section, on présente une stratégie efficace pour localiser les pairs pertinents à une requête posée. Cette localisation est réalisée en exploitant l'index global (représenté par la hiérarchie des résumés). La solution proposée consiste à adapter l'algorithme 1 dans le sens où la réponse retournée n'est plus un ensemble de résumés, mais un ensemble de pairs. Ensuite, un algorithme de recherche des meilleurs tuples satisfaisant la requête est proposé. Cette recherche se fait en interrogeant la base de données originale par le biais d'un index local.

6.1 Localisation des pairs

Lorsqu'une requête est posée sur un pair, l'index (hiérarchie globale) est exploré à l'aide de l'algorithme 2 (une version adaptée de l'algorithme 1) donné ci-dessous.

Algorithme 2. Fonction SelectPair (z, C)

```
Entrée : Résumé z et la caractérisation C
 1. P_Q := ϕ /* la liste est vide */
 2. Si Corr(z, C) = excès Alors
 3.  Pour chaque nœud fils z_fils de z Faire
 4.          P_Q := P_Q ∪ SelectPair (z_fils, C)
 5.  Fin pour
 6. Sinon
 7.    Si Corr(z, C) = exacte Alors
 8.       Si z est une feuille Alors
 9.            ajouter (P_z, P_Q)
 10.      Sinon
 11.           P_Q := P_Q ∪ SelectPair (z, C)
 12.      Fin si
 13.   Fin si
 14. Fin si
Résultat : P_Q  /*la liste des pairs pertinents pour Q */
```

Rappelons que chaque résumé z contient dans sa description les pairs (P_z) qui sont décrits par ce résumé. Soit P_Q l'ensemble des pairs pertinents pour une requête Q. Le

principe de l'algorithme est : pour chaque nœud z du résumé qui correspond exactement à la requête, P_z est ajouté à l'ensemble P_Q. Puis, la requête est propagée vers l'ensemble des pairs contenus dans P_Q. Chaque pair qui a reçu le message, doit vérifier la satisfaction de la requête par rapport à ses données stockées localement (voir la sous-section suivante) et ainsi identifier les meilleurs tuples. Ensuite, il renvoie la réponse vers l'initiateur de requête. Lorsque le pair initiateur reçoit tous les résultats des pairs de l'ensemble P_Q, il filtre les résultats et renvoie seulement les réponses les plus satisfaisantes.

6.2 Evaluation de la requête sur chaque pair

Après que les pairs contribuant à la réponse d'une requête ont été identifiés. Il est nécessaire d'évaluer la requête flexible sur les données locales de chaque pair et donc la hiérarchie des résumés des données du pair (l'index local) est explorée pour trouver les résumés qui satisfont la requête. Comme les résumés feuilles regroupent les tuples candidats qui sont réécrits par les mêmes descripteurs linguistiques, on étend l'algorithme 1 pour qu'il retourne les enregistrements initiaux (tuples) ayant les degrés de satisfaction les plus élevés. L'algorithme 3 illustre cette procédure.

Algorithme 3. Fonction Recherche_Tuples (z, C)

Entrée : Résumé z et la caractérisation C
1. L_{Ct} := ϕ /* la liste des tuples candidats est vide */
2. L_{meil_Ct} :=ϕ /*liste des meilleurs tuples candidats vide */
3. $L_{réponse}$:= ϕ /* la liste des réponses est vide */
4. Si Corr(z, C) = excès Alors
5. Pour chaque nœud fils z_{fils} de z Faire
   L_{Ct} := L_{Ct} ∪ Recherche_Tuples (z_{fils}, C)
6. Fin pour
7. Sinon
8. Si Corr(z, C) = exacte Alors
9. Si z est une feuille Alors
10. Pour un tuple candidat Ct de z Faire
11. ajouter(Ct, L_{Ct})
    Fin pour
12. Sinon
13. L_{Ct} := L_{Ct} ∪ Recherche_Tuples (z, C)
14. Fin si
15. Fin si
16. Fin si
17. L_{meil_Ct} := BNL (L_{Ct})
18. Pour chaque Ct de L_{meil_Ct} faire
19. retourner le tuple original t
20. ajouter(t, $L_{réponse}$)
21. Fin pour

Résultat : $L_{réponse}$

La recherche est basée sur un test de correspondance comme dans l'algorithme 1, mais dans ce cas les résumés sont des index sur les données recherchées, quand un résumé feuille *z* correspond exactement à la requête, l'ensemble des tuples candidats couverts par *z* est ajouté à la liste des tuples candidats (L_{Ct}). Ensuite, la fonction *BNL*[3] retourne les meilleurs tuples L_{meil_Ct} parmi la liste L_{Ct}.

Une connexion à la base de données est nécessaire pour chaque tuple candidat *Ct* de la liste L_{meil_Ct}. Elle permet d'identifier les tuples de la base qui lui correspond. Le résultat final de l'algorithme est donc l'ensemble des tuples satisfaisant au mieux la requête.

7 Un Exemple Illustratif

Reprenons l'exemple de référence de la Section 2 et considérons la requête flexible *Q* concernant la recherche des caméras qui sont "*Pas_Trop_Cher*" et ayant une "*Bonne*" qualité. La solution proposée consiste premièrement à la construction de l'index de routage. Puis l'évaluation de la requête Q sur chaque source.

Pour construire l'index de routage, le pair 1 résume ses données par le résumé *Z1* comme expliqué en Section 4.1. De la même manière, les pairs 2 et 3 résument leurs données par *Z2* et *Z3*, voir Tableau 4 (pour plus de détails voir [1])

Tableau 4. Résumé global pour le réseau P2P.

Site 1 (Z_1)	(1.0/Ch, 1.0/P_T_C, 1.0/T_C, 0,1/Mauv, 1.0/Moy, 1.0/Bon, 1.0/Meil)
Site 2 (Z_2)	(0.5/B_Marché, 1.0/M_Ch, 0.5/Ch, 1.0/T_Ch, 1.0/Moyen, 1.0/Meil)
Site 3 (Z_3)	(1.0/M_Ch, 1.0/Ch, 1.0/P_T_C, 1.0/T_C, 1.0/Moy, 1.0/Bon, 1.0/Meil)
Résumé global (Z)	({0.5/B_Marché,1.0/M_Ch,1.0/Ch, 1.0/P_T_C, 1.0/T_C}, {0.1/Mauv, 1.0/Moy, 1.0/Bon, 1.0/Mei})

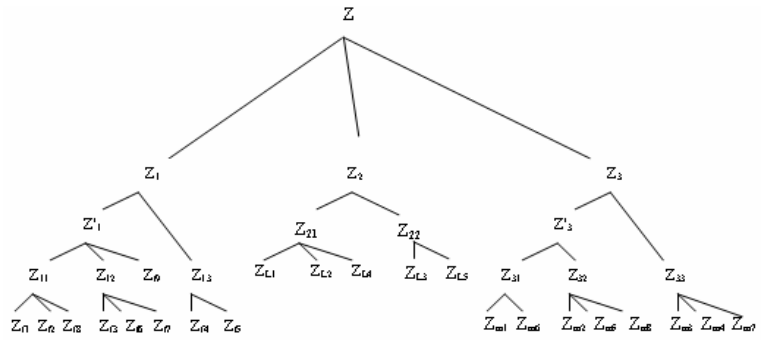


Fig 4 Hiérarchie de résumés pour le réseau P2P

Les deux pairs feuilles (site 1, site 3) envoient leurs résumés de données (index locaux) vers leur voisin (le nœud central site 2). Le pair 2 fusionne les résumés reçus

[3] basée sur le principe d'optimalité de Pareto, voir [2].

de ses voisins avec son propre résumé et délivre un résumé *Z* global (index global) qui décrit l'ensemble des données du réseau. Enfin, l'index *Z* (voir Fig 4) est renvoyé aux nœuds feuilles. Le Tableau 4 présente l'index local de chaque pair ainsi que l'index global construit.

Maintenant la requête Q est redirigée vers le pair 2. Celui-ci commence par exécuter l'algorithme 2, sur l'index global, pour déterminer les pairs pertinents. Premièrement, la racine *Z* correspond à la requête par excès, il faut tester ses nœuds fils, le nœud *Z2* à une correspondance nulle par rapport à la requête, donc le sous arbre de racine *Z2* est éliminé et ne peut être considéré par la suite, alors que les sous arbres de racines *Z1* et *Z3* (respectivement) sont considérés et ainsi de suite. Pour *Z1*, le résumé (nœud) Z_{f7} correspond exactement à la requête et son pair P_z= *{site1}*, alors que pour *Z3*, *Pz= {site3}*. Le résultat final de l'algorithme est P_Q = *{pair1, pair3}*.

La requête est donc routée vers les pairs 1 et 3. Chaque pair exécute l'algorithme 3. Le résultat de cet algorithme sur le pair 1 est le résumé Z_{f7} qui couvre les tuples candidats : {Ct_{61}, $Ct_{7,2}$, $Ct_{11,2}$} avec $Ct_{6,1}$ = (1.0/P_T_C, 0.9/Bon), $Ct_{7,2}$ = (0.8/P_T_C, 1.0/Bon) et $Ct_{11,2}$ = (1.0/P_T_C, 0.9/Bon). Les tuples de la base retournés sont donc {S6, S7, S11}. De la même façon, on peut vérifier que l'application de l'algorithme 3 sur le site 3 retourne les tuples candidats : {$Ct_{1,2}$, $Ct_{1,2}$} avec $Ct_{1,2}$ = (0.83/P_T_C, 1.0/Bon). Les tuples de la base retournés sont donc {X1, X11}.

Ensuite les deux pairs envoient leurs réponses au site initiateur de la requête, le site 2, qui fusionne ces résultats et délivre la réponse finale à la requête Q, notée $\sum_Q$, sous forme d'un ensemble flou de la forme :

$$\sum_Q = \{min(1.0,1.0)/S6,\ min(1.0,0.9)/S11,\ min(0.83,1.0)/X1,\ min(1.0,0.9)/X11\}$$
$$= \{1.0/S6,\ 0.9/S11,\ 0.9/X11,\ 0.83/X1\},$$

où chaque élément est associé avec un degré de satisfaction. On peut observer que la caméra *S6*, qui se trouve dans le site 1, satisfait totalement la requête *Q*. Par contre, la caméra *X11* du site 3 satisfait que partiellement la requête *Q* (avec un degré de 0.9).

8 Conclusion

Dans cet article, nous avons abordé le problème d'évaluation de requêtes flexibles dans un système P2P. Un index de routage global et distribué est proposé. Deux algorithmes ont également été étudiés. Le premier traite le problème de la localisation des pairs pertinents à une requête posée. Le second permet de retourner les tuples, contenus dans les sources associées aux pairs du système, qui satisfont au mieux la requête.

L'étude présentée ici est encore préliminaire et beaucoup de perspectives existent quant à des travaux futurs. Un premier travail à court terme concerne l'implémentation des algorithmes développés et la réalisation d'expérimentations afin de montrer la pertinence et l'efficacité de l'approche. Un second axe consiste à considérer d'autres modèles de résumé comme, par exemple, la typicité [4][14] des données contenues dans une source associée à un pair du réseau P2P.

Références

1. A. Alem, Contribution à l'étude de requêtes à préférences dans un système P2P, Mémoire de Magister, Dépt. d'Informatique, Université de Tiaret (Algérie), Novembre 2009.
2. S. Borzsonyi, D. Kossmann, K. Stocker, The skyline operator. Proc. 17th IEEE Inter. Conf. on Data Engineering, Heidelberg, 2001, pp. 421-430
3. P. Bosc, L. Liétard, O. Pivert, D. Rocacher, Gradualité et imprécision dans les bases de données, Paris, Editions Ellipses, 2004
4. P. Bosc, A. Hadjali, H. Jaudoin, O. Pivert, Flexible querying of multiple data sources through fuzzy summaries, Proc. of FlexDBIST'07, in conjunction with DEXA'07, Regensburg, Germany, September 3-7, pp. 350-354, 2007
5. J. Chomicki, Preference formulas in relational queries, ACM Transactions on Database Systems, 27, 2003, pp. 153-187.
6. A. Crespo, H. Garcia-Molina, Routing Indices for Peer-to-Peer Systems, Inter. Conference on Distributed Computing Systems, 2002.
7. A. Hadjali, S. Kaci, H. Prade, Database preferences queries - A possibilistic logic approach with symbolic priorities, Proc. of the 5th Inter. Symposium on Foundations of Information and Knowledge Systems (FoIKS'08), LNCS 4932, pp. 291–310, 2008.
8. R. Hayeky, G. Raschiay, P. Valduriezz, N. Mouaddib: Summary Management in P2P Systems, EDBT'08, 2008, Nantes, France
9. K. Hose, C. Lemke, K. Sattler, Processing relaxed skylines in PDMS using distributed data summaries, Proc. CIKM'06, USA, 2006
10. http://www.gnutella.com
11. http://www.napster.com
12. H. Li, Q. Tan, W. Lee, Efficient progressive processing of skyline queries in P2P systems, International Conference on Scalable Information Systems (INFOSCALE'06), 2006
13. E. Lo, KY Yip, K.I Lin, D.W. Cheng, Progressive skylining over Web-accessible databases, Data & Knowledge Engineering, 57, pp. 122-147, 2006
14. D. Merad Boudia, Contribution à l'étude de la typicité en vue de son application à l'interrogation flexible de bases de données, Dépt. d'Informatique, Université de Tlemcen (Algérie), Novembre 2009
15. G. Raschia, N. Mouaddib, A fuzzy set-based approach to database summarization, Fuzzy Sets and Systems 29(2), pp. 137– 162, 2002
16. I. Stoica, R. Morris, D. Karger, F. Kaashoek, H. Balakrishnan, Chord: A scalable Peer-To-Peer lookup service for internet applications. Proc. of the ACM SIGCOMM, pp. 149–160, 2001
17. W.A. Voglozin, G. Raschia, L.Ughetto, N. Mouaddib, Querying a summary of database, J Intell Inf Syst, 26, 2006, pp. 59–73
18. S. Wang, B. Ooi, A. Tung, L. Xu, Efficient Skyline query processing on P2P Networks, Proc. ICDE'07, 2007
19. D. Zinn, Skyline Queries in P2P Systems, Master's Thesis, TU Ilmenau, Germany, 2005

Supporting Failing Database Queries in a Flexible Context: A Data-Driven Approach

Lila Oudjoudi[1] and Allel Hadjali[2]

[1] ESI, BP 68M, 16270, Oued Smar, Algérie
l.oudjoudi@gmail.com

[2] IRISA/ENSSAT, University of Rennes 1
Technopole Anticipa 2205 Lannion Cedex France
hadjali@enssat.fr

Abstract. We investigate the problem of handling of failing queries involving fuzzy predicates. We propose an approach that leverages data distribution of the target database. It consists in two steps: i) Query translation that aims at translating the failing fuzzy query into a crisp query by means of a particular semantic distance between sets; ii) Query relaxation which consists in expanding the translated query criteria with similar values. To rank-order the approximate query results, a method is proposed

Keywords: Flexible queries, empty answers, semantic distance, similarity measures, relaxation.

1 Introduction

The practical need for endowing intelligent information systems with the ability to exhibit cooperative behavior has been recognized since the early' 90s. The most well-known problem approached in this field is the *failing query problem*: users' queries return an *empty set of answers*. Several approaches have been proposed to deal with this issue, see [9] for an overview. Most of them rely on the relaxation paradigm that aims at expanding the scope of a query searching for answers that are in the neighborhood of the original user's query.

On the other hand, relying on *flexible (or fuzzy) queries* (i.e., queries that could contain fuzzy constraints) has the main advantage of diminishing the risk of empty answers. Indeed, fuzzy queries express preferences and retrieve elements that are more or less satisfactory rather than necessarily ideal. However, it still may happen that the database does not have any element that satisfies, even partially, the fuzzy criteria formulated by the user. Only few works have been done for dealing with this problem in the fuzzy database querying [2][4][10][18]. They mainly aim at relaxing the fuzzy requirements involved in the failing query. *Query relaxation* can be achieved by applying an appropriate transformation to gradual predicates of a failing query. Such a transformation aims at modifying a given predicate into an enlarged one by *widening its support*. Recently, other kind of approaches which are based on

leveraging a past query workload (log of past user queries) have been proposed in [3][13]. The principle consists in replacing the failing query by the most similar one among the queries of the workload. All the approaches can be viewed as query-driven methods, i.e., they primarily operate on the failing query.

In this paper, we propose an approach for dealing with failing flexible conjunctive queries that leverages data distribution of the target database. It constitutes another direction to address the problem in a flexible context; the idea is somewhat close to the principle of similarity search. Instead of relaxing the failing initial query, we first look for the values in the database that are maximally close to the fuzzy predicates specified in that query and then we explore the neighborhoods of such values. Informally speaking, the approach proceeds in two steps: i) Query translation that aims at translating the failing fuzzy query into a (crisp) point query by means of a particular semantic distance measure between sets; ii) Query relaxation which consists in expanding the translated query criteria with similar values. To rank-order the query results, a ranking method is proposed.

The paper is structured as follows. Some basic notions are introduced in section 2. In section 3, we review some related work. Section 4 provides an overview of the approach proposed. Section 5 discusses the relaxation of both categorical and numerical query conditions. Section 6 describes a query results ranking method by learning attribute importance weights. In section 6, we conclude and outline some future works.

2 Basic Notions

2.1 Flexible Queries

Flexible queries [6] are requests in which user's preferences can be expressed. Here, the fuzzy sets framework is used as a tool for supporting the expression of preferences. The user does not specify crisp conditions (Boolean predicates), but fuzzy ones (which correspond to fuzzy predicates such as *Young*, *Tall* or *Cheap*) whose satisfaction may be regarded as a matter of *degree*. As a consequence, the result of a query is no longer a flat set of elements but is a set of discriminated elements according to their global satisfaction of the fuzzy constraints appearing in the query.

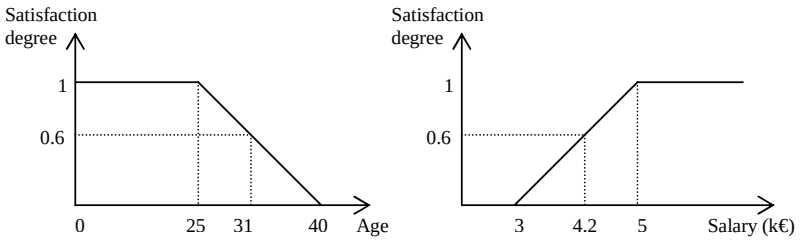


Fig. 1. Fuzzy predicates *Young = (0, 25, 0, 15)* and *Well-paid = (5, +∞, 2, 0)*.

An elementary fuzzy predicate P can be modeled as a function μ_P from a domain U to the unit interval. The degree $\mu_P(u)$ represents the extent to which element u satisfies the vague predicate P (or equivalently the extent to which u belongs to the fuzzy set of objects which match the fuzzy concept P). Here, we use trapezoidal membership functions (t.m.f.) that can be encoded by a quadruplet (A, B, a, b) where [A, B] (resp. [A-a, B+b]) represents the core (resp. the support) of P. A typical example of a fuzzy query is: "retrieve the employees which are *Young* and *Well-paid*", see Figure 1.

2.2 Semantic Distance

We introduce here a particular semantic distance between fuzzy (or crisp) sets. It relies on the *Hausdorff distance measure* whose principle is reviewed hereafter.

***2.2.1 Crisp Sets*:** Consider two subsets A and B of a space U (equipped with a metric). The most popular scalar extension of distance between A and B is the *Hausdorff distance* defined as [7][11]:

$$d_H(A, B) = max\{H(A, B), H(B, A)\}, \quad (1)$$

where $H(A, B)$ stands for the *directed Hausdorff distance* from A to B. We have $H(A, B) = Sup_{u\in A}\ d(u, B)$ and $d(u, B) = Inf_{v\in B}\ d(u, v)$. The expression $d(u, v)$ stands for a standard distance (such as Euclidean distance). Formula (1) can be written in the following condensed form:

$$d_H(A, B) = max\{sup_{u\in A}\ inf_{v\in B}\ d(u, v), sup_{v\in B}\ inf_{u\in A}\ d(u, v)\}. \quad (2)$$

The idea that governs this distance is the following: for each element in A look for the closest element in B, then check for the element in A for which the distance to the closest element in B is maximal. The same is done exchanging B and A and the *longest* distance of the two component is kept. Intuitively, if the Hausdorff distance is δ, then every point of A must be within a distance δ of some point of B and vice versa.

Example 1. Let $A = [a_1, a_2]$ and $B = [b_1, b_2]$ be two regular intervals and let $d(u, v) = |u - v|$. Then, it easy to check that $d_H(A, B) = max(|a_1 - b_1|, |a_2 - b_2|)$. ♦

***2.2.2 Fuzzy Sets*:** The Hausdorff distance between fuzzy sets can be either fuzzy or scalar. Hereafter, we only focus on the scalar version. For the fuzzy evaluation, more details are available in [11]. Here, we use the definition proposed in [7]. This definition is more general and is valid in the case of two fuzzy sets with unequal maximum memberships. In the following, we consider only fuzzy sets with the same supremum.

Let F and G be two discrete fuzzy sets. Let $T = \{t_1, t_2, \ldots, t_m\}$ the set of all the distinct membership values of F and G. The Hausdorff distance between F and G is defined by the following expression:

$$d_H^2(F,G) = \frac{\sum_{i=1}^{m} t_i d_H(F_{t_i}, G_{t_i})}{\sum_{i=1}^{m} t_i}\ , \quad (3)$$

where F_{t_i} (resp. G_{t_i}) stands for the t_i-level cut[1] of F (resp. G). $d_H^2(F,G)$ can be seen as a membership-weighted average of the crisp Hausdorff distances between the level sets of the two fuzzy sets.

Example 2. Let *U = {1, 2, 3, 4, 5, 6, 7}* be a universe of discourse. Let also F and G be two discrete fuzzy sets on U defined as follows: *F = {0.7/1, 0.2/2, 0.6/4, 0.5/5, 1/6}* and *G = {0.2/1, 0.6/4, 0.8/5, 1/7}*. One can see that *T = {0.2, 0.5, 0.6, 0.7, 0.8, 1}*.

Table 1. The Hausdorff distance between the α-cuts of F and G

α_i	F_{α_i}	G_{α_i}	$H(F_{\alpha_i}, G_{\alpha_i})$	$H(G_{\alpha_i}, F_{\alpha_i})$	$d_H(F_{\alpha_i}, G_{\alpha_i})$
0.2	{1, 2, 4, 5, 6}	{1, 4, 5, 7}	1	1	1
0.5	{1, 4, 5, 6}	{4, 5, 7}	3	1	3
0.6	{1, 4, 6}	{4, 5, 7}	3	1	3
0.7	{1, 6}	{5, 7}	4	1	4
0.8	{6}	{5, 7}	1	1	1
1	{6}	{7}	1	1	1

By formula (3), and using Table 1, we get

$d_H^2(F,G) = (0.2 \cdot 1 + 0.5 \cdot 3 + 0.6 \cdot 3 + 0.7 \cdot 4 + 0.8 \cdot 1 + 1 \cdot 1) / 3.8 \cong 2.13$ ♦

In case of continuous fuzzy sets, formula (3) is modified in the following form [7]:

$$d_H^2(F,G) = \int_0^1 t d_H(F_t,G_t)dt \Big/ \int_0^1 t dt = 2\int_0^1 t d_H(F_t,G_t)dt \qquad (4)$$

Example 3. Let now U represent the numeric universe of discourse of the variable "*age*" of a person. Let also *F* = "*about thirty*" and *G* = "*between_26_and_28*" two fuzzy sets on U defined by the following two *t.m.f.*: *F = (30, 30, 3, 3)* and *G = (26, 28, 1, 1)*. One can observe that $F_\alpha = [3\alpha + 27, 33 - 3\alpha]$ and $G_\alpha = [\alpha + 25, 29 - \alpha]$. Then, Applying formula (4), we get

$d_H^2(F,G) = 2\int_0^1 t \max(|(t+25)-(3t+27)|, |(29-t)-(33-3t)|)dt = 7/2$ ♦

It has been pointed out in [7] that expression (3) (resp. (4)) is a metric and reduces to the classical Hausdorff distance when sets are crisp.

3 Related Work

Several Works have been proposed to deal with the empty answers problem. Such works can be found in both domains of databases and information retrieval, including web search. Due to space limitation, we only provide here a review on some approaches proposed in the database fuzzy querying context. See [9][4][16][14] for an overview of the approaches suggested in the crisp queries context.

In the fuzzy querying setting, approaches can be classified into two main categories and are mainly query-driven. The first one is based on the relaxation

[1] An α-levet cut of the fuzzy set F is defined as $\{u \in U, \mu_F(u) \geq \alpha\}$.

paradigm. Query relaxation aims at expanding the scope of a query searching and consists in modifying some query conditions by enlarging them or just eliminating some of them. Andreasen and Pivert [2] have proposed an approach where the basic modification used relies on a particular *expansive linguistic modifier*. This approach is merely a technical operation, lacking of any semantics. Moreover, it provides no intrinsic *semantic limits* for controlling the relaxation process and fails to deal with classical crisp queries. In [5][4] a relaxation method is proposed that makes use of a parameterized proximity relation. Given a fuzzy predicate P, the idea is to compute the set of predicates that are close to P in the sense of the proximity relation defined on the domain of P. Even, if this method is endowed with a clear semantics, it can lead to a combinatory explosion induced by the relaxation of the predicates from a conjunctive query. To know whether these relaxed queries provide a non-empty answer, one has to evaluate them. In [18], the authors consider flexible queries addressed to data summaries and propose a method based on a specified distance to repair failing queries. If no summary fits a query *Q*, alternative queries are generated by modifying one or several fuzzy labels involved in *Q*. This requires a *pre-established order* over the considered attributes domains since a label is replaced by the closest one. The resulting queries are ordered according to their closeness to the original one (measured by the specified distance). See also the work done in [10].

The second category is based on leveraging a past query workload (i.e., a collection of queries that have been executed on the database system in the past and have produced non-empty answers). The principle consists in replacing the failing query by the most similar (semantically speaking) one among the queries of the workload. To compute the proximity between queries, a measure of substitution is suggested in [3] which assumes the availability of a resemblance relation over every attribute domain involved in the target database. An alternative proximity query measure is studied in [13]. It relies on a particular distance between sets, called the Hausdorff distance. Only attributes with domains endowed with a metric have been considered in this work.

Our work is inspired from [1] and [17] for computing the importance weights for each specified attribute and for deriving the similarity coefficients between two (categorical or numerical) values. In [1], an automatic ranking method based on Information Retrieval (IR) techniques for the empty answers problem is proposed. The importance scores of tuples are extracted using a workload and a data analysis. In [17], a system called AIMQ is proposed to address the problem of answering imprecise queries over Web databases. It learns attribute importance and values similarity measures from the database. It can only determine the attribute importance sequence (without the specific weights). This result is invariant for the different user queries. See also [12] for the incremental version of AIMQ, called IQPI. Let us mention the work done in [15] that uses, in a similar way as above, data and query workload statistics for relaxing crisp queries in order to provide approximate answers to the user.

Our approach differs from that in [2][3][4][5][13] in leveraging only data distribution for relaxing failing queries, and from [1][17][15] in focusing on fuzzy queries.

4 Overview of the Approach

Let us first state the problem of interest. Assume that $\mathcal{D}$ is a (Web) regular database with categorical and numerical attributes $A = \{A_1, \ldots, A_m\}$ and $D(A_i)$ represents the domain of values of attribute A_i in the database $\mathcal{D}$. Let also $Q = P_1 \wedge \ldots \wedge P_k$ $(k < m)$ be a conjunctive flexible query where the symbol '$\wedge$' stands for the connector *'and'* and is interpreted by the '*min'* operator. Let Σ_Q be the set of answers to Q over $\mathcal{D}$. The set Σ_Q contains the items of the database that satisfy *somewhat* the fuzzy requirements involved in Q, i.e., each item has a strict positive satisfaction degree.

Definition. We say that Q is a failing query if $\Sigma_Q = \varnothing$.

This means that no data in the database somewhat satisfies all of the fuzzy conditions involved in Q. In the literature, this problem is known as the *Empty Answers Problem*.

Let us assume that Q is a failing user query. To deal with this problem, one way is to provide approximate answers to the user. To this end, we propose a data-driven approach that leverages the data distribution of the target database. It consists in a two-step procedure (see Figure 2):

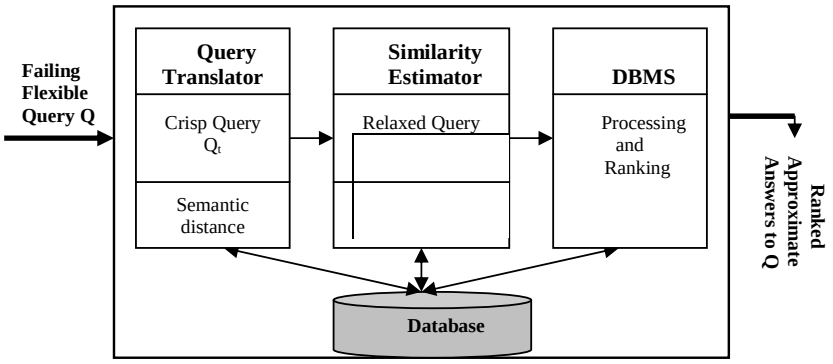


Fig.2. Architecture of the approach

Step 1: Query Translation. It aims at translating Q into a (crisp) point query of the form:

$$Q_t = v_1 \wedge \ldots \wedge v_k,$$

where $v_i \in D(A_i)$. For each P_i, $i = 1, \ldots, k$, pertaining to attribute A_i, we look for the point $v_i \in D(A_i)$ that is maximally close, semantically speaking, to the fuzzy set modelling P_i. To do so, we assess the semantic distance between the set E_i and P_i where the t.m.f. of E_i is given by $(v_i, v_i, 0, 0)$ (resp. $\{1/ v_i\}$) if P_i is a numerical (resp. categorical) attribute. This can be achieved by computing the Hausdorff distance between E_i and P_i (discussed in Section 2.2).

Example 4. To illustrate this step, consider a relation *EMP(Name, Age, Salary)* describing employees of a company, whose extension is given in Table 2. Consider also the query Q = *"find employees who are young and earn around 38 k$"*. Q can

simply write $Q = (Age = Young) \wedge (Salary = Around\_38)$ where *Young* and *Around_38* are fuzzy sets represented respectively by the t.m.f. *(0, 25, 0, 10)* and *(38, 38, 2, 2)*. One can observe that Q returns an empty answer when evaluating over relation EMP of Table 2.

Now according to step 1 and using the Hausdorff distance between each *Age.value* (resp. *Salary.value*) and *Young* (resp. *Around_38*), *Q* is rewritten in $Q_t = (Age = 38) \wedge (Salary = 35)$. ♦

Table 2. Extension of Relation EMP

Name	**Age**	**Salary** (k$)	$\mu_{Young}(u)$	$\mu_{Around\_38}(v)$
Dupont	48	45	0	0
Martin	46	42	0	0
Durant	42	35	0	0
Dubois	38	28	0	0
Lorant	40	30	0	0

Now according to step 1 and using the Hausdorff distance between each *Age.value* (resp. *Salary.value*) and *Young* (resp. *Around_38*), *Q* is rewritten in $Q_t = (Age = 38) \wedge (Salary = 35)$. ♦

Remark. If there are several values $\{v_i^1, \ldots, v_i^h\}$ from $D(A_i)$ that are maximally close to P_i, we translate the fuzzy query condition $(A_i \text{ is } P_i)$ into $(A_i \in \{v_i^1, \ldots, v_i^h\})$.

Step 2: Query Relaxation. We rewrite Q_t under the form

$$Q_t = C_1 \wedge \ldots \wedge C_k$$

where $C_i = (A_i = v_i)$ for $i = 1,\ldots, k$. Then, for each condition C_i in Q_t, we extract values of its corresponding attribute A_i having similarity factor above some user-defined sub-threshold λ_i. Then, we add those values into the range of C_i. Thus, we get a relaxed version of C_i, denoted by $\widetilde{C_i}$. A relaxed version, denoted by $\widetilde{Q_t}$, of the query Q_t is then built by joining all the relaxed conditions $\widetilde{C_i}$.

Full details about the above two steps will be provided in the next sections.

5 Query Relaxation

The idea of query relaxation advocated consists in expanding the translated query criteria with similar values. So, we need to measure the similarity between the different pairs of values.

5.1 Relaxation of Categorical Query Conditions

We present an approach for measuring the similarity index between two categorical attribute values. This approach is borrowed from [17]. The similarity between two

values binding a categorical attribute is measured as the percentage of common *Attribute-Value pairs* (AV-pairs) that are associated to them. Consider a used car selling Web database *CarDB(Make, Model, Price, Color, Year).* Each tuple in *CarDB* represents a used car for a sale. For instance, *Make = Ford* is AV-pair over the database CarDB.

- Each AV-pair is considered as a selection query and submitted to (a sample of) the database, separately.
- The result of running each query is a set of tuples which is called a *supertuple (ST).*
- The supertuple contains a bag of keywords for each attribute in the relation not bound by the AV-pair.

For example, Table 3 shows the supertuples of the AV-pair "*Make = Toyota*" and "*Make = Ford*" over the database *CarDB.*

Table 3. The supertuples obtained from running

(a) the query "Make = Toyota".

Model	Camry: 3, Corolla: 4
Price	10k-15k: 4 , 15k-20k: 3
Color	Blue: 1 , Black: 3 , White : 3
Year	2005: 2 , 2006: 3 , 2007 : 2

(b) the query "Make = Ford"

Model	Focus: 2, F150: 3
Price	10k-15k: 3 , 15k-20k: 2
Color	Blue: 2 , Red: 2 , White : 1
Year	2005: 1 , 2006: 4

The values within the supertuple of Table 3-(a) indicate that there are totally *7* records in the database having "*Make = Toyota*". ♦

The similarity between two attribute values (AV-pairs) is measured as the similarity shown by their supertuples. This latter is measured by using the *Jaccard* coefficient. Let ST_1 and ST_2 be two supertuples with *m* attributes and A_i is i^{th} attribute, we have

$$VSim(ST_1, ST_2) = \sum_{i=1}^{m} J(ST_1.A_i, ST_2.A_i),$$

where *J(.,.)* stands for the Jaccard Coefficient and is computed as *J(A, B)* = $/A \cap B/ / /A \cup B/$. Consider for instance the attribute "*Make*", if we want to measure the similarity between the value "*Toyota*" and the value "*Ford*". First, we compute the supertuple ST_1 (resp. ST_2) resulting from the query "*Make = Toyota*" (resp. "*Make = Ford*)" (See Table 3). Then, we compute $VSim(ST_1, ST_2)$ = (2·0)/(4·7) + (2·5)/(4·7) + (2·1)/(3·10) + (1·4)/(4·8) ≈ 0.54 [2]. So, *VSim(Toyota, Ford) =* $VSim(ST_1, ST_2)$ *= 0.54.*

[2] Here, we use *Jaccard Coefficient* with bag semantics to determine the similarity between two supertuples, see [17].

Now if needed, one can normalize the above similarity measure by using for instance the arithmetic mean, i.e., $VSim(ST_1, ST_2) = \frac{1}{m}\sum_{i=1}^{m} J(ST_1.A_i, ST_2.A_i)$.

5.2 Relaxation of Numerical Attribute Values

To estimate the similarity coefficient between a pair of different numerical values, we use an approach which is inspired from [8][1]. Let $\{a_1,.., a_n\}$ be the values of numerical attribute A occurring in the database. Then the similarity coefficient $VSim(a, v)$ between the two values a and v can be defined by the following equation

$$VSim(v, a) = 1 /(1 + ((a - v)/h)^2)$$

where h is the bandwidth parameter. A popular estimation for the bandwidth is $h = 1.06\sigma n^{-1/5}$ where σ is the standard deviation of $\{a_1, ….., a_n\}$, see [1] for more details. Let λ be a given similarity threshold, and v the numerical value specified by the query. Then, one can observe that the values that have similarity degree (above λ) with v are restricted by the following interval:

$$I(v,\lambda) = \left[v - h\sqrt{(1-\lambda)/\lambda}, v + h\sqrt{(1-\lambda)/\lambda}\right]$$

Input: $Q_t = \{C_1, …, C_k\}$ with $C_i = (A_i = v_i)$ for $i = 1,k$
Sub-thresholds $\{\lambda_1, …, \lambda_k\}$

1. $\tilde{Q}_t = \emptyset$; $i := 1$;
2. **while** $i \leq k$ **do**
3. **begin**
5. $\tilde{C}_i := C_i$;
6. **if** A_i is numerical attribute **then**
7. replace the range of $\tilde{C}_i$ with $I(v_i, \lambda_i)$;
8. **if** A_i is categorical attribute **then**
9. **For** a **in** $D(A_i)$ **do**
10. **If** $VSim(v_i, a) = VSim(ST(v_i), ST(a)) > \lambda_i$ **then**
11. add a into the range of
12. **endif**
13. **endif**
14. $\tilde{Q}_t := \tilde{Q}_t \cup \tilde{C}_i$
15. $i := i + 1$;
16. end

Output: the query relaxation $\tilde{Q}_t = \tilde{C}_1 \wedge \ldots \wedge \tilde{C}_k$

Algorithm 1. Query relaxation (where $ST(v)$ is the *supertuple* associated with the value v).

5.3 Query Rewriting

Let us assume that the sub-threshold λ_i *(i = 1, k)* for each specified attribute in *Q* is given by the user. The principle of the query relaxation algorithm (see Algorithm 1) is to replace each condition $C_i = (A_i = v_i)$ involved in Q_t by its relaxed variant, $\widetilde{C}_i$, as explained in Sections 5.1 and 5.2.

Remark. If the relaxed query, $\widetilde{Q}_t$, resulting from Algorithm 1 still returns an empty answer set, one can re-execute Algorithm 1 by assigning other appropriate values to the sub-threshold λ_i.

Note that for large-scale databases, the evaluation of the relaxed query may result in too many relevant answer items. So, it is extremely desirable to rank such query results according to their relevance. This is what we will discuss in the next section.

6 Results Ranking Strategy

One factor that can affect query results ranking is the attribute importance weights (since attribute importance of the same attribute is usually different for users). It would then be interesting to automatically learn such importance weights. Approaches for estimating attribute importance can be divided into two classes [17]: (i) *data driven* where the attribute importance is identified using the data distribution of the database; (ii) *query driven* where the importance of an attribute is determined by the frequency with which it appears in user queries. So, this last technique requires a database workload (log of past user queries) which constrains its use for new systems. In the following, we use the first technique to learn the importance of each attribute by leveraging the distribution of its value specified in the query in the database.

6.1 Attribute Weight Assignment

The idea is to associate a weight to each specified attribute according to the distribution of its value in the database. For instance, for a query with condition "*Year = 2008* and *Price < 10000*", the specified attribute *Year* may have less importance for user (there may be many used car have the date of shipment in 2008) than the attribute *Price* (relatively fewer used cars priced below $10000).

To this end, we use the well-known Inverse Document Frequency (*IDF*) factor that has been used extensively in IR. *IDF* suggests that commonly occurring words convey less information about user's needs than rarely occurring words, and thus should be weighted less. Recall that *IDF(w)* of a word *w* is a measure indicating how many documents in which *w* appears. We can then adapt this technique to our problem by considering each database tuple (and query) as a small document [1].

- Categorical Attribute

For every *v* in the domain of attribute A_i, we define $IDF_i(v)$ such that

$$IDF_i(v) = log(n/F_i(v)),$$

where n is the number of tuples in the database and $F_i(v)$ is the frequency of tuples t in the database where $t.A_i = v$. In the following, the importance of specified categorical attribute value is treated as the importance of its corresponding attribute.

- Numerical Attribute

For numerical data, the definition of traditional *IDF* as above is inappropriate. The frequency (and hence the *IDF*) of a numerical value should depend on nearby values. To measure the closeness of numerical attribute values, we make use of a robust definition proposed in [1]. Let $V_i = \{v_i^1, \ldots..., v_i^n\}$ be the set of values of attribute A_i that occur in the database. For any value v, $IDF_i(v)$ is defined in the following way (where h is the bandwidth parameter mentioned in section 5.2):

$$IDF_i(v) = log\left(\frac{n}{\sum_{j=1}^{n} e^{-\frac{1}{2}\left(\frac{v_i^j - v}{h}\right)}}\right)$$

The intuition of the above formula is: the denominator represents a numeric extension of the concept of frequency of v (the sum of "contributions" to v from every the other point v_i^j in the database). The further v is from v_i^j, the smaller its contribution. The importance of specified numerical attribute value is treated as the importance of its corresponding attribute

Denoting by $W_i(A_i)$ the weight associated with attribute A_i specified in the user query and by applying a normalization, we finally obtain:

$$W_i(A_i) = \frac{IDF_i(v_i)}{\sum_{j=1}^{k} IDF_j(v_j)}$$

6.2 Ranking

To rank-order the answer tuples t returned to the relaxed query, one can use the following ranking score:

$$d_{Q_t}(t) = \frac{1}{k}\sum_{i=1}^{k} W_i(A_i) \times Sim(Q_t.A_i, t.A_i),$$

where $W_i(A_i)$ is the importance weight of attribute A_i specified in the original query Q, and $Sim(v, a)$ stands for the similarity measures between categorical or numerical attribute values as explained in section 5.

The idea is that the larger the similarity score $d_Q(t)$ is, the higher the ranking score is for the result tuple t. Thus, we can provide the end-user with the top-N answers according to the above ranking.

7 Discussion and Conclusion

In this paper, an alternate approach for dealing with failing queries in a flexible context is proposed. Starting from the user query, we translate the original query into a crisp query and then we rewrite this resulting query by relaxing the query criteria ranges. The two key concepts of the approach are the distance semantic introduced and the similarity measures discussed (between two categorical or numerical attribute values). Ranking the relevant answers is also investigated by learning the importance of each attribute specified in the query. Since the approach mainly operates on the actual data of the queried database, it could constitute a promising alternative to approximately answer a failing query.

As can be seen, the approach requires some pre-computations and some additional access to the entire database before getting final approximate answers to the failing query. Pre-computations mainly consists in: i) calculating the semantic distance between attribute values in the database and the fuzzy predicate present in the query for every attribute specified in the query; ii) measuring the similarity between each value of attribute involved in the translated query and values binding this attribute in the database. Let us take a look at the second type of calculus in case of categorical attribute. Instead of performing this calculus for each failing query, one can measure the similarity between every pair of values binding this attribute once and for all (provided that database will not change), and the similarity results will be stored in a Table. To mitigate also the problem of scanning the entire database, one can select a sample dataset (sufficiently representative) for estimating the similarity measures.

We acknowledge that experiments on real databases are needed to demonstrate the efficiency and effectiveness of the approach. To this end, one can observe that at the end of the query answering process, it is a precise (point or range) query ($\tilde{Q}_t$) that is evaluated over the database. So, one can implement the approach over (traditional) existing database systems. To such existing systems, two modules have to be added (see Figure 2) (i) a query translator that converts the flexible query into a crisp query; (ii) a similarity estimator that computes the similarity measures between attribute values. Experiments to perform allow also for providing an idea about the extra cost resulting from the use of the approach. Besides, only attributes with domains endowed with a metric can be addressed by the approach. It would be interesting to extend it to attributes with non metricized domains (as *color* attribute).

References

1. S. Agrawal, S. Chaudhuri, G. Das, V. Hristidis, A. Gionis, Automated ranking of database query results. CIDR 2003.
2. T. Andreasen, O. Pivert, On the weakening of fuzzy relational queries, in *8th Int. Symp. On Meth. for Intell. Syst.*, Charlotte, USA, 1994, pp. 144-151.
3. P. Bosc, C. Brando, A. Hadjali, H. Jaudoin, O. Pivert, "Semantic proximity between queries and the empty answer problem", In IFSA World Congress, Lisbon, Portugal, 2009.
4. P. Bosc, A. Hadjali, O. Pivert, Incremental Controlled Relaxation of Failing Flexible Queries, *Journal of Intelligent Information Systems*, Vol. 3(3), 2009, pp. 261-283.

5. P. Bosc, A. Hadjali, O. Pivert, Empty versus Overabundant Answers to Flexible Queries, *Fuzzy sets and Systems Journal*, Vol. 159(16), 2008, pp. 1450-1467.
6. P. Bosc and O. Pivert.: SQLf : a relational database language for fuzzy querying. IEEE Transactions on Fuzzy Systems, vol. 3(1),pp. 1–17, 1995.
7. B.B. Chaudhuri and A. Rosenfeld, "A modified Hausdorff distance between fuzzy sets", *Information Sciences*, Vol. 118, pp. 159-171, 1999.
8. V. Cross and T. Sudkamp, "Similarity and Compatibility in Fuzzy Set Theory: Assessment and Applications", Studies in Fuzziness and Soft Computing, No 93, Physica-Verlag, 2002.
9. F. Corella, K.P. Lewison, A brief Overview of Cooperative Answering. Technical Report, http://www.pomcor.com/whitepapers/cooperative_responses.pdf, August 2009.
10. M. De Calmès, D. Dubois, E. Hullermeier, H. Prade, and F. Sedes, F, Flexibility and fuzzy case-based evaluation in querying: An illustration in an experimental setting. *Int. Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, 11(1), 43-66, 2003.
11. D. Dubois and H. Prade, "On distances between fuzzy points and their use for plausible reasoning", In Proc. Int. Conf. on Systems, Man and Cybernetics, 1983, pp. 300-303.
12. S.M. Fakhr Ahmad, M.H. Sadreddini, M. Zolghadri Jahromi, IQPI: An incremental system for answering imprecise queries using approximate dependencies and concept similarities. Journal of Computer Sciences, 34(2), 2007.
13. A. Hadjali, Providing Approximate Answers to Flexible Failing Queries. Technical Report, September 2009.
14. G. Luo, Efficient detection of empty-result queries. VLDB '06, pp. 1015-1025, 2006.
15. X. Meng, ZM. Ma, L. Yan, Providing flexible queries over Web databases. KES'08, pp. 601-606, 2008.
16. C. Mishra, N. Koudas, Interactive query refinement. EDBT'09, pp. 862-873, 2009.
17. U. Nambiar, S. Kambhampati, Answering Imprecise Queries over Autonomous Web Databases. ICDE'06, pp. 45-54, 2006.
18. W.A. Voglozin, G. Rashia, L. Ughetto and N. Mouaddib, Querying the SaintEtiq summaries: Dealing with null answers. Proc. *IEEE Inter. Conf. on Fuzzy Systems*, pp. 585-590, USA, 2005.

Traitement des requêtes co (content only) sur un corpus de documents xml

Samia Berchiche-Fellag[1], Mohand Boughanem[2]

[1] Université Mouloud Mammeri de Tizi-Ouzou, 15000 Tizi-Ouzou, Algérie.
samfellag@yahoo.fr
[2] IRIT - SIG-RI, 118 Route de Narbonne, 31 062 Toulouse Cedex 4. bougha@irit.fr

Résumé. L'interrogation des documents XML peut se faire de deux manières : avec des requêtes portant sur le contenu (CO Content Only) ou avec des requêtes portant sur le contenu et la structure (CAS Content And Structure). Les requêtes CO sont celles qui sont le plus facilement formulée par l'utilisateur mais restent les plus difficilement traitées par le système de recherche d'information. En effet l'utilisateur ne donnant aucune indication au système de recherche sur la taille de l'unité d'information qu'il désire voir retourner, c'est à ce dernier de décider de la granularité de l'information appropriée à la requête formulée. La problématique qui en découle est : Comment retourner l'unité d'information pertinente ? C'est précisément à cette question que nous allons donner des réponses dans le présent article en proposant une méthode de propagation de termes des nœuds feuilles vers la racine du document XML. Pour ce faire nous considérons qu'un document XML est un arbre composé de nœuds et donc l'unité d'information considérée est un sous arbre.

Mots clés : XML, propagation de termes, requêtes CO, unité d'information

KeyWords : XML, word propagation, CO request, information unit

1. Introduction

L'apparition de nouveaux standards comme XML, permet de séparer la structure logique d'un document de son contenu. Un document est ainsi caractérisé par un contenu informationnel (du texte) et des contraintes structurelles (balises). Ce type de documents ne peut être, cependant, exploité efficacement par les techniques classiques de recherche d'information. En effet, ces dernières traitent le document comme un granule d'information indivisible, alors que le format standard XML permet d'envisager des granules d'information plus fins, tant du point de vue de la représentation que de l'accès. Ceci nécessite alors, la création de nouveaux outils et de nouveaux modèles capables de restituer des *unités d'information* (non plus le document) pertinentes à une requête utilisateur.

Différents modèles de RI ont été proposés et adaptés pour le traitement des documents XML, et ont tenté de répondre efficacement aux requêtes utilisateurs en renvoyant les *unités d'information* (pas le document entier) les plus pertinentes. Ces unités dépendent fortement des requêtes utilisateurs qui peuvent être formulées de

deux manières : par de simples mots clés on parle alors de requêtes orientée contenu (CO pour **C**ontent **O**nly), ou avec en plus des contraintes sur la structure on parle alors de requêtes orientées contenu et structure (CAS pour **C**ontent **A**nd **S**tructure).
L'interrogation des documents XML avec des requêtes CAS suppose une connaissance au moins partielle de la structure des documents à considérer puisque les requêtes formulées possèdent des conditions sur la structure des documents qu'on veut voir retourner par le système. Ces conditions permettent d'aider le système à localiser de manière exacte l'unité d'information recherchée. Dans le cas des requêtes CO ceci n'est pas envisageable puisque aucune condition n'est considérée, c'est donc au système de recherche d'information (SRI) de décider de la granularité d'information appropriée à retourner à l'utilisateur. En effet, le SRI doit pouvoir identifier les *unités d'informations* les plus pertinentes à une requête utilisateur.
La pertinence dans le cadre de la recherche d'information structurée est estimée selon deux grandeurs : l'*exhaustivité* et *la spécificité* [1]. La *spécificité* mesure si tout le contenu de l'unité d'information concerne la requête. L'*exhaustivité* mesure si toutes les informations requises dans la requête sont présentes dans l'unité d'information.

Le problème crucial dans la recherche avec des requêtes CO est : *comment identifier les unités d'information pertinentes ?* Sachant que ces unités doivent être de taille appropriée ni trop grande sous peine que l'information recherchée soit noyée au milieu d'autres sujets ni trop petite sous peine de ne pas être informative. Nous nous intéressons dans cet article à cette problématique que nous allons tenter de résoudre en proposant une méthode de propagation de termes avec leurs poids associés des nœuds feuilles vers la racine du document XML afin d'identifier les unités d'information pertinentes recherchées.

2. Etat de l'art

Nous allons présenter brièvement dans cette section quelques travaux se rapportant à la recherche d'information avec des requêtes orientée contenu dans un corpus XML.

Une approche basée sur la correspondance d'arbres *« tree matching »*, l'arbre du document et l'arbre de la requête est proposée par Schlieder, Meuss et Naumann [2], [3]. Cette correspondance traduit la distance entre les arbres. Grabs et schek [4] proposent quand à eux de mesurer l'importance d'un terme dans un élément en fonction de son importance dans les éléments de même type. Dans le but de retourner le fragment de document répondant de manière pertinente à la requête utilisateur, le score de pertinence d'un nœud (élément) est calculé. Pour ce faire, certaines approches, adoptent la méthode de propagation des termes [5], d'autres, la méthode de propagation de pertinence (score) [6] en utilisant une combinaison linéaire des scores des enfants appelées « *maximum-by-category* » et « *summation* ». Sauvagnat [7], propose dans le même ordre d'idées une méthode de propagation de score des nœuds feuilles vers la racine du document dans son modèle XFIRM. Fuhr &al [8][9]utilise une méthode de propagation de poids des nœuds les plus spécifiques dans l'arbre du document en utilisant un facteur d'*augmentation* (multiplication par un facteur donné) basée sur le modèle probabiliste. Il en découle que, quelque soit l'approche adoptée, le score de pertinence d'un nœud dépend fortement du score de ses descendants.

3. Traitement des requêtes

Nous considérons *qu'un document XML est un arbre, composé de nœuds interne, nœuds feuilles et d'attributs* (voir figure suivante).

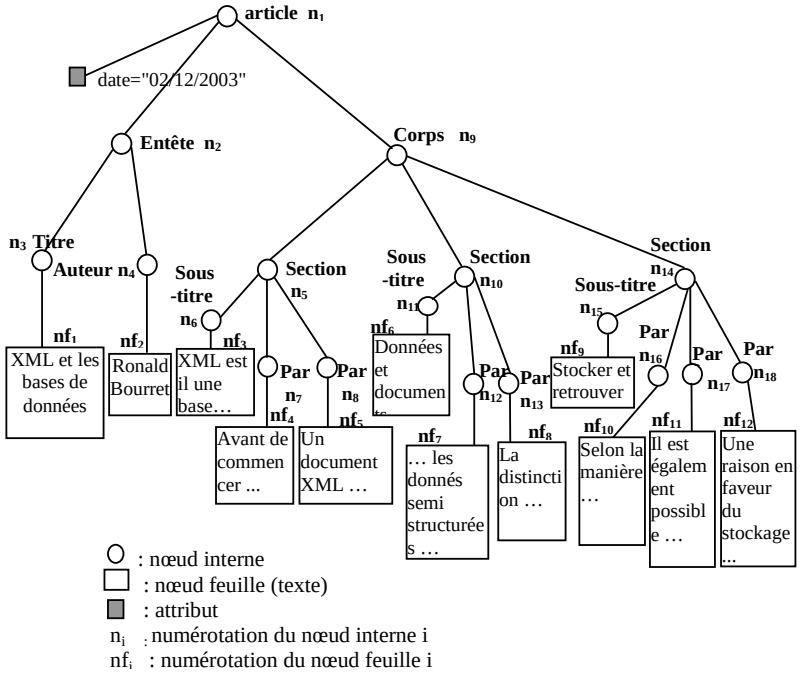


Fig. 1. document article.xml

Le traitement des requêtes que nous effectuons a pour but de renvoyer les unités d'informations les plus *spécifiques* et les plus *exhaustives* à l'utilisateur.
Les requêtes que l'utilisateur peut formuler, sont composées de *simples mots clés sans précision aucune sur la structure*, et c'est au système, de décider de la granularité appropriée de l'information à renvoyer.

Le traitement de requêtes CO que nous proposons de faire, s'effectue comme suit :

- la première phase, consiste à pondérer les termes des nœuds feuilles,
- la seconde phase concerne, l'élagage de l'arbre du document, en ne conservant que les nœuds informatifs,
- la troisième phase, consiste à propager les termes bien distribués dans les nœuds feuilles, vers les nœuds ascendants. Afin de pouvoir identifier les unités d'information, pertinentes et informatives,

- enfin, la dernière phase consiste, à calculer le score de pertinence des unités d'information ainsi identifiées, avec la requête et à présenter les résultats obtenus, par ordre décroissant des scores.

Le calcul du score d'un nœud et la pondération des termes dans les nœuds sont des éléments prépondérants dans la phase d'évaluation de la pertinence d'un nœud vis-à-vis d'une requête. Avant de présenter le processus de propagation que nous avons adopté, nous commençons par décrire ces deux points.

3.1. Calcul du score des nœuds

La requête utilisateur est représentée par des mots clés éventuellement pondérés. On peut avoir la représentation suivante de la requête :
$q = \{(t_1,p_{q1}), \ldots\ldots ,(t_M,p_{qM})\}$ avec t_k un terme de la requête et p_{qk} le poids de t_k dans la requête q et M le nombre de termes dans la requête.
Le score d'un nœud n_i identifié dans l'arbre, est calculé avec une fonction de similarité $RSV(q,n_i)$ du modèle vectoriel (Inner product) comme suit :

$$RSV(q, n_i) = \sum_{k=1}^{M} p_{qk} \times p_{nik} \ . \qquad (0)$$

ou p_{qk} et p_{nik} sont respectivement le poids du terme k dans la requête q et le nœud n_i.
Le poids P_k d'un terme k dans les nœuds feuilles, et dans la requête est calculé grâce à la formule de pondération que nous avons proposée :

$$P_k = tf_k \times Idf_k \times ief_k \ . \qquad (1)$$

Où :
tf_k : term frequency (fréquence d'un terme k dans le nœud feuille ou dans la requête)
Idf_k : Inverted document frequency = fréquence inverse de document = $\log\left(\frac{N}{n}\right)$
Ief_k: Inverted element frequency = fréquence inverse d'éléments
= $\log\left(\frac{Ne}{ne} + \alpha\right)$ avec $0.5 \le \alpha \le 1$ est une adaptation de Idf à la granularité de l'information à traiter qui n'est plus le document mais l'élément.
N : Nombre total de documents dans la collection
n : Nombre de documents contenant le terme k
Ne : Nombre total de nœuds feuilles dans le document
ne : Nombre de nœuds feuilles contenant le terme k dans le document
Les facteurs, tf, idf et ief et leur combinaison peuvent prendre plusieurs formes. Nous avons fait une série d'expérimentation afin d'évaluer et mesurer les performances des différentes combinaisons.

3.2. Propagation des termes

Afin d'identifier la partie du document qui répond le mieux à la requête utilisateur, nous proposons, une méthode de propagation des termes et des poids, en partant des nœuds feuilles jusqu'à la racine du document.

La question qui en découle est : *quels termes propager et comment ?*
Dans ce but, nous introduisons une notion fondamentale, qui est *l'informativité d'un nœud*.
Un nœud est dit informatif s'il est porteur de suffisamment d'informations pour répondre efficacement à une requête utilisateur. S'il aisé de définir l'informativité, encore faut il la mesurer ! L'intuition guidant cette mesure étant la *taille* d'un nœud (c'est-à-dire le nombre de termes qu'il contient). En effet, un nœud qui ne contient *que* les termes de la requête, est s*pécifique* à cette requête.
Il est cependant, *non informatif* car il n'apporte pas l'information requise à l'utilisateur (un nœud *titre* par exemple peut être pertinent pour une requête mais pas informatif). Nous définissons, à cette fin, un *seuil* qui consiste en, le *nombre de termes minimal* qu'un nœud doit avoir pour être considéré comme informatif.

Deux cas dans la propagation des termes, sont à considérer : *le premier*, consiste à traiter les nœuds dont le nombre de termes est inférieur au seuil, et *le second*, consiste à prendre en compte les nœuds dont le nombre de termes est supérieur au seuil. Il est clair que nous n'avons aucun moyen théorique pour déterminer ce seuil. Dans notre cas, nous proposons de le fixer par expérimentation.

A. Cas ou le nombre de termes des nœuds est inférieur à un seuil

L'arbre du document est parcouru en commençant par les nœuds feuilles. Durant le parcours, lorsqu'un nœud visité a un nombre de termes inférieur à un seuil (à définir), ce nœud est supprimé de l'arbre et son contenu remonté vers son nœud parent. Ce procédé se fait de manière récursive jusqu'à atteindre (et éventuellement dépasser) le seuil, ou atteindre le nœud racine du document, ou atteindre un nœud interne, dont le nombre de termes d'au moins un de ses fils est supérieur ou égal au seuil. L'algorithme illustratif est comme suit :

Algorithme de propagation cas A

1. Commencer par les nœuds feuilles
2. Visiter un nœud de l'arbre
3. Si le nombre de termes du nœud est inférieur au seuil alors
3.1 remonter les termes vers le nœud parent avec leurs poids respectifs,
3.2 supprimer le nœud
4. reprendre les étapes à partir de 2 jusqu'à atteindre ou dépasser le seuil, ou atteindre le nœud racine ou atteindre un nœud interne dont le nombre de termes, d'au moins un de ses fils est supérieure ou égal au seuil.

B. Cas ou le nombre de termes des nœuds est supérieur au seuil

L'idée sous-jacente est la suivante : *« des termes bien distribués dans les éléments enfants d'un élément peuvent être représentatifs pour cet élément »*.
Deux cas peuvent se présenter, un nœud peut avoir plusieurs nœuds fils, ou n'en posséder qu'un seul (seul un nœud feuille n'a pas de fils).

1. **Cas ou un nœud possède plusieurs nœuds fils:** de manière intuitive, on peut penser qu'un terme d'un nœud peut être représentatif pour son parent, s'il existe au moins, dans un de ses frères. Cette intuition à elle seule ne suffit pas, car il faudrait tenir compte d'un facteur très important, qui est la pondération des termes dans les nœuds. En effet, un terme peut appartenir à tous les nœuds fils d'un élément, mais si son poids est faible, par rapport à l'ensemble des termes des nœuds fils. Il ne pourra pas être discriminant pour ces nœuds. C'est dans cet ordre d'idée, que nous avons adjoins une autre intuition (en plus de la précédente) qui consiste à ne prendre en considération, que les termes dont le poids moyen au niveau des nœuds fils ou ils apparaissent, est compris entre, le poids moyen des termes des nœuds fils et leurs poids maximal.

2. **Cas ou un nœud possède un seul nœud fils :** L'intuition utilisée dans ce cas, est de considérer qu'un terme d'un nœud fils, ne peut être discriminant pour son parent, que si son poids (du terme), est compris entre le poids moyen des termes du nœud fils et leurs poids maximal.

Le terme vérifiant les intuitions de l'un des cas, 1 ou 2, sera supprimé de son nœud d'origine et remonté vers son nœud parent considéré. Son poids dans le nœud parent, est égal à son poids moyen au niveau de tous les nœuds fils, dans le cas 1, ou à son poids dans le nœud fils considéré, dans le cas 2.
Ces intuitions sont formalisées de la manière suivante :

1. Soit *e* un nœud possédant *plusieurs* nœuds fils *e'*. Soit *t* un terme d'un nœuds fils *e'* de *e*. *P(t,e')* le poids du terme *t* dans le nœud *e'*, calculé avec la formule (1). *t* peut être remonté vers *e*, si *t* existe dans au moins un nœud frère de *e'* et si la moyenne du poids de *t* dans les nœuds fils de *e* ou il apparaît, vérifie la condition suivante :

$$P_{moy} \leq \underset{e' \in enf(e)}{moy}(P(t,e')) \leq P_{max} \qquad (2)$$

Avec

$$P_{moy} = \frac{\sum_{e' \in enf(e)} \sum_{i=1}^{Nte'} p(t_i, e')}{Nt}. \qquad (3)$$

P_{moy} : le poids moyen des termes dans les nœuds *e'* fils de *e*

$$\text{moy}_{e' \in enf(e)}\left(P(t,e')\right) = \frac{\sum_{e' \in enf(e) / t \in e'} P(t,e')}{Ne'} . \quad (4)$$

Nte' : nombre de termes dans le nœud *e'*
Nt : nombre de termes dans tous les nœuds *e'* enfants de *e*
Ne' : nombre de nœuds *e'* contenant le terme *t*
P_{max} : le poids maximum des termes dans tous les nœuds *e'* enfants de *e*
Le terme *t* sera supprimé des nœuds fils *e'*, et remonté vers le nœud père *e*, et son poids dans *e* sera :

$$P(t,e) = \underset{e' \in enf(e)}{\text{moy}}(P(t,e')) . \quad (5)$$

2. Soit *e* un nœud possédant *un seul* nœuds fils *e'*. Soit *t* un terme du nœud *e'*. *P(t,e')* le poids du terme *t* dans le nœud *e'*. *t* peut être remonté vers *e*, s'il vérifie la condition (6) suivante :

$$P_{moy} \leq P(t,e') \leq P_{max} \quad (6)$$

avec

$$P_{moy} = \frac{\sum_{i=1}^{Nte'} P(t_i,e')}{Nte'} \quad (7)$$

P_{moy} : le poids moyen des termes dans *e'*
P_{max} : le poids maximum des termes dans *e'*
Nte' : nombre de termes dans le nœud *e'*

Le terme *t* sera supprimé du nœud fils *e'* et remonté vers le nœud père *e* en conservant son poids, donc :

$$P(t,e) = P(t, e') \quad (8)$$

Indiquons que, durant la remontée d'un terme *t* du nœud fils *e'* vers son parent *e*, celui ci peut s'y trouver déjà. Dans ce cas, le terme *t* sera supprimé du (ou des) nœud(s) fils *e'*, et son poids dans le nœud *e* sera égal, à la moyenne de son poids, dans le(s) nœud(s) fils *e'* et le nœud parent *e*, c'est-à-dire

$$P(t,e) = \frac{P_{(5)/(8)}(t,e) + P_0(t,e)}{2} \quad (9)$$

avec :

$P_0(t,e)$: poids initial du terme *t* dans le nœud *e*
$P_{(5)/(8)}(t,e)$: poids que devait avoir (s'il n'y existait pas) le terme *t* dans *e* calculé avec la formule (5) ou (8) selon le cas considéré.
Le processus de propagation des termes, se déroule de manière récursive des feuilles de l'arbre jusqu'à la racine.

Nous résumons ces différents cas dans l'algorithme suivant :

Algorithme propagation cas B

1. Commencer par les feuilles
2. visiter un nœud
3. lire un terme
4. si le nœud possède des frères alors
   4.1 si le terme existe dans au moins un nœud frère alors vérifier la formule (2)
4. sinon vérifier la formule (6)
5. si la formule (2 ou 6) vraie alors supprimer le terme du (des) nœud(s) fils et le remonter vers son parent
   5.1 si le terme existe dans le nœud parent alors le poids du terme se calcule avec la formule (9)
   1.1 sinon si la formule (2) vraie alors le poids du terme se calcule avec (5) sinon avec (8)
6. reprendre à partir de l'étape 3 jusqu'à lire tous les termes du nœud considéré
7. reprendre à partir de l'étape 1 jusqu'à atteindre le nœud racine du document

A l'issue de ce traitement, le score de similarité des termes de la requête avec les nœuds représentés par ces termes sera calculé, les résultats seront présentés par ordre décroissant des scores. Les sous arbres pertinents et informatifs seront ainsi présentés à l'utilisateur.

Exemple de traitement de requêtes

Afin de bien illustrer la méthode de propagation de termes que nous avons proposée, nous déroulons dans ce qui suit, un exemple de requête sur un exemple de document.
Reprenons le document précédant extrait de l'article de *Ronald Bourret* sur *"XML et les bases de données"*. Soit la requête suivante *"base XML native"*, composée de trois termes *base*, *XML* et *native*. Les nœuds feuilles contenant les termes de la requête sont : nf_1, nf_3, nf_4, nf_5, $\mathbf{nf_7}$, $\mathbf{nf_{10}}$, $\mathbf{nf_{11}}$, $\mathbf{nf_{12}}$, les quatre derniers nœuds cités contiennent *tous* les termes de la requête.

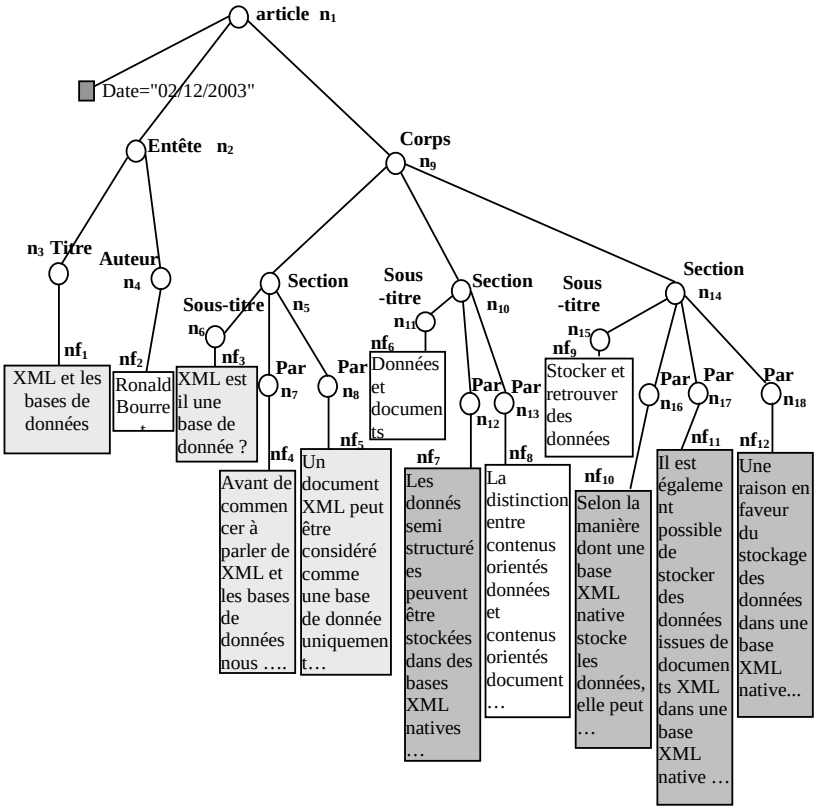


Fig. 2. Représentation en arbre du document article.xml après localisation des nœuds contenant les termes de la requête

Après application de l'algorithme A (suppression des nœuds non informatifs en considérant un seuil de 20 termes) sur le document XML considéré, on obtient un arbre ne contenant que des nœuds informatifs comme l'illustre la figure 3. Nous appliquons sur ce dernier sur l'arbre l'algorithme B et on obtient l'arbre illustré par la figure 4 dans lequel on retrouve :

Les unités d'information représentée par les termes *base*, *XML* et *native* sont :

- le nœud feuille nf_5,
- le sous arbre de racine n_5,
- le sous arbre de racine n_{12},
- le sous arbre de racine n_{14},
- et enfin l'arbre de racine n_1.

Après calcul du score de similarité des sous arbres ainsi localisés avec les termes de la requête, les unités d'informations exhaustives, spécifiques et informatives à retourner par ordre décroissant des scores sont :

- le sous arbre de racine n_{12},
- le sous arbre de racine n_{14} ,
- le nœud feuille nf_5,
- le sous arbre de racine n_5,
- et enfin plus général l'arbre de racine n_1.

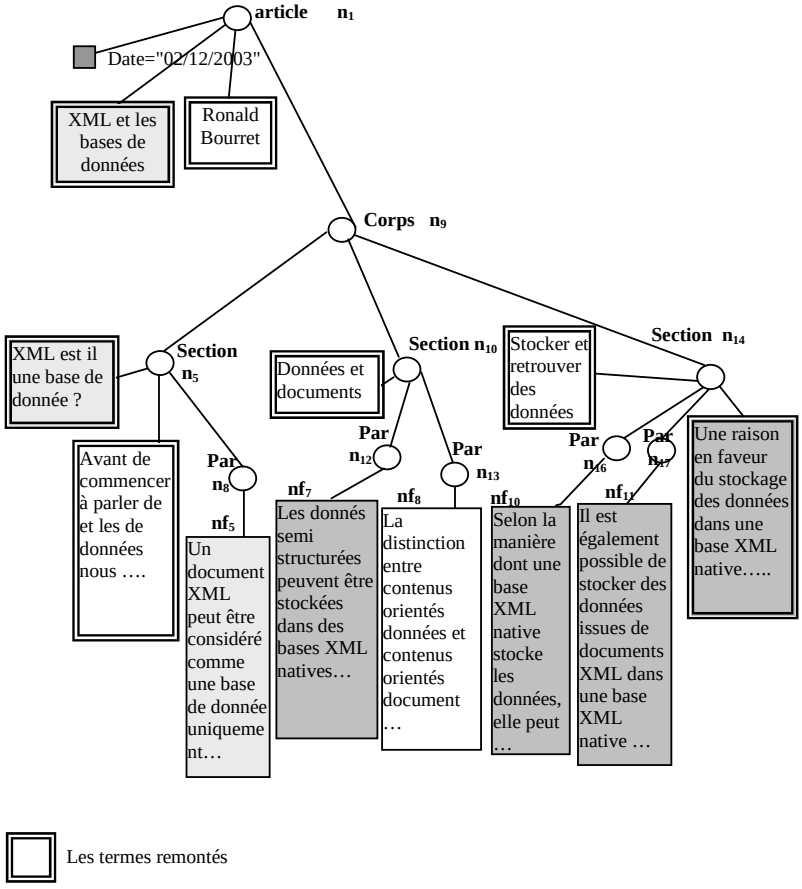


Fig. 3. Le document article.xml après suppression des nœuds non informatifs

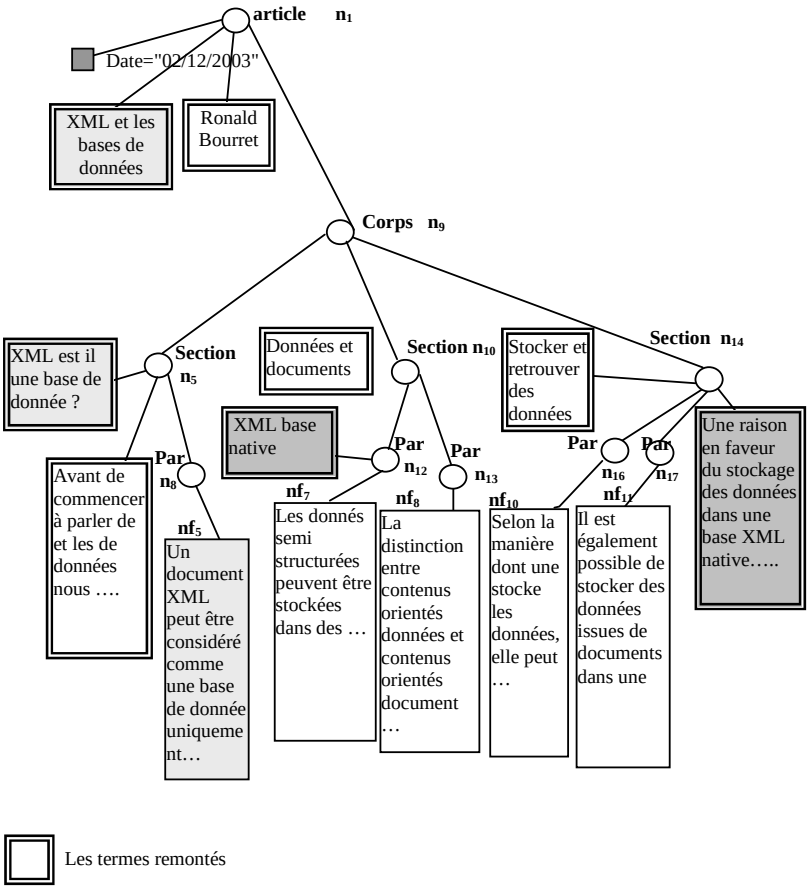


Fig. 4. Le document article.xml après application de l'algorithme B

Il est essentiel de noter que le traitement que nous venons d'effectuer peut se faire (est préférable d'être fait) de manière *statique* indépendamment de la requête. A l'arrivée d'une requête, seuls les scores seront calculés. Précisons aussi, que le traitement statique permet un gain de temps considérable, et un temps de réponse du système à une requête très appréciable, ce qui constitue l'un des atouts d'un SRI.

4. Conclusion

Dans cet article nous avons présenté notre contribution au problème de RI dans les documents semi structurés XML avec des requêtes CO en proposant une méthode de propagation des termes des nœuds feuilles vers la racine du document. En effet, cette méthode a pour but non seulement de retourner les unités d'information les plus *spécifiques* et *exhaustives* à une requête utilisateur mais surtout de renvoyer des unités *informatives* grâce à la contrainte sur le nombre de termes d'un nœud, que nous avons imposé. De plus cette méthode peut être traitée de manière *statique* c'est-à-dire avant la formulation d'une requête, et de ce fait, le traitement des requêtes devient aisé et leur temps d'exécution considérablement réduit.
Les résultats expérimentaux sur cette méthode de propagation sont en cours, néanmoins des tests manuels ont été faits et des résultats probants ont été obtenus.

5. Bibliograhie

1. M. Lalmas.: Dempster-Shafer's theory of evidence applied to structured documents: Modeling uncertainty. In: Proceedings of ACM-SIGIR, pp. 110-118. Philadelphia, 1997.
2. T.Schlieder and H.Meuss.: Querying and ranking XML documents. journal of the American society for information Science and Technology, 53(6) :489-503, 2002.
3. T.Schlieder and F.Naumann.: Approximate tree embedding for querying XML data. In: proceedings of the first annual workshop of INEX, Dagstuhl, Germany, December 2002.
4. T.Grabs and H .J.Scheck.: Flexible information retrieval from XML with Power DB XML. In : proceedings of the first annual workshop of INEX, pages 141-148, December 2002
5. H.cui, J-R.Wen, J-R.Chua.: Hierarchical indexing and flexible element retrieval for structured document. april 2003.
6. Vo Ngoc Anh, Alistair Moffat.: Compression and an IR approach to XML Retrieval. In: INEX 2002 Workshop Proceedings, p. 100-104, Germany, 2002.
7. Karen Sauvagnat. : Modèle flexible pour la recherche d'information dans des corpus de documents semi-structurés. Thèse Doctorat, Université Paul Sabatier de Toulouse, 2005
8. N. Fuhr, K. Grossjohann. : XIRQL, a query language for information retrieval in XML documents. In: proceedings of SIGIR 2001, Toronto Canada 2001.
9. N. Gövert, M. Abolhassanni, N. Fuhr, K. Grossjohann,.: Content-Oriented XML Retrieval with HyreX. In: INEX 2002 Workshop Proceedings, p. 26-32, Germany, 2002.

Optimisation I

Analyse de sensibilité en programmation mathématique

Khadra NACHI
Université d'Es-Sénia, Oran
Faculté des sciences, Département de Mathématiques
BP 1524, El-M'naouer
nachikhadra@yahoo.fr

Résumé : Nous présentons dans cette note quelques principaux résultats de stabilité et de sensibilité dans un problème d'optimisation paramètrée. Nous donnons des propriétés topologiques des solutions primales et duales ainsi que de la fonction de performance. Les questions de différentiablité sont aussi examinées.

1 Introduction

Les questions de sensibilité en programmation mathématique non linéaire pour des problèmes paramétrés ont retenu l'attention des mathématiciens depuis de nombreuses années. Des résultats très divers et complèmentaires ont été établis. Plus particulièrement, le résultat fondamental de ces travaux est l'obtention d'une représentation lipschitzienne et de sa différentiabilité directionnelle (par rapport aux paramétres) des solutions optimales du problème de minimisation d'une fonction paramétrée soumises à des contraintes elles-mêmes paramétrées.

Notons que le problème standart en dimension finie (avec un nombre fini de contraintes égalités-inégalités) a été étudié par Fiacco [12], Cornet-Laroque [8], Bergounioux [6], Bertsekas [7] qui ont démontré la différentiabilité des points stationnaires (i.e. solutions du système de Kuhn-Tucker) et des multiplicateurs de Lagranges associés. L'utilisation du théorème des fonctions implicites est à la base de ces résultats. Pour cela, des hypothèses de "complémentarité stricte" et "d'indépendance linéaire" (des gradients des contraintes actives ou saturées) sont supposés satisfaites ainsi qu'une condition du second ordre.

Les propriétés de différentiabilité de la valeur optimale ont été étudiées par de nombreux auteurs dont principalement Auslender-Cominetti [4], Rockafellar ([20], [21], [22]), Shapiro ([23], [24]).

Dans le cadre général d'un problème de programmation mathématique posé en dimension infinie, l'étude de la sensibilité a connu un développement très important en surmonttant plusieurs difficultées liées notamment à des propriétés de compacité et aussi à la forme même des contraintes. Sur cet aspect, on pourra consulter de nombreux travaux, citons, par exemple, Alt ([1], [2], [3]), Barbet [5], Dontchev-Hager ([9]), Dontchev-Rockafellar [10], Janin [13], Penot ([18], [19]),...

Nous présentons dans ce travail quelques résultats de sensibilté établis dans ([17]) où la condition du second ordre et la condition de régularité du domaine jouent un rôle crucial. Plus précisément, nous montrons l'existence d'une représentation continûment différentiable (par rapport aux paramètres) des solutions optimales (primales et duales) d'un problème d'optimisation paramétré. Une estimation des premières variations des solutions et de la fonction de performance (fonction valeur) est aussi donnée.

2 Existence d'un champ optimal C^1

Dans ce chapitre, on se place dans le cadre fonctionnel suivant : Soient X et E des espaces de Banach; soit Λ espace vectoriel normé. Et soient les fonctions $f : X \times \Lambda \to \mathbb{R}$, $g : X \times \Lambda \to E$ et $h := (h_i)_{i=1,...s} : X \times \Lambda \to \mathbb{R}^s$.

On considère, pour tout paramètre $\lambda \in \Lambda$, le problème de programmation mathématique qui consiste à trouver $x_\lambda \in X$ satisfaisant :

$$g(x_\lambda, \lambda) = 0,\ h(x_\lambda, \lambda) \leq 0 \text{ et } f(x_\lambda, \lambda) = \min f(x, \lambda).$$

Le but général est d'étudier la dépendance du paramètre λ des solutions éventuelles du problème $P(\lambda)$ formulé ci-dessous lorsque λ varie au voisinage d'une valeur λ_0 donnée;

$$P(\lambda) \qquad \left\{ \inf_{x \in D_\lambda} f(x, \lambda) \right.$$

où $D_\lambda := \{x \in X : g(x, \lambda) = 0 \text{ et } h_i(x, \lambda) \leq 0\ (i = 1, ...s)\}$.

L'objectif est aussi d'obtenir des résultats de sensibilité, i.e. de différentiabilité (resp. différentiabilité directionnelle) par rapport au paramètre de la solution optimale et de son multiplicateur de Lagrange associé.

Introduisons le lagrangien classique associé à $P(\lambda)$:

$L : X \times E^* \times \mathbb{R}^s \times \Lambda \to \mathbb{R}$ tel que pour tout $(x, p, q, \lambda) \in X \times E^* \times \mathbb{R}^s \times \Lambda$

$$L(x, p, q, \lambda) := f(x, \lambda) + \langle p, g(x, \lambda) \rangle_{E,E^*} + (q \mid h(x, \lambda))_{\mathbb{R}^s},$$

où $\langle \cdot, \cdot \rangle_{E,E^*}$ est le crochet de dualité entre E et E^*, $(\cdot \mid \cdot)_{\mathbb{R}^s}$ est le produit scalaire dans $\mathbb{R}^s$.

Notre étude a été faite sous les hypothèses générales suivantes :

$(\mathcal{H}_1)$: [Existence d'une solution]

Le problème $P(0)$ admet une solution locale en $\overline{x}$,

$(\mathcal{H}_2)$: [Régularité des données]

Les fonctions f, g, $(h_i)_{i=1,...s}$ sont de classe C^2 au voisinage de $(\overline{x}, 0)$,

$(\mathcal{H}_3)$: [Qualification des contraintes]

$\text{Im} \begin{bmatrix} g'_x(\overline{x}, 0) \\ \overline{h}'_x(\overline{x}, 0) \end{bmatrix} = E \times \mathbb{R}^m$ où $\overline{h} := (h_i)_{i=1,...m}$ avec $m := card I(\overline{x})$ et

$I(\overline{x}) := \{i \in \{1, ...s\} : h_i(\overline{x}, 0) = 0\}$: Ensemble des indices des contraintes actives (pour simplification d'écriture, on omet le dépendance en λ).

$(\mathcal{H}_4)$: [Condition du second ordre]

$\exists (\overline{p}, \overline{q}) \in K.T(\overline{x})$ tel que $L''_x(\overline{x}, \overline{p}, \overline{q}, 0)$ est $\alpha-$coercif sur $Ker \begin{bmatrix} g'_x(\overline{x}, 0) \\ \overline{h}'_x(\overline{x}, 0) \end{bmatrix}$ i.e.

$$\langle v, L''_x(\overline{x}, \overline{p}, \overline{q}, 0)v \rangle_{X,X^*} \geq \alpha \|v\|^2 \ \forall v \in Ker \begin{bmatrix} g'_x(\overline{x}, 0) \\ \overline{h}'_x(\overline{x}, 0) \end{bmatrix}.$$

$(\mathcal{H}_5)$: [Stricte-complémentarité]

$\overline{q_i} \gneqq 0 \qquad \forall i \in I(\overline{x})$.

Notons que les hypothèses $(\mathcal{H}_1)$, $(\mathcal{H}_2)$ et $(\mathcal{H}_3)$ (Karush-Kuhn-Tucker [14]) assurent l'existence d'un multiplicateur de Lagrange (ou d'un état adjoint) $(\overline{p},\overline{q}) \in E^* \times \mathbb{R}^s_+$ tel que le système suivant soit satisfait :

$$KKT(\overline{x}) \qquad \left\{ \begin{array}{lcll} L'_x(\overline{x},\overline{p},\overline{q},0) & = & 0, & \\ g(\overline{x},0) & = & 0, & \\ (\overline{q} \mid h(\overline{x},0))_{\mathbb{R}^s} & = & 0,\ h(\overline{x},0) \leq 0,\ \overline{q} \geq 0. & \end{array} \right.$$

L'hypothèse $(\mathcal{H}_3)$ permet aussi de montrer que l'état adjoint est unique. On reconnait aussi les conditions suffisantes du second ordre de minimalité locale (le système ci-dessus et l'hypothèse $(\mathcal{H}_4)$; voir Maurer-Zowe [15]) : $\overline{x}$ est assurément minimum local strict de $f(\cdot,0)$ sur D_0 ou une solution locale stricte de $P(0)$ dans le sens suivant :

Un point $\overline{x}$ est une solution locale stricte d'un problème d'optimisation (P) de type $\inf\limits_{x\in D} f(x)$ si

$$\exists V \in \mathcal{V}(\overline{x}), \forall x \in V \cap D\ (x \neq \overline{x}) : f(\overline{x}) < f(x).$$

Enfin, grâce à l'hypothèse $(\mathcal{H}_5)$, le système $KKT(\overline{x})$ ci-dessus se réduit à un système d'égalités seulement.

Sous les hypothèses précédentes, on obtient le résultat suivant :

Théorème 1 *il existe $W \in \mathcal{V}(0)$ dans Λ, $U \times V \in \mathcal{V}(\overline{x},\overline{p},\overline{q})$ dans $X \times E^* \times \mathbb{R}^s$ et une fonction $z(\cdot) := (x(\cdot),p(\cdot),q(\cdot)) : W \to U \times V$ de classe C^1 tels que :*
a) $z(0) = (\overline{x},\overline{p},\overline{q})$.
b) $\forall \lambda \in W$, $(x(\lambda),p(\lambda),q(\lambda))$ satisfait le système de Karush-Kuhn-Tucker.

Démonstration : La preuve fera usage du théorème des fonctions implicites, nous allons donc exhiber une application satisfaisant les conditions d'application.
Posons $Z := X \times E^* \times \mathbb{R}^s$, $\overline{z} := (\overline{x},\overline{p},\overline{q})$.

Considérons l'application $\Psi : Z \times \Lambda \to X^* \times E \times \mathbb{R}^s$ telle que

$$\Psi(z,\lambda) := (L'_x(x,p,q,\lambda), g(x,\lambda), q_1 h_1(x,\lambda), ..., q_s h_s(x,\lambda)).$$

Comme $(\overline{p},\overline{q}) \in K.T(\overline{x})$, on a $\Psi(\overline{z},0) = 0$, de plus, d'après la régularité des données, il est clair que Ψ est de classe C^1 au voisinage de $(\overline{z},0)$ et que :

$$\Psi'_z(\overline{z},0) = \left[\begin{array}{ccc} L''_x(\overline{z},0) & g'^*_x(\overline{x},0) & h'^*_x(\overline{x},0) \\ g'_x(\overline{x},0) & 0 & 0 \\ \overline{q}h'_x(\overline{x},0) & 0 & diag(h_i(\overline{x},0))_{i=1,..s} \end{array} \right].$$

D'autre part, $\Psi'_z(\overline{z},0)$ est inversible :
Grâce aux hypothèses $(\mathcal{H}_3)$, $(\mathcal{H}_4)$ et $(\mathcal{H}_5)$, on montre facilement l'injection de $\Psi'_z(\overline{z},0)$. En effet, soit $(v,p,q) \in Z$ tel que $\Psi'_z(\overline{z},0)(v,p,q) = 0$.

Notons par $I := I(\overline{x}) = \{i \in \{1,...s\} : h_i(\overline{x},0) = 0, \overline{q}_i > 0\}$,
$J := \{i \in \{1,...s\} : h_i(\overline{x},0) < 0\}$ et $K := \{i \in \{1,...s\} : h_i(\overline{x},0) = 0, \overline{q}_i = 0\}$. D'où, d'après $(\mathcal{H}_5)$, l'ensemble K est vide et $I \cup J = \{1,...s\}$. On obtient donc le système d'égalités suivant :

$$\left\{ \begin{array}{lcll} L''_x(\overline{z},0)v + g'^*_x(\overline{x},0)p + h'^*_x(\overline{x},0)q & = & 0, & \\ g'_x(\overline{x},0)v & = & 0, & \\ \overline{q}_i h'_{ix}(\overline{x},0)v & = & 0, & i \in I \\ h_i(\overline{x},0)q_i & = & 0, & i \in J. \end{array} \right.$$

On en déduit que $v \in Ker \begin{bmatrix} g'_x(\overline{x},0) \\ \overline{h}'_x(\overline{x},0) \end{bmatrix}$ et que $q = (q_I, 0_J)$.

La première équation donne alors $\langle v, L''_x(\overline{z},0)v\rangle_{X,X^*} = 0$ et l'hypothèse de second ordre permet de conclure que $v = 0$. Et donc

$$\begin{pmatrix} p \\ q_I \end{pmatrix} \in Ker \begin{pmatrix} g'^*_x(\overline{x},0) & \bar{h}'^*_x(\overline{x},0) \end{pmatrix} = \{0\}$$

par l'hypothèse $(\mathcal{H}_3)$. On conclut l'injection de l'opérateur $\Psi'_z(\overline{z},0)$.

Soit maintenant $(x^*, y, \alpha) \in X^* \times E \times \mathbb{R}^s$, l'étude de la surjection revient à la résolution du système suivant :

$$\begin{cases} L''_x(\overline{z},0)v + g'^*_x(\overline{x},0)p + h'^*_x(\overline{x},0)q &= x^*, \\ g'_x(\overline{x},0)v &= y, \\ \overline{q}_i h'_{ix}(\overline{x},0)v &= \alpha_i, \quad i \in I \\ h_j(\overline{x},0)q_j &= \alpha_j, \quad j \in J. \end{cases}$$

Comme $h_j(\overline{x},0) < 0 \ \forall\ j \in J$, de la dernière équation on a : $q_j = \dfrac{\alpha_j}{h_j(\overline{x},0)}$.

Considérons alors le système :

$$(S) \qquad \begin{cases} L''_x(\overline{z},0)v + g'^*_x(\overline{x},0)p + \bar{h}'^*_x(\overline{x},0)q_I &= x^* - \sum\limits_{j\in J} \bar{h}'^*_{jx}(\overline{x},0)q_j, \\ g'_x(\overline{x},0)v &= y, \\ \bar{h}'_x(\overline{x},0)v &= \beta_I := (\frac{\alpha_i}{\overline{q}_i})_{i\in I} \end{cases}$$

D'après la surjection de l'opérateur $B_0 := \begin{bmatrix} g'_x(\overline{x},0) \\ \overline{h}'_x(\overline{x},0) \end{bmatrix} : X \to E \times \mathbb{R}^m$, il existe un vecteur $w \in X$ satisfaisant les équations suivantes :

$$\begin{cases} g'_x(\overline{x},0)w &= y, \\ \bar{h}'_x(\overline{x},0)w &= \beta_I. \end{cases}$$

Donc

$$\forall v \in KerB_0 : \begin{cases} g'_x(\overline{x},0)(v+w) &= y, \\ \bar{h}'_x(\overline{x},0)(v+w) &= \beta_I. \end{cases}$$

Ainsi la résolution du système (S) revient à résoudre dans $KerB_0 \times E^* \times \mathbb{R}^m$, l'équation suivante :

$$L''_x(\overline{z},0)v + g'^*_x(\overline{x},0)p + \bar{h}'^*_x(\overline{x},0)q_I = u^*$$

où $u^* := x^* - \sum\limits_{j\in J} \bar{h}'^*_{jx}(\overline{x},0)q_j - L''_x(\overline{z},0)w \in X^*$.

D'autre part, grâce à l'hypothèse de coercivité, on peut définir sur $KerB_0$ un produit scalaire en posant :

$$(u \mid v)_{KerB_0} := \langle u, L''_x(\overline{z},0)v\rangle_{X,X^*}, \forall u, v \in KerB_0.$$

De plus, il est facile de montrer que ce produit scalaire induit une norme hilbertienne sur $KerB_0$ notée $\|\cdot\|_{KerB_0}$ (voir [16]).

Comme $u^*_{|KerB_0}$ est une forme linéaire continue sur $KerB_0$ car pour tout $v \in KerB_0$ on a

$$|u^*(v)| \leq \|u^*\|_{X^*} \|v\|_X ,$$

et d'après $(\mathcal{H}_4)$,

$$|u^*(v)| \leq \frac{1}{\sqrt{\alpha}} \|u^*\|_{X^*} \|v\|_{KerB_0} .$$

Ainsi, le théorème de Riez assure que :

$$\exists! \zeta \in KerB_0, \forall v \in KerB_0 : u^*(v) = (v \mid \zeta)_{KerB_0} .$$

D'où

$$\forall v \in KerB_0 : \langle v, u^* - L''_x(\overline{z}, 0)\zeta \rangle_{X,X^*} = 0.$$

On en déduit que $u^* - L''_x(\overline{z}, 0)\zeta \in \operatorname{Im} B_0^*$, i.e. :

$$\exists (p, q_I) \in E^* \times \mathbb{R}^m : u^* - L''_x(\overline{z}, 0)\zeta = g'^*_x(\overline{x}, 0)p + \bar{h}'^*_x(\overline{x}, 0)q_I.$$

Ce qui permet de conclure la surjection de $\Psi'_z(\overline{z}, 0)$.
$\Psi'_z(\overline{z}, 0)$ est donc un homéomorphisme d'après le théorème de Banach.

Le théorème des fonctions implicites affirme donc que l'équation $\Psi(z, \lambda) = 0$ admet une solution au voisinage de $(\overline{z}, 0)$. Plus précisément, il existe des voisinages $W \in \mathcal{V}(0)$, $U \times V \in \mathcal{V}(\overline{z})$ et il existe une fonction $z(\cdot) := (x(\cdot), p(\cdot), q(\cdot)) : W \to U \times V$ de classe C^1 tels que :

a) $x(0) = \overline{x}$, $p(0) = \overline{p}$, $q(0) = \overline{q}$.

b) $\Psi(z(\lambda), \lambda) = 0 \quad \forall \lambda \in W$, i.e. :

$$\left\{ \begin{array}{lcll} L'_x(z(\lambda), \lambda) & = & 0, & \\ g(x(\lambda), \lambda) & = & 0, & \\ q_i(\lambda) h_i(x(\lambda), \lambda) & = & 0, & i = 1, ...s. \end{array} \right. \tag{1}$$

De plus, par la continuité en $\lambda = 0$, on a $q_i(\lambda) \geq 0$ et $h_i(x(\lambda), \lambda) \leq 0$ pour tout λ très petit et $i = 1, ...s$.

Par conséquent, $(x(\lambda), p(\lambda), q(\lambda))$ satisfait le système de Kuhn-Tucker au voisinage de $\lambda = 0$. □

Il existe donc un champ de Kuhn-Tucker continûment différentiable au voisinage du paramétre $\lambda = 0$. Le résultat suivant montre que ce champ est, en fait, optimal :

Théorème 2 *Sous les mêmes hypothèses, il existe un voisinage de* 0, *noté* W, *tel que pour tout* $\lambda \in W$, $x(\lambda)$ *est une solution locale stricte du problème* $P(\lambda)$ *et* $(p(\lambda), q(\lambda))$ *est l'unique état adjoint associé.*

Démonstration : Il s'agit de vérifier les conditions d'optimalité du second ordre. Pour cela, il suffit de montrer que :

(a) il existe $\beta > 0$ tel que

$$\langle v, L''_x(z(\lambda), \lambda)v \rangle_{X,X^*} \geq \beta \|v\|^2 \ \ \forall v \in KerB_\lambda, \forall \lambda \in W.$$

et

(b) la condition de qualification des contraintes reste satisfaite au voisinage de $\lambda = 0$, i.e. que l'opérateur $B_\lambda := \begin{bmatrix} g'_x(x(\lambda), \lambda) \\ \overline{h}'_x(x(\lambda), \lambda) \end{bmatrix}$ est surjectif pour tout λ dans un certain voisinage W de 0.

Pour le point (a) : Comme l'application $\lambda \to A_\lambda := L''_x(z(\lambda), \lambda)$ est continue en 0 alors , en utilisant l'hypothèse $(\mathcal{H}_4)$, pour tout λ dans un certain voisinage W de 0, on a $(\alpha' := \frac{\alpha}{2})$

$$\langle v, A_\lambda v\rangle_{X,X^*} \geq \alpha' \|v\|^2 \ \forall v \in KerB_0. \tag{2}$$

Soit, maintenant, $v \in KerB_\lambda$. L'opérateur, $B_0 : X \to E \times \mathbb{R}^m$ étant linéaire continu surjectif alors , d'après le principe de l'application ouverte la multi-application B_0^{-1} est lipschitzienne, i.e. il existe une constante $l > 0$ telle que et pour tout $w, w' \in E \times \mathbb{R}^m$:

$$B_0^{-1}w \subset B_0^{-1}w' + l\,\|w - w'\|\,\mathcal{B}_X, \tag{3}$$

où $\mathcal{B}_X$ est la boule unité dans X.

En particulier, pour $w := B_0 v$ et $w' := B_\lambda v$, alors comme $v \in B_0^{-1}(B_0 v)$ il existe $\bar{v} \in X$ tel que $B_0\bar{v} = B_\lambda v = 0$ et tel que

$$\|\bar{v} - v\| \leq l\,\|B_0 - B_\lambda\|\,\|v\|\,,$$

l'inégalité triangulaire donne alors un encadrement de $\|\bar{v}\|$; en effet :

$$(1 - l\,\|B_0 - B_\lambda\|)\,\|v\| \leq \|\bar{v}\| \leq (1 + l\,\|B_0 - B_\lambda\|)\,\|v\|\,.$$

Posons $w := \bar{v} - v$ donc $v = \bar{v} - w$ avec $\bar{v} \in KerB_0$. D'après (2), il vient

$$\begin{aligned}\langle v, A_\lambda v\rangle_{X,X^*} &\geq \alpha' \|\bar{v}\|^2 - 2\,\langle w, A_\lambda \bar{v}\rangle_{X,X^*} + \langle w, A_\lambda w\rangle_{X,X^*} \\ &\geq \alpha' \|\bar{v}\|^2 - 2\,\|A_\lambda\|\,\|\bar{v}\|\,\|w\| - \|A_\lambda\|\,\|w\|^2\end{aligned}$$

Utilisant l'inégalité bien connue :

$$2ab \leq \delta a^2 + \frac{1}{\delta} b^2, \ \forall \delta \in \mathbb{R}_+^*,$$

pour $\delta_0 := \dfrac{\alpha'}{2}$ alors

$$\langle v, A_\lambda v\rangle_{X,X^*} \geq (\alpha' - \delta_0)\,\|\bar{v}\|^2 - (1 + \frac{1}{\delta_0})\,\|A_\lambda\|\,\|w\|^2 \tag{4}$$

Soit, maintenant, $\theta \in (0, \frac{1}{2})$, en utilisant la continuité en 0, il existe W un voisinage de 0 et il existe une constante $c \in \mathbb{R}_+^*$ tels que

$$\|A_\lambda\| \leq c \text{ et } l\,\|B_0 - B_\lambda\| \leq \min(\theta, \frac{\theta}{c}(1 + \frac{1}{\delta_0})^{-\frac{1}{2}}(\alpha' - \delta_0)^{\frac{1}{2}}).$$

Donc l'inégalité (4) donne :

$$\begin{aligned}\langle v, A_\lambda v\rangle_{X,X^*} &\geq ((\alpha' - \delta_0)(1 - l\,\|B_0 - B_\lambda\|)^2 \\ &\quad - (1 + \frac{1}{\delta_0})\,\|A_\lambda\|^2\, l^2\,\|B_0 - B_\lambda\|^2)\,\|v\|^2 \\ &\geq (\alpha' - \delta_0)((1 - \theta)^2 - \theta^2)\,\|v\|^2 \\ &\geq \frac{\alpha'}{2}(1 - 2\theta)\,\|v\|^2\,.\end{aligned}$$

Par conséquent, il existe une constante $\beta := \dfrac{\alpha'}{2}(1 - 2\theta) > 0$ telle que pour tout $\lambda \in W$, A_λ est β-coercif sur $KerB_\lambda$. Ce qui achève la preuve du point **(a)**.

En conclusion, $x(\lambda)$ est solution locale stricte de $P(\lambda)$ ($\lambda \in W$).

Pour le point (b) : Toujours par la continuité de l'application $\lambda \to B_\lambda \in \mathcal{L}(X,E \times \mathbb{R}^m)$ en $\lambda = 0$, pour $\varepsilon_0 > 0$ fixé, il existe un voisinage de 0, noté W tel que pour tout $\lambda \in W$:

$$\exists \varepsilon_\lambda \in \mathcal{L}(X,E \times \mathbb{R}^m) : \varepsilon_\lambda := B_\lambda - B_0 \text{ avec } \|\varepsilon_\lambda\|_{\mathcal{L}(X,E\times\mathbb{R}^m)} \leq \varepsilon_0. \tag{5}$$

D'autre part, B_0 est surjectif, pour tout $x \in X$, il existe $x_\lambda \in X$ satisfaisant $\varepsilon_\lambda(x) = B_0 x_\lambda$, *i.e.* $B_\lambda x = B_0(x + x_\lambda)$ et tel que $\|B_0 x_\lambda\| \leq \varepsilon_0 \|x\|$. Ainsi, pour tout $\lambda \in W$, il vient

$$\forall x \in X, \exists x_\lambda \in X : B_\lambda(x - x_\lambda) = B_0(x). \tag{6}$$

Notre but est de montrer que, pour tout $\lambda \in W$, B_λ est surjectif. Soit alors $y \in E \times \mathbb{R}^m$, il existe donc un vecteur $x \in X$ tel que $B_0 x = y$. D'après la propriété (6), il existe $x_\lambda \in X : B_\lambda(x - x_\lambda) = B_0(x) = y$. D'où la surjection de l'opérateur B_λ pour λ voisin de 0.

En conséquence, $(p(\lambda), q(\lambda))$ est l'unique multiplicateur de Lagrange (ou état adjoint) associé à la solution $x(\lambda)$ pour tout paramétre λ dans un certain voisinage W de 0. D'où le théorème. □

3 Estimation des premières variations

Le champ optimal étant continûment différentiable au voisinage de la perturbation nulle et satisfait, pour tout $\lambda \in W$, les égalités dans (1), nous pouvons alors donner l'expression de sa dérivée en $\lambda = 0$. Nous déduisons aussi une approximation d'ordre 1 de la solution et de son multiplicateur au voisinage de $\lambda = 0$. En effet, on a :

Corollaire 1 *Sous les mêmes hypothèses, on a*

(*a*) $\begin{bmatrix} x'_\lambda(0) \\ p'_\lambda(0) \\ q'_{I\lambda}(0) \end{bmatrix} = -M^{-1}N : \Lambda \to X \times E^* \times R^m$ *et* $q'_{J\lambda}(0) = 0.$

(*b*) $\begin{bmatrix} x(\lambda) \\ p(\lambda) \\ q_I(\lambda) \end{bmatrix} = \begin{bmatrix} \bar{x} \\ \bar{p} \\ \bar{q}_I \end{bmatrix} - M^{-1}N(\lambda) + o(\lambda).$

(*c*) *la fonction valeur* $v(\cdot) : W \to \mathbb{R}$ *telle que* $v(\lambda) := f(x(\lambda), \lambda)$ *est de classe* C^1 *et* $v'_\lambda(0) = L'_\lambda(\overline{z}, 0)$.

Avec :

$$M := \begin{bmatrix} L''_x(\overline{z},0) & g'^*_x(\overline{x},0) & \bar{h}'^*_x(\overline{x},0) \\ g'_x(\overline{x},0) & 0 & 0 \\ \bar{h}'_x(\overline{x},0) & 0 & 0 \end{bmatrix} \text{ et } N := \begin{bmatrix} L''_{\lambda,x}(\overline{z},0) \\ g'_\lambda(\overline{x},0) \\ \bar{h}'_\lambda(\overline{x},0) \end{bmatrix}.$$

Démonstration : Comme pour tout $\lambda \in W$, on a le système d'égalités suivant :

$$\begin{cases} L'_x(x(\lambda), p(\lambda), q(\lambda), \lambda) & = & 0 & \\ g(x(\lambda), \lambda) & = & 0 & \\ h_i(x(\lambda), \lambda) & = & 0 & i \in I \\ q_j(\lambda) & = & 0 & j \in J \end{cases}$$

Par différentiation par rapport au paramétre, on obtient :

$$\left\{\begin{array}{lll} L''_x(\overline{z},0)\cdot x'_\lambda(0) + L''_{p,x}(\overline{z},0)\cdot p'_\lambda(0) + L''_{q,x}(\overline{z},0)\cdot q'_\lambda(0) &=& -L''_{\lambda,x}(\overline{z},0) \\ g'_x(\overline{x},0)\cdot x'_\lambda(0) &=& -g'_\lambda(\overline{x},0) \\ h'_{ix}(\overline{x},0)\cdot x'_\lambda(0) &=& -h'_{i\lambda}(\overline{x},0) \\ q'_{j\lambda}(0) &=& 0, \end{array}\right.$$

donc

$$\left[\begin{array}{c} x'_\lambda(0) \\ p'(0) \\ q'_I(0) \end{array}\right] = -\left[\begin{array}{ccc} L''_x(\overline{z},0) & g'^*_x(\overline{x},0) & \bar{h}'^*_x(\overline{x},0) \\ g'_x(\overline{x},0) & 0 & 0 \\ \bar{h}'_x(\overline{x},0) & 0 & 0 \end{array}\right]^{-1} \left[\begin{array}{c} L''_{\lambda,x}(\overline{z},0) \\ g'_\lambda(\overline{x},0) \\ \bar{h}'_\lambda(\overline{x},0) \end{array}\right].$$

Ce qui permet d'obtenir les expressions données dans (a) et de déduire (b).
D'autre part, la fonction valeur $v(\cdot) : W \to \mathbb{R}$ telle que $v(\lambda) := f(x(\lambda),\lambda)$ est de classe C^1 et on a $v(\lambda) = L(x(\lambda),p(\lambda),q(\lambda),\lambda)$. D'où $v'_\lambda(0) = L'_\lambda(\overline{z},0)$ car :

$$\left\{\begin{array}{lllll} L'_x(\overline{z},0) &=& 0 & & \\ L'_p(\overline{z},0) &=& g(\overline{x},0) &=& 0 \\ L'_{Iq}(\overline{z},0) &=& \overline{h}(\overline{x},0) &=& 0 \\ L'_{Jq}(\overline{z},0) &=& -\frac{\overline{q}_J}{c} &=& 0. \end{array}\right.$$

□

References

[1] Alt, W. (1989) Stability of solutions for a class of nonlinear cone constrained optimization problems, part 1 : basic theory, Numerical Functional Analysis and Optimization 10, 1053-1064.

[2] Alt, W. (1989) Stability of solutions for a class of nonlinear cone constrained optimization problems, part 2 : application to parameter estimation, Numerical Functional Analysis and Optimization 10, 1065-1076.

[3] Alt, W. (1983) Lipschitzian perturbations of infinite optimization problems, Mathematical Programming with Data Pertubations, Ed. A. V. Fiacco, 7-21.

[4] Auslender, A. and Cominetti, R. (1990) First and second order sensitivity analysis of nonlinear programs under directional constraint qualification conditions, Optimization 21, 3, 351-363.

[5] Barbet, L. (1992) Etude de sensibilité différentielle dans un problème d'optimisation paramétré avec contraintes en dimension infinie. Thèse de Doctorat. Université de Poitiers.

[6] Bergounioux, M. (1985) Analyse de sensibilité d'un problème paramétré en optimisation. Etude globale et locale des variations d'une solutions. Thèse de Doctorat du $3^{ième}$ cycle. Université des sciences et techniques de Lille.

[7] Bertsekas, D.P. Constrained optimization and lagrange multiplier methods. Computer science and applied mathematics. A serie of monographs and textbooks. Ed : Werner Rheinboldt. University of Pittsburgh.

[8] Cornet, B. and Laroque, G. (1987) Lipschitz properties of solutions. Mathematical Programming Journal of optimazation Theory and Applications. Vol 53, N° 3, June.

[9] Dontchev, A.L. and Hager, W.W (1998) Lipshitzian stability for state contrained nonlinear optimal control. SIAM J. Control Opt. 36 (2), 698-718.

[10] Dontchev, A.L. and Rockafellar, R.T. (1997) Characterizations of Lipshitzian stability in nonlinear programming. In: A. Fiacco, ed., Mathematical programming with data perturbations. 17th symposium, George Washington University, Washington, DC, USA, May 1995. New York, NY: Marcel Dekker. Lect. Notes Pure Appl. Math. 195, 65-82.

[11] Fiacco, A.V. (1976) Sensitivity analysis for nonlinear programming using penalty methods. Mathematical Programming 10, 287-311.

[12] Fiacco, A.V. Introduction to sensitivity and stability analysis in nonlinear programming. Academic Press (1983).

[13] Janin, R. (1984) Directional derivative of the marginal function in nonlinear programming. Mathematical Programming 21, 110-126.

[14] Kuhn, H.W. and Tucker, A.W. (1951) Nonlinear programming in proceedings of the second Barkeley symposium on mathematical statistics and probability. J. Neyman. university of California. Press. Barkeley, 481-492.

[15] Maurer, H. and Zowe, J. (1979) First and second-order necessary and sufficient optimality conditions for infinite-dimensional programming problems. Math. Programming 16, 98-110.

[16] Nachi, K. (1991) Dualité non convexe et analyse de sensibilité en dimension infinie. Thèse de Magister. Université d'Oran.

[17] Nachi, K. (2007) Thèse de Doctorat d'Etat. Université d'Oran.

[18] Penot, J.-P. (1984) Differentiability of relations and differential stability of perturbed optimization problems. SIAM J. of Control and Optimization 22, 529-551.

[19] Penot, J.-P. (2004) Differentiability properties of optimal value functions. Canad. J. Math. 56 (4), 825-842.

[20] Rockafellar, R.T. (1982) Lagrange multipliers and subderivatives of optimal value functions in nonlinear programming. Math. Progr. St. 17, 28-66.

[21] Rockafellar, R.T. (1982) Augmented Lagrangians and marginal values in parametrized optimization problems. Wierzbicki, A. Ed. Generalized Lagrangian methods in optimization. Pergamon Press, New York.

[22] Rockafellar, R.T. (1984) Directional differentiability of optimal value function in nonlinear programming problems. Math. Progr. St. 21, 213-226.

[23] Shapiro, A. (1985) Second order sensitivity analysis and asymptotic theory of parametrized nonlinear programs. Math. Progr 33, 280-299.

[24] Shapiro, A. (1988) Sensitivity analysis of nonlinear programs and differentiability properties of metric projections. SIAM J. of Control and Optim. 26, 628-645.

Generating efficient solutions with reservation levels in Multiobjective Stochastic Integer Problems

Fatima BELLAHCENE

Département de Mathématiques, Faculté des sciences, Université Mouloud Mammeri, Tizi-Ouzou, Algérie.
f_bellahcene@yahoo.fr

Abstract. This paper considers a special class of multiobjective stochastic integer linear programming (MOSILP) problems involving random variable coefficients in the constraints. The presumed constraints reliability levels are less than one, then chance constrained programming (CCP) is used to handle with the randomness. It is shown how these problems can be transformed into equivalent multiobjective nonlinear integer programming (EMONLIP) problem when the random variables are independent and normally distributed with mean and variance that are linear in the decision variables. The algorithm developed here is based on the notion of level sets and level curves. It finds the Pareto optimal solutions throughout a linear integer program defined by eigenvalue relaxation.

keywords: Multiple objective programming, Stochastic Programming, Chance constrained programming, nonlinear programming, level sets, level curves.

1 Introduction and background

Much of decision making in the real-world takes place in an environment where the objectives, constraints or parameters are not known precisely (see Lai and Hwang [14] and Liu [15]). Therefore a decision is often made on the basis of vague information or uncertain data.The uncertainty may be interpreted as randomness or fuzziness. The randomness occurring in the multiobjective linear programming (MOLP) problems is categorized as the multiobjective stochastic linear programming (MOSLP) problems. As we have known in the stochastic optimization problems, the coefficients of the problem are assumed as random variables with known distributions in most of cases. The books written by Birge and Louveaux [7], Kall [11], Prékopa [18], Klein Haneveld & Van der Vlerk [12] provide many interesting ideas and useful techniques for tackling the stochastic optimization problems. (MOSLP) models are appropriate when data evolve over time and decisions need to be prior to observing the entire data stream. Then, the way of modeling the (MOSLP) problems and obtaining efficient solutions depends in large part on the nature of available information about the random parameters.

The (MOSLP) models have been developed for a variety of applications, including portfolio selection (Aouni and al. [3], Ballestero and al. [4], Shing and al. [20], Ogryczak [17]), investment planing (Ben Abdelaziz and al. [5]) and electric power generation (Teghem and al. [22]), to mention a few.

Most previous efforts in this field have been devoted to positive decision variables (see, Stancu-Minasian [21] and Caballero [8]. In many situations, however, fractional

values of the variables are not physically meaningful. Therefore, modeling with multiobjective stochastic integer linear programming (MOSILP) programs and the development of solution algorithms for such problems are of great interest to management scientists.

Progress has not been substantial on (MOSILP) and the present literature on it is surprisingly thin (see for example, Abbas and Bellahcene [1], Saad and Kittani [19], Teghem [23]). In [1], the Benders decomposition method [6] and four types of cuts are used to develop a generating technique for identifying a compromise solution from a set of available candidates. The stochastic data are treated by a recourse approach to obtain an equivalent deterministic two-stages multiobjective integer linear programming (MOILP) problem and duality proprieties are used to check for feasibility of the recourse function. In [19], a solution algorithm is presented for solving integer linear programming problems involving dependent random parameters in the objective functions and linearly independent random parameters in the constraints. The STRANGE-MOMIX method presented by Teghem in [23] is interactive and based on the generalized Tchebycheff norm to generate efficient solutions.

In this paper we consider a special stochastic model called the multiobjective chance constrained integer linear problem. The random variables are assumed to be normally distributed with mean and variance that are linear in the decision variables. Since problems including randomness are usually transformed into nonlinear programming problems, it is difficult to find a global optimal solution efficiently. Furthermore, since our proposed models are multi-criteria stochastic programming problems, it is almost impossible to solve them directly. We manage to construct efficient solution methods for them using the equivalent transformations to the main problem based on the properties of random variable. Based on the notion of level sets and level curves, the algorithm developed here computes all the Pareto optimal solutions respecting given reservation levels.

The next section describes the considered stochastic model and shows how to convert it into an equivalent multiobjective deterministic nonlinear program. In section 3, we review some basic properties of level sets. Section 4, presents the eigenvalue relaxation of the resulting nonlinear program. The algorithm development is detailed in section 5. We conclude the paper with some considerations on possible future research in this field.

2 Problem statement and structural properties

We consider the multiobjective linear programming problems involving random variable coefficients in the objectives functions formulated as:

$$\begin{array}{c} "maximize"(u_1, u_2, \ldots, u_p) \\ subject\ to\ \ P_r[C_i^t(w)x \geq u_i] = \alpha_i,\ i = 1, \ldots, p \\ x \in S \end{array} \qquad \text{(P1)}$$

Where $S = \{x \in R^n \mid Ax \leq b,\ x \geq 0\ and\ integer\}$.
The parameters u_i, $i = 1, \ldots, p$, $A \in R^{m \times n}$, $b \in R^m$ and $x \in R^n$ represent deterministic problem data; w is a random vector from the probability space (Ω, Σ, P)

and $C_i(w) \in R^n$ represent stochastic parameters; $P_r\{t\}$ denotes the probability of the event $t \in \Sigma$ under the probability measure P_r ; and $\alpha_i,\ i = 1, \ldots, p$ are probability levels. This model deals with the optimization of upper allowable limits $u_i,\ i = 1, \ldots, p$ for given probabilities $\alpha_i,\ i = 1, \ldots, p$. For instance, this is a situation when expected value of the profit is considered not to be a good measure of criteria. In the following, the basic technique of chance constrained programming (CCP) is used to transform problem (P1) into an equivalent multiobjective nonlinear integer programming (EMONLIP) problem according to the predefined probabilities $\alpha_i,\ i = 1, \ldots, p$.

Assume that each random variable $C_i(w)$ has a multinormal distribution function with mean value vector $\overline{C}_i = (\overline{C}_{i,1}, \overline{C}_{i,2}, \ldots, \overline{C}_{i,n})$ and variance-covariance matrix V_i. Therefore, it is known that $C_i^t(w)x$ has a normal distribution function with mean $\overline{C}_i^t x$ and standard deviation $(x^t V_i x)^{1/2}$. Then the probabilistic constraints can be written as:

$$P_r[C_i^t(w)x \ \geq \ u_i] = \alpha_i \Longleftrightarrow \Phi\left(\frac{u_i - \overline{C}_i^t x}{(x^t V_i x)^{1/2}}\right) = 1 - \alpha_i$$
$$\Longleftrightarrow u_i(x) = \overline{C}_i^t x - \Phi^{-1}(\alpha_i)(x^t V_i x)^{1/2}$$

Where $\Phi(\cdot)$ is the distribution function of the standard normal distribution. The (EMOIP) of problem (P1) is shown in problem (P2) :

$$\begin{array}{c} maximize\ u_i(x) = \overline{C}_i^t x - \Phi^{-1}(\alpha_i)(x^t V_i x)^{1/2} \quad ,\ i = 1, \ldots, p \\ subject\ to\ \ x \in S \end{array} \tag{P2}$$

Kataoka [13] is credited for formulating problem (P1) and the development of the (EMONLIP) problem (P2). When $\alpha_i > 0.5$, the objective functions $u_i,\ i = 1, \ldots, p$ are concave. That is because $x^t V_i x$ is convexe for $i = 1, \ldots, p$ (see Ishii [10,p 184]). The values $\Phi^{-1}(\alpha_i)$ which are positive can be obtained from any standard normal distribution table.

3 Multiobjective optimization and level sets

In the following, we will use the concept of Pareto optimality to define the minimization in (P2).

Definition 1. *A solution $x^* \in S$ is called Pareto optimal if and only if there is no $x \in S$ such that $u_i(x) \geq u_i(x^*)$, $i = 1, \ldots, p$ and $u_i(x) > u_i(x^*)$ for at least one $i \in \{1, \ldots, p\}$. The set of all Pareto optimal solutions is denoted by $Spar$. If x^* is Pareto optimal then $u(x^*) = (u_1(x^*), \ldots, u_p(x^*))$ is called efficient.*

Independent of the properties of the objective functions u_i or the constraint set S, Pareto optimal solutions can be characterized geometrically. In order to state this characterization, we introduce the notion of level sets and level curves.

Definition 2. *Let $\beta_i \in R$ for $i = 1, \ldots, p$*
1. The set $L^i_{\geq}(\beta_i) = \{x \in S \mid u_i(x) \geq \beta_i\}$ is called the level set of u_i with respect to the level β_i
2. The set $L^i_{=}(\beta_i) = \{x \in S \mid u_i(x) = \beta_i\}$ is called the level curve of u_i with respect to the β_i.

The following characterization of Pareto optimal solutions by level sets and level curves was given by Ehrgott and al. [9].

Lemma 1. *Let $x^* \in S$. Then x^* is Pareto optimal if and only if*

$$\bigcap_{i=1}^{p} L^i_{\geq}(u_i(x^*)) = \bigcap_{i=1}^{p} L^i_{=}(u_i(x^*))$$

i.e. x^* is Pareto optimal if and only if the intersection of all p level sets of u_i with respect to levels $u_i(x^*)$ is equal to the intersection of the level curves of $u_i,\ i = 1, \ldots, p$ with respect to the same levels.

Because we will use the result of Lemma 1 throughout the paper the following notation will be convenient.
For $\beta \in R^p$ let

$$S(\beta) = \{x \in S \mid u_i(x) \geq \beta_i,\ i = 1, \ldots, p\} = \bigcap_{i=1}^{p} L^i_{\geq}(\beta_i)$$

Correspondingly, $S(\beta)_{par}$ will denote the Pareto set of $S(\beta)$.

Note that the range of values that efficient points can reach is given by a lower and upper bound on the efficient set defined by the ideal and the nadir point of the multiobjective programming (P2). The ideal point $u^I = (u^I_1, u^I_2, \ldots, u^I_p)$ is given by $u^I_i = max\ u_i(x)$ and the nadir point $u^N = (u^N_1, u^N_2, \ldots, u^N_p)$ is given by $u^N_i = min\ u_i(x)$. With the nadir point, we can choose $\beta_i = u^N_i$ for $i = 1, \ldots, p$ as lower bounds. The major difficulty in this choice is the need for a nonlinear algorithm. This article introduces an eigenvalue relaxation to define two linear functions as lower and upper bounds for the nonlinear objective function u_i.

4 The eigenvalue relaxation

In order to define the eigenvalue relaxation problem, we state some of well known results regarding the eigenvalues of symmetric positive definite matrices. The proofs of these results can be found in [16].

Proposition 1. *If V is an n by n symmetric positive definite matrix, then its eigenvalues are real and positive.*

Proposition 2. *If V is an n by n symmetric positive definite matrix, σ_1 and σ_n are its smallest and largest eigenvalue respectively, then*

$$\sigma_1 x^t x \leq x^t V x \leq \sigma_n x^t x \quad \forall x \in R^n$$

Knowing that a and b are two nonnegative numbers, the following inequality will hold:

$$(a+b)^{1/2} < a^{1/2} + b^{1/2} \tag{1}$$

According to a property of inequality (1), the linear function $\sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j$ may be used as an upper bound for the nonlinear term $(x^t V_i x)^{1/2}$ appearing in (P2). This can be stated as :

$$(x^t V_i x)^{1/2} < \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \tag{2}$$

For generalization purposes and for having both the upper and lower bound linear functions to correspond with the nonlinear term of problem (P2), we define inequality (3) as shown below :

$$-\sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j < -(x^t V_i x)^{1/2} < \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \tag{3}$$

By multiplication of the positive constant $\Phi^{-1}(\alpha_i)$ by all terms of inequality (3) and addition of a linear function of all decision variables, such as $\overline{C}_i^t x$, to those terms, inequality (4) would be obtained:

$$\overline{C}_i^t x - \Phi^{-1}(\alpha_i) \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \leq \overline{C}_i^t x - \Phi^{-1}(\alpha_i)(x^t V_i x)^{1/2} \leq \overline{C}_i^t x + \Phi^{-1}(\alpha_i) \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \tag{4}$$

Another way of exhibiting inequality (4) is shown in (5).

$$f_i(x) \leq u_i(x) \leq g_i(x) \ , \ \ i = 1, \ldots, p \tag{5}$$

Since inequality (5) would hold for all values of the feasible decision variables, therefore inequality (6) would also hold :

$$f_i^* \leq u_i^* \leq g_i^* \ , \ \ i = 1, \ldots, p \tag{6}$$

In (6), u_i^* is the optimal value of problem (P3) restricted to the ith objective function and f_i^* and g_i^* are the optimal values of problems (P4) and (P5) that are presented below.

$$\underset{x \in S}{max} \ f_i(x) = \overline{C}_i^t x - \Phi^{-1}(\alpha_i) \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \tag{P4}$$

and

$$\underset{x \in S}{max} \ g_i(x) = \overline{C}_i^t x + \Phi^{-1}(\alpha_i) \sum_{i=1}^{p} (\sigma_n^i)^{1/2} x_j \tag{P5}$$

5 Pareto optimal solutions with reservation levels

In this section, we develop a method for the determination of Pareto optimal solutions in the multiobjective (P2) based on the characterization given in Lemma 1. The procedure uses an algorithm which solves the problem of finding a K-best solution in a multiobjective integer programming problem (see for example [2] or [24]). The goal is to find all Pareto optimal solutions of problem (P2) respecting given reservation levels $\beta_i,\ i = 1, \ldots, p$. In other words we want to to compute $S(\beta)_{par}$. Instead of an explicit computation of the intersection of level sets and checking the condition of Lemma 1, we will generate one level set $L_{\geq}(\beta_1)$ (without loss of generality) in order of decreasing values of the corresponding objective function, and then check for each element of this level set if it is also contained in the other level sets and if it dominates or is dominated by a solution found before.

Algorithm

Input: Instance of a (MOSILP) with p criteria, reservation levels $\beta_1, \ldots, \beta_p$.
Output: The set $S(\beta)_{par}$ of all Pareto optimal solutions respecting reservation levels β.

Step 1 : Set $(\beta_1, \beta_2, \ldots, \beta_p) = (f_1^*, f_2^*, \ldots, f_p^*)$.

Step 2 : Let x_1 be the optimal solution of problem $\underset{x \in S}{max}\ g_1(x)$.

If $g_1(x^1) < \beta_1$ then stop $S(\beta)_{par} = \varnothing$
$k = 1$
$S(\beta)_{par} = \left\{x^k\right\}$.

Step 3 : $k = k + 1$
Apply a ranking algorithm to compute the k-best solution x^k of g_1.
If $g_1(x^k) < \beta_1$ then stop $S(\beta)_{par} = \varnothing$.

Step 4 : If $x^k \in L_{\geq}$ for all $i = 2, \ldots, p$ then goto step 5
else goto step 3.

Step 5 : For $1 \leq i \leq k-1$
If x^kdominates x^i then $S(\beta)_{par} = S(\beta)_{par} \setminus \left\{x^i\right\}$
else if x^i dominates x^k then goto step 3.
else if $g_1(x^k) = u_1(x^i)$ then $S(\beta)_{par} = S(\beta)_{par} \cup \left\{x^k\right\}$ goto step 3.

Step 6 : $S(\beta)_{par} = S(\beta)_{par} \cup \left\{x^k\right\}$
goto step 3.

6 Conclusion

In this paper, we attempted to solve a particular multiobjective stochastic integer linear problem by level sets. The nondominated solutions are determined by solving a linear integer program defined by eigenvalue relaxation. The construction of the lower bounds

(levels) is fairly simple. Our method does not require specific mathematical properties to be satisfied by the objectives. It appears—on several examples— that the algorithm performs faster, however further experimental validation of this observation is needed.

References

1. Abbas, M. and Bellahcene F., (2006). Cutting plane method for multiple objective stochastic integer linear programming problem. European Journal of operational research 168 (3), 967-984.
2. Abbas M. and Moulaï M., (1999). Solving multiple objective integer linear programming. Ricerva Operativa, 29 (89), 15-38.
3. Aouni, B., Ben Abdelaziz, F., Martel, J.M., (2005). Decision-maker's preferences modeling in the stochastic goal programming. European Journal of Operational Research 162, 610–618.
4. Ballestero, E., (2005). Using stochastic goal programming: Some applications to management and a case of industrial production. Information Systems and Operational Research Journal 43 (2), 63–77.
5. Ben Abdelaziz, F., Mejri, S., (2001). Application of goal programming in a multi-objective reservoir operation model in Tunisia. European Journal of Operational Research 133, 352-361.
6. Benders J.F, (1962). Partitioning procedures for solving mixed-variables programming problems. Numer. Math. 4, 238-252.
7. Birge A. J.R., and Louveaux F.V., (1997). Introduction to Stochastic Programming. Springer Verlag, New York.
8. Caballero R., Cerda E., Muvnoz M.M., and Rey L., (2000). Relations among every several efficiency concepts in stochastic multiple objective programming. In: Research and Practice in Multiple Criteria Decision Making Haimes, Y.Y., Steuer, R. (Eds.), Lectures notes in Economics and Mathematical Systems. Springer-Verlag, Berlin, Germany, 57-68.
9. Ehrgott M. , H.W. Hamacher, K. Klamroth, S. Nickel, A. Schobel, and M.M. Wiecek., (1997). A note on the equivalence of balance points and Pareto solutions in multiple-objective programming. Journal of Optimization Theory and Applications, 92(1), 209 - 212.
10. Ishii, H. Nishida, T., and Nanbu Y., (1978). A generalized chance constrained programming problem. Journal of Operations Research Society of Japan 21(1).
11. Kall, P. and Wallace S.W., (1994). Stochastic Programming. Wiley, Chichester, Also available as PDF file at http://www.unizh.ch/ior/Pages/Deutsch/ Mitglieder/Kall/bib/ka-wal-94.pdf.
12. Klein Haneveld, W.K. and van der Vlerk, M.H. (1999). Stochastic integer programming: General models and algorithms. Ann. Oper. Res., 85, 39-57.
13. Kataoka, S. (1963). A stochastic programming model. Econometrica 31, 181-196.
14. Lai, Y.J. and Hwang, C.L. (1994). Fuzzy Multiple Objective Decision Making: Methods and Applications. Springer-Verlag, Berlin, New York.
15. Liu, B. (2002). Theory and Practice of Uncertain Programming; Physica-Verlag, Heidelberg.
16. Minc, H. and Marcus. M., (1964). A survey of Matrix Theory and Matrix inequalities. Allyn and Bacon Inc., Boston.
17. Ogryczak, W. (2000). Multiple criteria linear programming model for portfolio selection. Annals of Operations Research 97, 143-162.
18. Prékopa, A. (1995). Stochastic Programming. Kluwer Academic Publishers, Boston.
19. Saad, O. M. and Kittani, H. F. (2003). Multiobjective integer linear programming problems under randomness. LAPOR TRANSACTIONS, 28(2), 101-108.

20. Shing, C., Nagasawa, H., (1999). Interactive decision system in stochastic multi-objective portfolio selection. International Journal of Production Economics 60-61,187-193.
21. Stancu-Minasian I.M., (1984). Stochastic Programming with Multiple Objective Functions. D. Reidel Publishing Company.
22. Teghem J., and Kunsch P. L., (1985). Multi-objective decision making under uncertainty: an example for power systems. In Haimes Y.Y. and Chankong V. (Eds), Decision Making with Multiple Objective, Springer, 443-456.
23. Teghem J., (1990). Strange-Momix : an Interactive method for mixed integer linear programming; in R. Slowinski and J. Teghem (eds), Stochastic Versus Fuzzy Approaches to Multiobjective Mathematical Programming Under Uncertainty; Dordrecht: Kluwer Academic Publishers, 101-115.
24. Verma, V., Bakhshi H.C., and Puri M.C., (1991). Constrained integer linear programming; Asia-Pacific Journal of Operational Research 8, 72-79.

Résolution d'un problème de programmation quadratique avec une M-matrice

Katia HASSAINI et Mohand Ouamer BIBI

Laboratoire de Modélisation et d'Optimisation des Systèmes (LAMOS)
Université de Béjaia 06000, Algérie,
Hassaini@Gmail.com mohandbibi@yahoo.fr

Résumé Dans ce papier, une méthode de résolution d'un problème de programmation quadratique avec une M-matrice et des contraintes simples est présentée. Elle se base sur les algorithmes de R. Chandrasekaran [1] et de F.T. Luk et M. Pagano [2]. Ces méthodes utilisent le fait qu'une M-matrice possède une inverse non négative qui permet d'avoir une suite monotone de solutions réalisables [3,4]. En introduisant le concept de support pour une fonction objectif [16], notre approche se différencie par une condition plus générale qui permet d'avoir une solution réalisable initiale plus proche de la solution optimale. La programmation sous MATLAB de notre méthode et de celle de Luk et Pagano nous a permis de faire une comparaison entre ces dernières, de les illustrer par deux exemples pratiques, et ce, en variant le nombre de variables.

Mots clés : Programmation quadratique convexe, M-matrices, méthode de Chandrasekaran , méthode de Newton projetée, méthode de support.

1 Introduction

Dans la littérature, plusieurs approches ont été proposées pour la résolution des problèmes de programmation quadratique quand la matrice D associée est définie positive. Cependant, il est possible d'exploiter les propriétés spéciales d'une M-matrice pour obtenir des algorithmes spéciaux plus efficaces. D'ailleurs, de tels problèmes avec des M-matrices trouvent des applications dans la résolution numérique des équations aux dérivées partielles elliptiques. Ces problèmes incluent divers types de problèmes de Dirichlet avec obstacles [11,12], les modèles d'application de torsion à une barre [19], etc. Les M-matrices sont aussi connues pour avoir de nombreuses applications dans la modélisation des systèmes dynamiques, dans les sciences économiques et écologiques [13,14,15]. Plusieurs de leurs propriétés sont utilisées pour établir des résultats de stabilité pour les systèmes dynamiques en général [8,9].

Dans cet article, on propose une nouvelle approche qui se distingue de celles de R. Chandrasekaran, F.T. Luk et M. Pagano, par une condition plus générale qui permet d'avoir une solution réalisable initiale plus proche de la solution optimale. La programmation sous MATLAB de notre méthode et de celle de Luk et Pagano nous a permis de faire une comparaison entre ces dernières, de les illustrer par deux exemples pratiques, et ce, en variant le nombre de variables.

2 Position du problème et définitions

Considérons le problème suivant de programmation quadratique avec contraintes simples :

$$\begin{cases} F(x) = \frac{1}{2}x^T Dx + c^T x \longrightarrow \min, \\ \\ x \geq 0, \end{cases} \tag{1}$$

où $c = c(J) = (c_j, \quad j \in J)$ et $x = x(J) = (x_j, \quad j \in J)$ sont des n-vecteurs réels, $J = \{1, 2, \cdots, n\}$. La matrice $D = D(J, J)$ est une M-matrice carrée symétrique $(D = D^T)$ d'ordre n. Le symbole (T) est celui de la transposition.

Rappelons qu'une M-matrice $D = (d_{ij}, \quad 1 \leq i, j \leq n)$ satisfait les conditions suivantes :

$$d_{ii} > 0, \qquad d_{ij} \leq 0, \quad i \neq j, \qquad D^{-1} \geq 0,$$

où le symbole $D^{-1} \geq 0$ veut dire que tous les éléments de la matrice D^{-1} sont positifs ou nuls.

Remarque 1. Une M-matrice symétrique est toujours définie positive $(x^T Dx > 0, \quad \forall \ x \neq 0)$.

Définition 1.
Un vecteur $x \geq 0$ est appelé *solution réalisable* du problème (1). Une solution réalisable x^0 est dite *optimale* si elle réalise le minimum de la fonction objectif du problème (1).
Ainsi, une solution réalisable x^0 du problème (1) est optimale si et seulement si pour tout $j \in J$, les conditions d'optimalité suivantes sont satisfaites :

$$\begin{cases} x_j^0 = 0 \quad \Rightarrow \quad g_j(x^0) \geq 0, \\ \\ x_j^0 > 0 \quad \Rightarrow \quad g_j(x^0) = 0, \quad j \in J, \end{cases} \tag{2}$$

où $g(x) = g(J) = Dx + c$ est le gradient de la fonction objectif F au point x.

Considérons le problème de programmation quadratique sans contraintes :

$$\begin{cases} F(x) = \frac{1}{2}x^T Dx + c^T x \longrightarrow \min, \\ \\ x \in \mathbb{R}^n. \end{cases} \tag{3}$$

La solution optimale $\widehat{x}$ du problème (3) vérifie

$$D\widehat{x} + c = 0 \iff \widehat{x} = -D^{-1}c.$$

Soient J_S et J_N une partition de J :

$$J_S \cup J_N = J, \quad J_S \cap J_N = \emptyset.$$

Le gradient de la fonction F au point x peut alors s'écrire sous la forme suivante :

$$g = \begin{pmatrix} g_S \\ g_N \end{pmatrix}, \quad g_S = g(J_S) = D_S x_S + D_{SN} x_N + c_S, \quad g_N = g(J_N) = D_{NS} x_S + D_N x_N + c_N,$$

où

$$x = \begin{pmatrix} x_S \\ x_N \end{pmatrix}, \; c = \begin{pmatrix} c_S \\ c_N \end{pmatrix}, \; D_S = D(J_S, J_S), \; D_N = D(J_N, J_N), \; D_{SN} = D(J_S, J_N).$$

Définition 2.

- Le sous-ensemble J_S est appelé *support* de la fonction objectif, car il vérifie toujours la condition :

$$\det D_S = \det D(J_S, J_S) \neq 0.$$

  La paire $J_p = \{J_S, J_N\}$ est alors dite support du problème (1).
- Le couple $\{x, J_p\}$, formé d'une solution réalisable x et du support J_p est appelé *solution réalisable de support*.

Définition 3.

- Un vecteur $\kappa = \kappa(J) = (\kappa(J_S), \kappa(J_N))$ vérifiant

$$\begin{cases} \kappa_N = 0, \\ \\ \kappa_S = -D_S^{-1} c_S \; . \end{cases}$$

  est dit *pseudo-solution* du problème (1).
  Une pseudo-solution vérifie toujours $g_S(\kappa) = 0$.
- Le support $J_p = \{J_S, J_N\}$ est appelé support coordinateur s'il existe une pseudo-solution κ telle que :

$$g_j(\kappa) \geq 0, \qquad j \in J_N. \tag{4}$$

  On dit dans ce cas que la pseudo-solution κ est associée au support coordinateur J_P.

Théorème 1. *Une pseudo-solution κ associée à un support coordinateur J_p est optimale dans le problème (1) si et seulement si*

$$\kappa_j \geq 0, \qquad j \in J_S. \tag{5}$$

Remarque 2. Toute pseudo-solution κ, associée à un support coordinateur J_p, est solution réalisable du dual du problème primal (1) :

$$\begin{cases} F(\kappa) = -\frac{1}{2} \kappa^T D \kappa \longrightarrow \max, \\ \\ \qquad D\kappa + c \geq 0. \end{cases} \tag{6}$$

Remarque 3. Comme $g(\widehat{x}) = 0$, alors la solution optimale $\widehat{x}$ du problème sans contraintes (3) est une pseudo-solution du problème (1), associée à un support coordinateur $J_p = \{J_S, J_N\}$, où $J_S = J$ et $J_N = \emptyset$. D'aprés le théorème 1, si $\widehat{x} \geq 0$, alors $x^0 = \widehat{x}$ est une solution optimale du problème (1).

Rappelons le lemme suivant :

Lemme 1. *[2].*

(a) *Si* $c \geq 0$, *alors* $x^0 = 0$ *résoud le problème (1),*

(b) *Si* $c \leq 0$, *alors* $x^{\,0} = -D^{-1}c$ *résoud le problème (1).*

Pour éviter les deux cas triviaux précédents, considérons le cas général où c contient des composantes positives et négatives. Construisons alors les deux ensembles suivants tels que :

$$J_S = \{j \in J : \widehat{x}_j \geq 0\}, \quad J_N = \{j \in J : \widehat{x}_j < 0\}, \qquad J_S \cup J_N = J.$$

– Si $J_S = J$, alors $x^0 = \widehat{x} = -D^{-1}c$ est une solution optimale du problème (1).

– Sinon, soit y la projection de $\widehat{x}$ sur le domaine admissible du problème (1), où $y = (y_j, \ j \in J)$, $y_j = \max\{0, \widehat{x}_j\}$. Donc on aura

$$\begin{cases} y_N = 0 > \widehat{x}_N \ . \\ \\ y_S = \widehat{x}_S, \end{cases}$$

Lemme 2. *L'inégalité suivante est vérifiée*

$$g_S(y) \leq g_S(\widehat{x}).$$

Preuve 1. *On a*

$$\begin{aligned} g_S(\widehat{x}) &= D_S \ \widehat{x}_S + D(J_S, J_N) \ \widehat{x}_N \ + \ c_S \\ &= D_S \ y_S + \ c_S + \ D(J_S, J_N) \ \widehat{x}_N \\ &= g_S(y) + \ D(J_S, J_N) \ \widehat{x}_N \\ &\geq \ g_S(y), \end{aligned}$$

car $D(J_S, J_N) \leq 0$ *et* $\widehat{x}_N < 0$. □

Puisque $g_S(\widehat{x}) = 0$, alors du lemme 2 on déduit que $g_S(y) \leq 0$. On construit alors un vecteur x tel que

$$\begin{cases} x_N = \ y_N = \ 0 \ , \\ \\ x_S = -D_S^{-1} \ c_S \ . \end{cases} \tag{7}$$

On a donc

$$g_S(y) \ \leq \ 0 \quad et \qquad g_S(x) = 0.$$

Lemme 3. *Les vecteurs* y *et* x *vérifient l'inégalité suivante :*

$$x_S \ \geq \ y_S \ \geq \ 0.$$

Preuve 2. *On a*

$$D_S \ (x_S - y_S) = D_S \ x_S + c_S - [D_S \ y_S + c_S].$$

Comme $x_N \ = \ y_N \ = \ 0$, $g_S(y) \ \leq \ 0$ *et* $g_S(x) = 0$, *alors on déduit*

$$D_S \ (x_S - y_S) = g_S(x) - g_S(y) \geq 0.$$

La sous-matrice D_S *étant une M-matrice [2], il en résulte que* $D_S^{-1} \geq 0$. *Par conséquent, en multipliant à gauche par* D_S^{-1} *l'inégalité ci-dessus, on obtient*

$$x_S \ \ \geq \ \ y_S \ \geq \ \ 0. \ \square$$

En vertu du lemme 3, le vecteur x construit (7) est une solution réalisable du problème (1), et d'après [4], il vérifie encore l'inégalité $x \ \leq \ x^0$, où x^0 est la solution optimale du problème (1). On a alors le lemme suivant :

Lemme 4.

$$F(x) \ \leq \ F(y).$$

Preuve 3. *Soit*

$$2F(x) = x_S^T \ D_S x_S + 2 \ c_S^T \ x_S.$$

Comme $x_S = -D_S^{-1} c_S$, *on aura*

$$2F(x) = c_S^T \ D_S^{-1} c_S - 2 \ c_S^T \ D_S^{-1} c_S = -c_S^T \ D_S^{-1} c_S.$$

La sous-matrice D_S *étant définie positive, alors on peut écrire*

$$\begin{aligned} 2F(x) \leq \ & (y_S - x_S)^T \ D_S (y_S - x_S) - c_S^T \ D_S^{-1} c_S \\ \leq \ & y_S^T \ D_S \ y_S + x_S^T \ D_S \ x_S - 2 y_S^T \ D_S \ x_S - c_S^T \ D_S^{-1} \ c_S \\ \leq \ & y_S^T \ D_S \ y_S + c_S^T \ D_S^{-1} \ D_S \ D_S^{-1} \ c_S + 2 \ y_S^T \ D_S \ D_S^{-1} \ c_S - c_S^T \ D_S^{-1} \ c_S \\ \leq \ & y_S^T \ D_S \ y_S + 2 \ c_S^T \ y_S \\ \leq \ & 2 \ F(y) \end{aligned}$$

D'où

$$F(x) \ \leq \ F(y). \square$$

Remarque 4. Le vecteur x construit (7) vérifie ainsi l'inégalité suivante :

$$F(\widehat{x}) < F(x^0) \leq F(x) \leq F(y), \tag{8}$$

où y est la projection de $\widehat{x}$ sur le domaine admissible du problème (1).

Théorème 2. *[2]. Supposons que $D_S^{-1}c_S \leq 0$ pour un certain sous-ensemble non vide $J_S \subset J$. On définit un vecteur x avec $x_S = -D_S^{-1}c_S$ et $x_N = 0$. Soient J_N^- et J_N^+ deux ensembles qui partitionnent J_N tels que*

$$J_N^- = \{j \in J_N : \ g_j(x) < 0\},$$
$$J_N^+ = \{j \in J_N : \ g_j(x) \geq 0\}.$$

Si l'ensemble J_N^- est vide, alors

1. *x résoud le problème (1), sinon*

2. *Soit $\overline{J_S} := J_S \cup J_N^-$. Construisons $\overline{x}$ avec $\overline{x}(\overline{J_S}) = -D^{-1}(\overline{J_S}, \overline{J_S})\, c(\overline{J_S})$ et $\overline{x}(J_N^+) = 0$. On obtient*

(a) $\overline{x}(J_S) \geq x(J_S) \geq 0, \quad \overline{x}(J_N^-) \geq 0, \quad \overline{x}(J_N^+) = 0,$

(b) $g_j(\overline{x}) \leq g_j(x) \quad j \in J_N^+,$

(c) $F(\overline{x}) < F(x)$.

3 Algorithme de la méthode

Calculer $\widehat{x}$, solution optimale du problème (3) :

$$g(\widehat{x}) = D\widehat{x} + c = 0 \Longrightarrow \widehat{x} = -D^{-1}c.$$

– Si $\widehat{x} \geq 0$, alors terminer et le vecteur $x^0 = \widehat{x}$ est une solution optimale du problème (1).

– Sinon, construire les ensembles :

$$J_S = \{j \in J : \ \widehat{x}_j \geq 0\}, \quad J_N = \{j \in J : \ \widehat{x}_j < 0\}.$$

– Construire x de la manière suivante :

$$x_N = 0, \quad x_S = -D_S^{-1}c_S.$$

Soient J_N^- et J_N^+ deux ensembles qui partitionnent J_N tels que

$$J_N^- = \{j \in J_N : \ g_j(x) < 0\}, \quad J_N^+ = \{j \in J_N : \ g_j(x) \geq 0\}.$$

Répéter jusqu'à ce que l'ensemble J_N^- soit vide

(1) Calculer $g_N(x) = D(J_N, J_S)\, x_S + c_N$,

(2) Poser $J_N^- = \{j \in J_N : \ g_j(x) < 0\}$,

(3) Si l'ensemble J_N^- est non vide, alors

(a) Poser $J_S := J_S \cup J_N^-$ et $J_N := J_N \setminus J_N^-$,

(b) Reconstruire x tel que $x_N = 0$ *et* $x_S = -D_S^{-1} c_S$.

4 Exemples numériques et comparaisons

Nous avons choisi deux problèmes représentatifs. Le but est d'examiner l'efficacité de notre algorithme et de faire une comparaison avec l'algorithme de F.T. Luk et M. Pagano [2]. Les deux méthodes ont été programmées sous le langage MATLAB 7.0 R14, sous le système d'exploitation Windows XP, avec une RAM de 512 Mo, CPU 5.00GHz. On définit par

- **Algorithme1** : Méthode de Chandrasekaran, de F.T. Luk et M. Pagano [2].
- **Algorithme2** : Notre algorithme.
- NJ_S : Le cardinal de l'ensemble J_S, juste après avoir calculé $\widehat{x} = -D^{-1}c$.
- $\overline{NJ_S}$: Le cardinal de l'ensemble J_S, à l'optimum.
- NP : Le cardinal de l'ensemble P, à l'initialisation de x.
- $\overline{NP}$: Le cardinal de l'ensemble P, à l'optimum.
- **Itération** : Le nombre d'itérations effectuées par chaque algorithme.
- **Temps** : Le temps machine (CPU times) d'exécution en secondes.

4.1 Exemple 1.

Considérons le programme quadratique (1) :

$$\begin{cases} F(x) = \frac{1}{2} x^T D x + c^T x \longrightarrow \min, \\ \\ \qquad\qquad\qquad\qquad\qquad x \geq 0. \end{cases}$$

La matrice D choisie est telle que

$$D = \begin{pmatrix} 2 & -1 & & & \bigcirc \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ \bigcirc & & & -1 & 2 \end{pmatrix}_{n \times n} . \tag{9}$$

On choisit de générer le vecteur c de façon à avoir trois cas de l'ensemble J_S. On pose r_i un nombre aléatoire qui suit une loi uniforme $r_i \in U\,[0,1]$. Les temps

machine d'exécution de chaque programme pour la résolution d'un programme quadratique sous la forme (1) sont représentés dans les tableaux ci-dessous :

Cas 1. On génère le vecteur c de manière à avoir $NJ_S = n$:

$$c_i = 11 - 23\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{10}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	Temps
500	09	259	500	0.7969	0	500	500	0.1250
1000	11	539	1000	4.3125	0	1000	1000	0.1875
1500	13	783	1500	14.2188	0	1500	1500	0.4219
2000	15	1028	2000	34.7500	0	2000	2000	1.3125

Tab. 1. CPU Times (en secondes) avec $NJ_S = n$.

Cas 2. Le vecteur c est généré par

$$c_i = 11 - 22\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{11}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	temps
500	16	262	498	1.1875	10	373	498	0.5000
1000	24	508	995	14.1250	08	922	995	1.4036
1500	23	752	1487	23.8125	21	727	1487	6.1719
2000	26	975	1921	70.0156	28	970	1921	13.7031

Tab. 2. CPU Times (en secondes) avec $NJ_S < n$.

Cas 3. On génère le vecteur c par

$$c_i = 11 - 20\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{12}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	temps
500	06	219	360	0.3438	08	0	360	0.2969
1000	09	467	764	2.5469	10	0	764	1.0156
1500	12	660	1118	6.2500	13	0	1118	2.6094
2000	13	930	1591	16.4063	14	0	1591	5.1875

Tab. 3. CPU Times (en secondes) avec $NJ_S = 0$.

Dans cet exemple, on remarque bien que notre approche est plus efficace en temps machine que l'approche de Chandrasekaran, de F.T. Luk et M. Pagano. Et cela, quelque soit le cardinal de J_S à l'étape initiale de l'algorithme.

4.2 Exemple 2.

Dans cet exemple, on choisit la matrice D comme étant une matrice déduite de l'approximation du Laplacien par des différences finies en 5-points :

$$D = \begin{pmatrix} B & -1 & & \bigcirc \\ -1 & B & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & B & -1 \\ \bigcirc & & & -1 & B \end{pmatrix}_{m^2 \times m^2}, \tag{13}$$

où

$$B = \begin{pmatrix} 4 & -1 & & \bigcirc \\ -1 & 4 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 4 & -1 \\ \bigcirc & & & -1 & 4 \end{pmatrix}_{m \times m}. \tag{14}$$

Soit $n = m^2$. On considère la matrice de "mise à jour partielle" E telle que :

$$E = \begin{pmatrix} B & & & \bigcirc \\ & B & & \\ & \ddots & \ddots & \ddots \\ & & & B & \\ \bigcirc & & & & B \end{pmatrix}_{n \times n}. \tag{15}$$

La variable r_i étant toujours un nombre aléatoire qui suit une loi uniforme $r_i \in U\,[0,1]$, on considère les 3 cas suivants :

Cas 1. Le vecteur c est généré par

$$c_i = 8 - 20\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{16}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	Temps
500	03	299	500	0.3438	0	500	500	0.1563
1000	04	594	1000	1.9844	0	1000	1000	0.7188
1500	04	921	1500	4.5000	0	1500	1500	2.2188
2000	04	1210	2000	9.8438	0	2000	2000	4.3281

Tab. 4. CPU Times (en secondes) avec $NJ_S = n$.

Cas 2. Le vecteur c est généré par

$$c_i = 8 - 16\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{17}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	Temps
500	10	259	472	0.7656	13	2	472	1.1094
1000	14	495	948	4.7188	16	388	948	7.7813
1500	16	729	1442	15.7969	18	153	1442	26.7656

Tab. 5. CPU Times (en secondes) avec $NJ_S < n$.

Cas 3. Le vecteur c est généré par

$$c_i = 8 - 10\ r_i \qquad pour \quad i = 1, 2, \ldots, n. \tag{18}$$

Dimension	Algorithme1				Algorithme2			
n	Itération	NP	$\overline{NP}$	Temps	Itération	NJ_S	$\overline{NJ_S}$	Temps
500	01	91	100	0.0938	4	0	100	0.2188
1000	03	219	246	0.2500	5	0	246	0.8125
1500	02	259	337	0.2969	05	0	337	2.2344
2000	02	392	444	0.9531	05	0	444	4.5781

Tab. 6. CPU Times (en secondes) avec $NJ_S = 0$.

5 Conclusion

Les lemmes 2, 3 et 4 nous ont permis de démarrer l'algorithme avec une solution réalisable x vérifiant les conditions du théorème 2 et l'inégalité (8). Dans le cas où la matrice de notre fonction objectif $D = I$, où I est l'identité, on aura

$$J_S = \{j \in J : c_j \leq 0\}, \quad J_N = \{j \in J : c_j > 0\},$$

et on retrouve les conditions d'initialisation des algorithmes de R. Chandrasekaran, de F.T. Luk et M. Pagano [1,2]. Remarquons encore que les algorithmes cités terminent avec J_N ou $J_{\overline{N}}$ vide, alors que le nôtre termine toujours avec $J_{\overline{N}} = \emptyset$ et $J_N \neq \emptyset$, et ce, à cause de notre initialisation. En effet, le cas $J_N = \emptyset$ correspond à la solution optimale $x^0 = \widehat{x}$.

Dans l'exemple 1, on remarque bien que notre approche est plus efficace en temps machine que l'approche de Chandrasekaran, de F.T. Luk et M. Pagano. Et cela, quelque soit le cardinal de J_S à l'étape initiale de l'algorithme.

Dans le deuxième exemple, on remarque dans le tableau (4) que notre approche est meilleure que l'autre et cela résulte du fait que la solution réalisable initiale est elle-même solution optimale du problème. Par contre, on voit bien que dans les tableaux (5) et (6), l'algorithme 1 se comporte mieux que le nôtre.

Références

1. Chandrasekaran, R. : A special case of the complementary pivot problem. Opsearch, **7** (1970), 263–268.
2. Luk, F. T. and Pagano, M. : Quadratic programming with M-Matrices. Linear Algebra And Its Applications, **33** (1980), 15–40.
3. Stachurski, A. : Monotone sequences of feasible solutions for quadratic programming problems with M-matrices and box constraints. In System Modeling and Optimization. Book series : Lecture Notes in Control and Information Sciences, (1986), Vol 84, 896–902.
4. Stachurski, A. : An Equivalence between two algorithms for a class of quadratic programming problems with M-matrices. Optimization, **21(6)** (1990), 871–878.
5. Li, L. and Kobayashi, Y. : A block recursive algorithm for the linear complementarity problem with an M-matrix. International Journal of Innovative, Computing, Information and Control, Vol 2, Number 6, (2006), 1327–1335.
6. Kunish, K. and Rendl, F. : An infeasible active set method for convex problems with simple bounds. SIAM Journal on optimization, **14(1)**(2003), 35–52.
7. Voglis, C. and Lagaris, I. E. : BOXCQP : An Algorithm for Bound Constrained Convex Quadratic Problems. 1st International Conference "From Scientific Computing To Computational Engineering", Vol 1, (2004), 261–268, Athens, Greece, September 8-10.
8. Stipanovic, D. M and Šiljak, D. D. : Stability of polytopic systems via convex M-matrices and parameter-dependent Liapunov functions. Nonlinear Analysis, **40** (2000), 589–609.
9. Stipanovic, D. M., Shankaran, S. and Tomlin, C.J. : Multi-agent avoidance control using an M-matrix property. Electronic Journal of Linear Algebra. A publication of the International Linear Algebra Society, **12** (2005), 64–72.
10. Pang, J. S. : On a class of least-element complementarity problems.(1976), Report SOL 76-10, Systems Optimization Lab, Stanford Univ.
11. Levati, G., Scarpini, F. and Volpi, G. : Sul trattemento numerico di alcuni problemi variazionali di tipo unilaterale. L.A.N. Pub.82, Univ. of Pavia (1974).
12. Scarpini, F. : Some algorithms solving the unilateral Dirichlet problems with two constraints. Calcolo, **12**(1975), 113–149.
13. Šiljak, D. D. : Large-scale Dynamic Systems : Stability and Structure. North-Holland, New York, (1978).
14. Varga, R. S. : Matrix Iterative Analysis. Prentice-Hall, Englewood Cliffs, N.J. (1962).
15. Berman, A. and Plemmons, R. J. : Nonnegative Matrices in Mathematical Sciences. Academic Press, New York, (1979).
16. Gabasov, R., Kirillova, F. M., Kostyukova, O. I. and Raketsky, V.M. : Constructive methods of optimization, volume 4 : Convex Problems. University Press, Minsk (1987).
17. Brahmi, B. and Bibi, M. O. : Méthode duale de support pour la résolution des problèmes quadratiques convexes à variables bornées. In Actes du Colloque sur l'Optimisation et les Systèmes d'Information, 11-13 juin 2007, USTO, Oran, pp.365-376.
18. Bibi, M. O. and Radjef, S. : Solution of a convex quadratic program with mixed variables via the direct support method. In Actes du Colloque MOAD : Méthodes et Outils d'Aide à la Décision, 18-20 Novembre 2007, Université de Béjaia, p.39-44.

19. Céa, J. and Glowinski, R. : Sur des méthodes d'optimisation par relaxation. RAIRO, **R-3** (1973) 5–32.

20. Bertsekas, D. P. : Projection Newton Method for Optimization Problems with Simple Constraints. SIAM Journal on Control and Optimization, **20**(1982)2, 221–246.

Algorithme itératif d'optimisation globale des fonctions β-höldériennes utilisant les courbes α-denses

M. Rahal [1] *et* A. Ziadi [2]

Laboratoire de Mathématiques Fondamentales et Numériques, Université de Sétif, Algérie
email : 1. mrahal_ dz@yahoo.fr 2. ziadiaek@yahoo.fr

Résumé : Dans ce papier on propose un algorithme itératif pour résoudre le problème d'optimisation globale des fonctions höldériennes à plusieurs variables. On présente une variante d'une méthode déterministe. Il s'agit de la méthode de la transformation réductrice Alienor. Cette transformation permet de ramener une fonction à n variables à une fonction d'une seule variable qui conserverait les minimiseurs globaux au moins de façon approchée. La technique utilisée est basée sur la réduction de la dimension en utilisant des "courbes remplissant l'espace". Ces courbes ont l'avantage d'être de classe C^∞ et préservent les propriétés de la fonction objectif. La méthode Aliénor s'est révélée être d'une grande efficacité en s'associant à certaines méthodes unidimensionnelles telles que les algorithmes de recouvrement. Le couplage d'Alienor avec ces algorithmes a été appliqué sur des fonctions tests de plusieurs variables ayant un minimum global difficile à trouver par les méthodes classiques. Nous étudions dans notre travail le couplage de la méthode Alienor avec l'algorithme d'Evtushenko dans le cas où la fonction objectif est höldérienne de paramètres h et β et définie sur un pavé P de $\mathbb{R}^n$. Des résultats intéressants concernant l'approximation du minimum et le temps de calcul ont été réalisés.

Mots clés : Optimisation Globale, Fonction höldérienne, Méthode de recouvrement, Méthode de la transformation réductrice, courbes α-denses.

1. Introduction

Considérons le problème de minimisation globale non-convexe suivant :

$$\min_{(x_1,x_2,...,x_n)\in P} f(x_1, x_2, ..., x_n) \qquad (\mathbf{P})$$

que l'on veut résoudre avec une précision exigée $\varepsilon > 0$, avec f une fonction höldérienne de constante $h > 0$ et d'exposant $\frac{1}{\beta}$ ($\beta > 1$) et définie sur un pavé $P = \prod_{i=1}^{n}[a_i,b_i]$ de $\mathbb{R}^n$, et à valeurs dans $\mathbb{R}$.

1.1. Définition et propriétés des fonctions höldériennes

Définition 1.1. Une fonction $f : P \subset \mathbb{R}^n \to \mathbb{R}$ est dite höldérienne sur P, s'il existe deux constantes réelles $h > 0$ et $\beta > 1$, telles que

$$\forall\ x, y \in P\ , \qquad |f(x) - f(y)| \leq h \left\|x - y\right\|^{\frac{1}{\beta}} \tag{1.1}$$

où $\|.\|$ est la norme euclidienne.

• Si f est une fonction höldérienne de constante h et d'exposant $\frac{1}{\beta}$ sur P , alors elle est de même pour toute constante $h^{'} > h$ et d'exposant $\frac{1}{\beta'}$ avec $\beta^{'} > \beta$, sur P .
• Bien que les fonctions höldériennes sont continues, elles peuvent être non différentiables. Intuitivement, les fonctions höldériennes telles que β est assez grand sont beaucoup plus irrégulières que celles ou β est assez petit. Ce qui explique le fait que $\beta > 1$.
• Puisque toutes les normes dans $\mathbb{R}^n$ sont équivalentes alors toute fonction höldérienne pour une norme est aussi höldérienne pour les autres normes. Par exemple si f vérifie (1.1), alors

$$|f(x) - f(y)| \leq h \left(\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \right)^{\frac{1}{\beta}} \leq h \left(\sqrt{n}\right)^{\frac{1}{\beta}} \left(\max_{1 \leq i \leq n} |x_i - y_i| \right)^{\frac{1}{\beta}},$$

d'où

$$|f(x) - f(y)| \leq h \left(\sqrt{n}\right)^{\frac{1}{\beta}} \left\|x - y\right\|_{\infty}^{\frac{1}{\beta}}.$$

2. Principe général des méthodes de recouvrement

Elles sont basées sur la détection des sous-régions ne contenant pas le minimum global et de leur exclusion de la poursuite de la recherche. Donnons d'abord l'idée clé de ces méthodes [7] (on va par la suite spécifier la description). Supposons que la fonction f est évaluée aux points d'un maillage $x_1, x_2, ..., x_k$ suffisamment dense du pavé P garantit la détection, avec une précision donnée, du minimum global. Soit

$$m_k^* = \min \left\{ f(x_1), f(x_2), ..., f(x_k) \right\}. \tag{2.1}$$

Pour chaque point x_i du maillage on définit un sous-ensemble $A_i \subset P$ tel que

$$A_i = \{x \in P :\ f(x) \leq m_k^* - \epsilon\}, \tag{2.2}$$

donc si $\bigcup_{i=1}^{k} A_i$ recouvre P alors le problème est résolu.
D'où le problème de minimisation est ramené à la construction d'une suite de points $\{x_i\}_{1 \leq i \leq k}$ vérifiant $P \subset \bigcup_{i=1}^{k} A_i$.

3. Méthode d'Evtushenko unidimensionnelle pour les fonctions höldériennes

Dans le cas unidimensionnel le problème (**P**) a été résolu dans les travaux réalisés de [4, 5, 6], en utilisant les sous-estimateurs de la fonction objectif. Le calcul des minimiseurs globaux des fonctions minorantes ce fait en déterminant le seul point d'intersection des courbes paraboliques ce qui conduit à résoudre à chaque itération une équation algébrique non-linéaires de degrée β.

Dans ce travail, nous allons présenter une méthode itérative d'optimisation globale pour résoudre le problème (**P**), en s'inspirant de l'algorithme de recouvrement itératif non-uniforme d'Evtushenko [1, 2, 3] pour les fonctions lipschitziennes. Cette méthode a la réputation d'être efficace en dimension 1, elle est plus connue et plus commentée dans la littérature, nous allons montrer qu'elle peut être etendue aux fonctions höldériennes. La mise en oeuvre de l'algorithme et la technique a l'avantage de ne pas utiliser des calculs intermédiaires difficiles, tels la construction des fonctions minorantes ou exiger de la fonction d'être dérivable, ce qui permet de réduire considérablement le temps de calcul par rapport aux autres techniques.
Le chema de l'algorithme qu'on utilise est de la forme suivante :

$$x_{k+1} = x_k + r_{ik}$$

où r_{ik} des scalaires dépends de f et les constantes h, β et ϵ.

Théorème 3.1. Soit f une fonction höldérienne de paramètres $h > 0$ et $\beta > 1$, définie sur un intervalle $[a, b]$ de $\mathbb{R}$. Supposons que f soit évaluée aux points $x_1, x_2, ..., x_k$. Posons $m_k^* = \min\limits_{1\leq i\leq k} \{f(x_i)\}$ et désignons par $(I(x_i, r_{ik}))_{1\leq i\leq k}$ une famille d'intervalles de centres $\{x_i\}_{1\leq i\leq k}$ et de rayons r_{ik}. Si on a $[a, b] \subset \bigcup\limits_{i=1}^{k} I(x_i, r_{ik})$ avec $r_{ik} = \left(\frac{f(x_i)-m_k^*+\varepsilon}{h}\right)^{\beta}$ alors m_k^* est le minimum global, à ϵ près, *de f sur* $[a, b]$.

Preuve. D'après l'inégalité (1.1) on a

$$\forall\ x, y \in [a, b]\ , \qquad f(y) - h\,|x - y|^{\frac{1}{\beta}} \leq\ f(x), \tag{3.1}$$

ce qui fait pour $y \in [a, b]$ fixé, si un certain x vérifie

$$m_k^* - \epsilon \leq\ f(y) - h\,|x - y|^{\frac{1}{\beta}}\,, \tag{3.2}$$

alors $m_k^* - \epsilon \leq f(x)$. On considère les intervalles

$$I(x_i, r_{ik}) = \{x \in \mathbb{R},\ \ |x - x_i| \leq r_{ik}\}\,;$$

de l'inégalité (3.2) on a $|x - y| \leq \left(\frac{f(y)-m_k^*+\varepsilon}{h}\right)^{\beta}$,

et pour $y = x_i$, on doit donc avoir les rayons r_{ik} donnés par $r_{ik} = \left(\frac{f(x_i)-m_k^*+\varepsilon}{h}\right)^{\beta}$.
On peut montrer facilement que pour tout $i = 1, ..., k$ et pour tout
$x \in I(x_i, r_{ik})$ on a

$$m_k^* - \epsilon \leq f(x).$$

Soit maintenant $M = \min\limits_{x\in[a,b]} f(x) = f(x_0)$, $x_o \in [a, b]$, on a $x_0 \in \bigcup\limits_{i=1}^{k} I(x_i, r_{ik})$, donc il existe $i_0 \in \{1, 2, ..., k\}$ tel que $x_0 \in I(x_{i_0}, \left(\frac{f(x_{i_0})-m_k^*+\varepsilon}{h}\right)^{\beta})$
par conséquent

$$|x_{i_0} - x_0| \leq \left(\frac{f(x_{i_0}) - m_k^* + \varepsilon}{h}\right)^{\beta},$$

donc

$$|x_{i_0} - x_0|^{\frac{1}{\beta}} \leq \frac{f(x_{i_0}) - m_k^* + \varepsilon}{h}. \tag{3.3}$$

Puisque f est (h,β)-höldérienne et d'après (3.3) on a

$$|f(x_{i_0}) - f(x_0)| \leq f(x_{i_0}) - m_k^* + \varepsilon.$$

D'où

$$m_k^* - M \leq \epsilon.$$

On déduit que m_k^* est une solution optimale de (**P**). ■

Donc si la réunion des intervalles $I(x_i, r_{ik})$ ne couvre pas $[a,b]$, le minimum global peut être atteint dans $[a,b]$-$\bigcup_{i=1}^{k} I(x_i, r_{ik})$. Par conséquent, les intervalles $I(x_i, r_{ik})$ peuvent être omis de l'ensemble faisable $[a,b]$; on cherche la solution dans la partie restante. Le problème (**P**) aura une solution lorsque la réunion des intervalles couvre complètement $[a,b]$.

La représentation donnée suggére une méthode constructive pour résoudre le problème (**P**). En résumé, la méthode consiste à procéder ainsi : supposons que pour une certaine suite de points $\{x_k\}$ le record m_k^* est déterminé par la relation (2.1). La suite des points x_i et les rayons r_{ik} de l'intervalle sont stockés dans une mémoire. Si au nouveau point x_{k+1} on a $f(x_{k+1}) < m_k^*$, on pose $m_{k+1}^* = f(x_{k+1})$ et on remplace le terme r_{ik} par r_{ik+1}. Si les intervalles I_{ik+1} recouvrent l'intervalle $[a,b]$, alors le calcul sera stoppé ; sinon, on prend un nouveau point x_{k+2} et on continue. L'ensemble $[a,b]$ est recouvert par des intervalles de différents rayons (non-uniformité).

Considérons un point x_i auquel $m_i^* = f(x_i)$, $1 \leq i \leq k$, on a évidemment $f(x_i) = m_i^* \geq m_k^*$ d'où,

$$r_{ik} = \left(\frac{f(x_i) - m_k^* + \varepsilon}{h}\right)^{\beta} \geq (\frac{\epsilon}{h})^{\beta}.$$

Donc le plus petit rayon est au point x_i auquel $m_i^* = f(x_i)$ i.e., $(\frac{\epsilon}{h})^{\beta}$. On prend donc $x_1 = a + (\frac{\epsilon}{h})^{\beta}$, parce que à l'initialisation $f(x_1) = m_1^*$. Avec cette valeur de x_1, on gagne du temps et on est sûr de ne pas avoir ignorer le minimum global au voisinage de ce point. En effet, si

$$x^* = \arg\min_{x \in [a,b]} f(x) \in [a, a + (\frac{\epsilon}{h})^{\beta}]$$

alors :

$$|f(x_1) - f(x^*)| \leq h\,|x_1 - x^*|^{\frac{1}{\beta}} \leq \varepsilon.$$

En général, pour $i \geq 1$, la suite $\{x_{i+1}\}$ est définie par :

$$x_{i+1} = x_i + \left(\frac{f(x_i) - m_k^* + \varepsilon}{h}\right)^{\beta} + (\frac{\epsilon}{h})^{\beta}.$$

Ce choix nous permet de ne pas rater le minimum global de f, car les intervalles I_i se rencontrent. On arrête quand k vérifie $[a,b] \subset \bigcup_{i=1}^{k} I(x_i, r_{ik})$.

Si $x_k < b$ et $x_k + r_{kk} \geq b$, alors le dernier point de la suite est x_k.

Si $x_k + r_{kk} < b$ et $x_k + r_{kk} + (\frac{\epsilon}{h})^{\beta} \geq b$ le dernier point de la suite sera $x_{k+1} = b$.

Algorithme d'Evtushenko

1. Initialisation

Poser $k = 1$, $x_1 = a + (\frac{\epsilon}{h})^{\beta}$, $x_{\epsilon} = x_1$, $f_{\epsilon} = f(x_{\epsilon})$.

2. Etapes, $k = 2, 3, ...$

Poser $x_{k+1} = x_k + (\frac{\epsilon}{h})^{\beta} + (\frac{f(x_k) - f_{\epsilon} + \varepsilon}{h})^{\beta}$

Si $x_{k+1} > b - (\frac{\epsilon}{h})^{\beta}$, alors arrêter.

Sinon, déterminer $f(x_{k+1})$.
Si $f(x_{k+1}) < f_\epsilon$, alors poser $x_\epsilon = x_{k+1}$, $f_\epsilon = f(x_{k+1})$.
Poser $k = k + 1$ et aller à 2.

4. La méthode de la transformation réductrice

Peu de travaux ont été faits pour l'optimisation globale des fonctions höldériennes à plusieurs variables. Le principe fondamental de la méthode de la transformation réductrice [12] consiste à effectuer une transformation qui permet de ramener le problème multidimensionnel à un problème unidimensionnel afin d'appliquer les méthodes d'optimisation plus efficaces adaptées au cas d'une seule variable. L'idée de base consiste à densifier l'ensemble faisable par une courbe paramétrée (α-dense) continue et assez régulière. La fonction multivariable est transformée en une fonction d'une seule variable. Ainsi notre problème est ramené à un problème plus facile à résoudre car il y a une seule direction à explorer.

Dans cette section, nous allons présenter la méthode Alienor, élaborée par Y. Cherruault et coll. [9, 10]. L'idée consiste à approcher une fonction de plusieurs variables par une fonction d'une seule variable, il devenait alors assez simple de déterminer les minima globaux car il suffisait de les chercher en suivant l'évolution d'une certaine courbe.
Soient n variables $x_1, ..., x_n$, la méthode donc consiste à exprimer ces variables à l'aide d'une seule en densifiant l'espace $\mathbb{R}^n$ à l'aide d'une simple courbe. Nous allons mettre en évidence ce que nous appellerons la transformation réductrice :

$$x_i = \varphi_i(t),\ t > 0,\ i = 1, ..., n.$$

On désigne par μ la mesure de Lebesgue, A un intervalle fermé et borné de $\mathbb{R}$ (en général $A = [0, T]$, avec $T > 0$) et α un nombre réel, strictement positif et supposé très petit par rapport aux dimensions du pavé $P = \prod_{i=1}^{n} [a_i, b_i]$, ($\alpha << \min_{1 \le i \le n} (b_i - a_i)$). Où $n \ge 2$ est un entier.

Définition 4.1. On dit qu'une courbe paramétrée de $\mathbb{R}^n$ définie par

$$\varphi : A \to P = \prod_{i=1}^{n} [a_i, b_i]$$

est α-dense dans P ou bien qu'elle densifie P avec une densité α si pour chaque $x \in P$, il existe $t \in A$ tel que

$$d(x, \varphi(t)) \le \alpha,$$

où d est la distance euclidienne dans $\mathbb{R}^n$.

On construit alors une courbe paramétrée $\varphi(t) = (\varphi_1(t), \varphi_2(t), ..., \varphi_n(t))$,
α-dense dans l'ensemble faisable $P = \prod_{i=1}^{n} [a_i, b_i]$ pour tout $t \in [0, T]$.
De cette façon, la fonction $f(x_1, ..., x_n)$ devient :

$$f^*(t) = f(\varphi(t)) = \ f(\varphi_1(t), \varphi_2(t), ..., \varphi_n(t)),$$

et le problème initial de minimisation ($\mathbf{P}$) est alors approximé par le problème de minimisation unidimensionnel

$$\min_{t \in [0,T]} f^*(t) \qquad (\mathbf{P}^*)$$

où T est la borne supérieure du domaine de définition de la fonction φ qui permet de α-densifier P.
Mais deux questions essentielles se posent, existe-t-il une méthode générale permettant de générer ces courbes α-denses ? et peut-on trouver des classes de transformation minimisant le temps de calcul du minimum global ?.
Des réponses positives ont été données par A. Ziadi et Y. Cherruault [11] et les ont améliorée par l'obtention de nouveaux résultats qui permettent de construire de grandes classes de courbes α-denses. Cette dernière voie s'est révélée être la plus intéressante car les courbes obtenues possèdent des représentations paramétriques plus simples. Une étude complète et détaillée est donnée dans [9, 10, 11].

4.1. Une nouvelle transformation réductrice

La caractérisation et la génération des courbes α-denses dans un pavé de $\mathbb{R}^n$ $(n \geq 2)$ est un sujet fondamental. D'autre part, l'un des objectif les plus importants est l'application des courbes α-denses à l'optimisation globale. Mais, tout cela dépend essentiellement de la longueur de la courbe qui densifie l'ensemble feasable.
En se basant sur un résultat donné dans [10]. On définit d'une manière constructive une courbe α-dense dans un pavé quelconque de $\mathbb{R}^n$.

Théorème 4.1. Soient $\varphi_1, \varphi_2, ..., \varphi_n$ des fonctions continues surjectives, respectivement définies de A dans $[a_i, b_i]$ pour $i = 1, 2, ..., n$, et soient aussi $t_1, t_2, ..., t_{n-1}$et α des nombres strictement positifs tels que pour tout
$i = 1, 2, ..., n-1$, il existe une partition finie de Acomposée d'intervalles $(I_{i,j})_{1 \leq j \leq m_i}$ et vérifiant :
(a) $\mu(I_{i,j}) = t_i, \quad \forall j = 1, 2, ..., m_i$.
(b) $\varphi_i(\overline{I}_{i,j}) = [a_i, b_i], \quad \forall j = 1, 2, ..., m_i$
(c) $\mu(\varphi_{i+1}(I_{i,j})) \leq \alpha, \quad \forall j = 1, 2, ..., m_i$.
Alors la courbe paramétrée définie par la fonction $\varphi = (\varphi_1, \varphi_2, ..., \varphi_n)$ est $\sqrt{n-1}\alpha$-dense dans $\prod_{i=1}^{n} [a_i, b_i]$.

On peut voir la preuve dans Ziadi [10].

Théorème 4.2. On considère dans $\mathbb{R}^n$ le pavé $\prod_{i=1}^{n} [a_i, b_i]$ et α un nombre strictement positif tel que le nombre $\frac{b_i - a_i}{\alpha}$ soit un entier pour tout $i = 1, 2, ..., n$. Posons $T = \frac{\prod_{i=1}^{n} (b_i - a_i)}{\alpha^n}$ et soient $\varphi_1, \varphi_2, ..., \varphi_n$des fonctions, respectivement définies de $[0, T]$ dans $[a_i, b_i]$ par :

$$\begin{aligned}
\varphi_i(t) &= a_i + \frac{\sigma_i(t)}{t_i}(b_i - a_i) \qquad \textit{pour } i = 1, 2, ..., n \ ; \\
\textit{avec} &\qquad t_0 = 1, \quad t_i = \frac{b_i - a_i}{\alpha} t_{i-1} \\
\sigma_i(t) &= (-1)^{\beta_i(t)} [t - (\beta_i(t) + \frac{1}{2}\left(1 - (-1)^{\beta_i(t)}\right)) t_i] \\
\beta_i(t) &= Ent(\frac{t}{t_i}) \quad \text{où } \textit{Ent} \ \textit{ est l'application " partie entière"}
\end{aligned}$$

Alors la fonction définie par $\varphi(t) = (\varphi_1(t), \varphi_2(t), ..., \varphi_n(t))$ pour $t \in [0, T]$ représente une courbe paramétrée $\sqrt{n-1}\alpha$-dense dans $\prod_{i=1}^{n} [a_i, b_i]$.

Preuve.
(*a*) Posons $m_i = \frac{\prod_{k=i+1}^{n}(b_k - a_k)}{\alpha^{n-i}}$ pour $i = 1, 2, ..., n-1$.
Les hypothèses impliquent que pour tout $i = 1, 2, ..., n-1$, le nombre m_i est un entier et $t_n = m_i t_i = T$. Considérons les intervalles

$$\begin{aligned} & I_{i,j} = [(j-1)t_i,\ jt_i[\qquad \text{pour } j = 1, 2, ..., m_i - 1 \\ \text{et} \quad & I_{i,m_i} = [(m_i - 1)t_i,\ m_i t_i]. \end{aligned}$$

Il est facile de voir que pour tout $i = 1, 2, ..., n-1$, la famille $(I_{i,j})_{1 \le j \le m_i}$ forme une partition de l'intervalle $[0, T]$ et $\mu(I_{i,j}) = t_i, \quad \forall j = 1, 2, ..., m_i$.
En outre, on peut montrer que :
(*b*) pour tout $i = 1, 2, ..., n-1$,

$$\varphi_i(\overline{I}_{i,j}) = [a_i, b_i], \quad \forall j = 1, 2, ..., m_i,$$

(*c*) pour tout $i = 1, 2, ..., n-1$,

$$\mu(\varphi_{i+1}(I_{i,j})) \le \alpha, \quad \forall j = 1, 2, ..., m_i.$$

Il en résulte que toutes les hypothèses du Théorème 3 sont satisfaites.
Donc la courbe paramétrée définie par $\varphi(t) = (\varphi_1(t), \varphi_2(t), ..., \varphi_n(t))$,
pour $t \in [0, T]$, est $\sqrt{n-1}\alpha$-dense dans $\prod_{i=1}^{n}[a_i, b_i]$. ■

Proposition 4.1. La fonction $f^*(t) = f(\varphi(t))$ pour $t \in [0, T]$ est une fonction höldérienne de constante h^* et d'exposant $\frac{1}{\beta}$, où h^* est donnée par l'expression suivante :

$$h^* = h\left(\sum_{i=1}^{n}(\frac{b_i - a_i}{t_i})^2\right)^{\frac{1}{2\beta}}.$$

4.2. La méthode mixte Alienor-Evtushenko

Nous étudions dans cette section, ce couplage dans le cas où la fonction objectif est höldérienne de paramètres h et β, et définie sur un pavé P.

Première étape. En se basant sur le Théorème 4.2, on définit l'application

$$\varphi = (\varphi_1, \varphi_2, ..., \varphi_n) : [0, T] \to \prod_{i=1}^{n}[a_i, b_i],$$

qui représente une courbe α-dense dans P, avec $\alpha = \frac{1}{\sqrt{n-1}}\left(\frac{\epsilon}{2h}\right)^{\beta}$, le paramètre α est choisi de telle façon que le minimum global soit estimé avec la précision ϵ.

Deuxième étape. On applique, maintenant l'algorithme d'Evtushenko à la fonction d'une seule variable $f^*(t) = f(\varphi(t))$ pour $t \in [0, T]$.
Bien évidemment $f^*(t)$ est (h^*, β)-höldérienne.

1. Initialisation Poser $k = 1$, $t_1 = (\frac{\epsilon}{h^*})^{\beta}$, $t_\epsilon = t_1$, $f_\epsilon^* = f^*(t_\epsilon)$.
2. Etapes, $k = 2, 3, ...$
Poser $t_{k+1} = t_k + (\frac{\epsilon}{h^*})^{\beta} + (\frac{f^*(t_k) - f_\epsilon^* + \varepsilon}{h^*})^{\beta}$
Si $t_{k+1} > T - (\frac{\epsilon}{h^*})^{\beta}$, alors arrêter.
Sinon, déterminer $f^*(t_{k+1})$.
Si $f^*(t_{k+1}) < f_\epsilon^*$, alors poser $t_\epsilon = t_{k+1}$, $f_\epsilon^* = f^*(t_{k+1})$, .
Poser $k = k + 1$ et aller à 2.

Théorème 4.3. La méthode mixte Alienor-Evtushenko appliquée au problème (**P**) converge en un nombre fini de points vers le minimum global avec une précision inférieure ou égale à ϵ.

Preuve. La suite générée par l'algorithme d'Evtushenko est évidemment finie puisque la distance entre deux points successifs de la suite est supérieure ou égale à $(\frac{\epsilon}{h})^{\beta}$.
Notons par m et $m^{'}$ respectivement les minima globaux de f et f^{*}. D'autre part, désignons par f_{ϵ}^{*} le minimum global du problème (**P**) obtenu par Alienor-Evtushenko. Montrons que

$$f_{\epsilon}^{*} - m \leq \epsilon.$$

Puisque f est continue sur P, il existe un point $y \in P$ tel que $m = f(y)$. Par ailleurs, il existe $t_0 \in [0, T]$ tel que

$$\|y - \varphi(t_0)\| \leq (\frac{\epsilon}{2h})^{\beta}, \text{ donc,} \qquad |f(y) - f(\varphi(t_0))| \leq \frac{\epsilon}{2}.$$

Il en résulte que $f(\varphi(t_0)) - m \leq \frac{\epsilon}{2}$.
Et puisque $m \leq m^{'} \leq f(\varphi(t_0))$, on en déduit

$$m^{'} - m \leq \frac{\epsilon}{2} \tag{4.1}$$

D'autre part, soit $(t_k)_{1 \leq k \leq N}$ la suite de points d'évaluation de la fonction f^{*} générée par l'algorithme d'Evtushenko. Il existe $t^{*} \in [0, T]$
tel que $f^{*}(t^{*}) = m^{'}$.
Si le point t^{*} est compris entre 0 et t_1, ou entre t_N et T, alors l'inégalité $f_{\epsilon}^{*} - m^{'} < \frac{\epsilon}{2}$ est évidente. Considérons le cas où $t_k \leq t^{*} \leq t_{k+1}$, pour un certain $k \in \{1, ..., N-1\}$. On a

$$t_{k+1} = t_k + \left(\frac{f^{*}(t_k) - f_{\epsilon}^{*} + \varepsilon}{h^{*}} \right)^{\beta}.$$

1) Si $t_k \leq t^{*} \leq t_k + \left(\frac{f^{*}(t_k) - f_{\epsilon}^{*}}{h^{*}} + \frac{\varepsilon}{2h^{*}} \right)^{\beta}$, puisque la fonction f^{*} est
h^{*}-höldérienne, alors

$$|f^{*}(t^{*}) - f^{*}(t_k)| \leq h^{*} |t^{*} - t_k|^{\frac{1}{\beta}} \leq h^{*} \left(\frac{f^{*}(t_k) - f_{\epsilon}^{*}}{h^{*}} + \frac{\varepsilon}{2h^{*}} \right),$$

par conséquent,

$$f_{\epsilon}^{*} - f^{*}(t^{*}) \leq \frac{\epsilon}{2}.$$

d'où, on obtient

$$f_{\epsilon}^{*} - m^{'} \leq \frac{\epsilon}{2}.$$

2) Si maintenant $t_k + \left(\frac{f^{*}(t_k) - f_{\epsilon}^{*}}{h^{*}} + \frac{\varepsilon}{2h^{*}} \right)^{\beta} \leq t^{*} \leq t_{k+1}$, alors

$$|f^{*}(t^{*}) - f^{*}(t_{k+1})| \leq h^{*} |t^{*} - t_{k+1}|^{\frac{1}{\beta}} \leq h^{*} \left(\left(\frac{\varepsilon}{2h^{*}} \right)^{\beta} \right)^{\frac{1}{\beta}} = \frac{\varepsilon}{2}.$$

Par conséquent

$$f^{*}(t_{k+1}) - f^{*}(t^{*}) \leq \frac{\varepsilon}{2},$$

d'où, on obtient

$$f_{\epsilon}^{*} - m^{'} < \frac{\epsilon}{2}. \tag{4.2}$$

Finalement, on déduit de (4.1) et (4.2) que

$$f_{\epsilon}^{*} - m < \epsilon.$$

4.3. Le temps de calcul du minimum global

Il n'est pas toujours facile d'estimer le temps de calcul nécessaire à l'obtention d'un minimum global. Pour la recherche d'un minimiseur global de la fonction approximée $f^*(t)$, on peut procéder à une discrétisation de pas Δt de l'intervalle $[0, T]$, sur lequel se trouve un minimiseur [13]. On construit ainsi l'ensemble :

$$M = \{f^*(i\Delta t),\ i = 1, ..., N, \text{ avec } N\Delta t = T\}$$

Nous allons voir comment choisir α et le pas Δt pour obtenir une précision désirée $\epsilon > 0$.

Soit l la constante de Lipschitz de la fonction $\varphi(t)$ qui représente la courbe α-dense dans P.
Il est évident que la fonction $f^*(t) = f(\varphi(t))$ est höldérienne de paramétres $hl^{\frac{1}{\beta}}$ et β sur $[0, T]$.

Notons par m (resp. m^*) le minimum global de f (resp. f^*) et par f^*_ϵ le minimum global obtenue par la méthode Alienor, on doit donc avoir

$$f^*_\epsilon - m \leq \epsilon.$$

La continuité de f sur P implique l'existence d'un point $x_0 \in P$ tel que $m = f(x_0)$. De plus, l'α-densité de φ entraîne l'existence de $t_0 \in [0, T]$ tel que :

$$\|x_0 - \varphi(t_0)\| \leq \alpha \text{ et } \|f(x_0) - f(\varphi(t_0))\| \leq h\alpha^{\frac{1}{\beta}}.$$

Mais

$$m \leq m^* \leq f(\varphi(t_0)),$$

et par conséquent

$$0 \leq m^* - m \leq h\alpha^{\frac{1}{\beta}}.$$

D'autre part, la continuité de f^* entraine :

$$f^*_\epsilon - m^* \leq hl^{\frac{1}{\beta}}\frac{\Delta t}{2};$$

ce qui implique

$$f^*_\epsilon - m \leq h\alpha^{\frac{1}{\beta}} + hl^{\frac{1}{\beta}}\frac{\Delta t}{2}.$$

Pour obtenir la précision $\epsilon > 0$ désirée, il suffit de choisir :

$$\alpha = \left(\frac{\epsilon}{2h}\right)^{\beta}, \quad \Delta t = \frac{\epsilon}{hl^{\frac{1}{\beta}}}$$

Le temps de calcul dans cette méthode est donné par

$$\mathbb{T} = \frac{Thl^{\frac{1}{\beta}}}{\epsilon}\mathbb{T}_0.$$

Où $\mathbb{T}_0$ est le temps moyen de calcul de $f^*(t)$ pour t fixé.

5. Tests numériques

Les fonctions tests données ci-dessous ayant la particularité de posséder plusieurs minima locaux.

Exemples de fonctions höldériennes (avec $\varepsilon = 0.1$).

1) $f_1(x) = \sqrt{1-x^2},\ x \in [-1,1]$, pour cette fonction on a $h = \sqrt{2}$, $\beta = 2,\ x_{opt} = -0.98521$

2) [5] $f_2(x) = \sum_{k=1}^{5} k\,|\sin((3k+1)x+k)|\,|x-k|^{\frac{1}{5}},\ x \in [0,10]$, on a $h = 77,\ \beta = 5,\ x_{opt} = 1.29865$

3) $f_3(x,y) = |x+y-0.25|^{\frac{2}{3}} - 3\cos\frac{x}{2},\ (x,y) \in \left[-\frac{1}{2},\frac{1}{2}\right]^2$, $h = 2,42\ ,\ \beta = \frac{3}{2},\ x^*_{opt} = (-0.00598, 0.39887)$

4) $f_4(x,y) = \sum_{k=1}^{3} \frac{1}{k}\left|\cos((\frac{3}{k}+1)x+\frac{1}{k})\right|\,|x-y|^{\frac{1}{3}},\ (x,y) \in [0,10]^2,\ h = 14,77,$ $\beta = 3,\ x^*_{opt} = (0.000898, 0.0011)$

5) [14] $f(x,y) = -\cos x \cos y \exp(1 - \frac{\sqrt{x^2+y^2}}{\pi}),\ (x,y) \in [-20,20]^2$, $h = 45,265,\ \beta = \frac{1}{2}, f^*_{opt} = -0.96354.$

6. Résolution des systèmes d'équations algébriques non-linéaires

Considérons le système d'équations algébriques non linéaires :

$$f_i(x_1, x_2, ..., x_n) = 0, \quad 1 \le i \le m \le n \qquad (\mathbf{S})$$

avec f_i des fonctions höldériennes de paramètres $h_i > 0$ et $\beta_i > 1$,
et définies sur le pavé $P = \prod_{i=1}^{n} [a_i,b_i]$. Nous allons voir l'application de l'algorithme itératif d'optimisation qu'on a vu précédemment sur le système ($\mathbf{S}$). D'abord, introduisons sur P, la fonction $H : \mathbb{R}^n \to \mathbb{R}^m$ par

$$H(x) = (f_1(x), ..., f_m(x))\,.$$

Les solutions approximatives de l'équation :

$$H(x) = 0, \quad x \in P,$$

serons données par les points de l'ensemble suivant

$$P_\epsilon = \{x \in P,\ \|H(x)\| \le \epsilon\}\,,$$

où ϵ est la précision. Pour résoudre ($\mathbf{S}$) il suffit de trouver au moins $x^* \in P_\epsilon$.
Le système ($\mathbf{S}$) est équivalent au problème d'optimisation globale

$$\min_{x \in\ P} f\ (x)$$

où $f\ (x) = \|H(x)\|$.

Proposition 6.1. Si f_i sont des fonctions höldériennes sur P de paramètres $h_i > 0$ et $\beta_i > 1$, pour $1 \le i \le m$, alors la fonction $f(x)$ est höldérienne sur P de paramètres $h = \left(\sum_{i=1}^{m} h_i^2\right)^{\frac{1}{2}}$ et $\beta = \max_i \beta_i$.

Proposition 6.2. $x^* \in P$ est une solution du système (**S**) si et seulement si : $0 = f_* = \|H(x^*)\| = \min\{\|H(x)\|, x \in P\}$.

Exemple. Soit le système d'équations algébriques

$$\begin{cases} \sqrt{1-x^2} = 1 \\ |\sin(x+1)|\,|x-2|^{\frac{1}{3}} = 1 \end{cases}$$

On pose : $f_1(x) = \sqrt{1-x^2} - 1$ et $f_2(x) = |\sin(x+1)|\,|x-2|^{\frac{1}{3}} - 1$

Ces deux fonctions sont höldériennes sur $[-1,1]$ successivement de constantes
$h = \sqrt{2},\ \beta = 2$ et $h^{'} = 2,\ \beta^{'} = 3$
Ce système est équivalent au problème d'optimisation globale suivant :

$$\min_{x\in[-1,1]} f(x),$$

avec $f\ (x) = \|H(x)\|$ et $H(x) = (f_1(x), f_2(x))$.

En appliquant l'algorithme d'Evtushenko, on trouve pour $\epsilon = 0.1$, la solution du système $x = -0.118963$.

7. Conclusion

La généralisation des méthodes de recouvrement est très difficile à réaliser au cas de plusieurs variables. Cette généralisation est encore plus compliquée si la classe traitée est formée de fonctions höldériennes. Ceci est dû au comportement de ce type de fonctions qui varient plus vite que les fonctions lipschitziennes et au fait que ces fonctions font intervenir deux paramètres h et β. On a présenté un algorithme de recouvrement itératif pour résoudre le problème d'optimisation globale de fonctions höldériennes à une seule variable. La généralisation de la méthode itérative d'Evtushenko au cas multidimensionnel peut être étendue facilement, mais sa mise en oeuvre sur machine est très compliquée. Concernant la méthode de la transformation réductrice, nous avons apporté des nouvelles idées pour pouvoir appliquer cette approche aux problèmes d'optimisation des fonctions höldériennes. L'algorithme obtenue à partir du couplage de la nouvelle variante d'Alienor avec l'algorithme d'Evtushenko est assez simple et relativement efficace et la convergence de la méthode mixte est prouvée. Aussi nous avons appliqué l'algorithme itératif à la résolution des systèmes d'équations algébriques non linéaires.

Références

[1] Yu. G. Evtushenko, *Algorithm for finding the global extremum of a function(case of a non-uniforme mesh)*, USSR Comput. Mathem. and Phys., 11, No. 6, 1390-1403. (1971).

[2] Yu. G. Evtushenko, V. U. Malkova, and A. A. Stanevichys, *Parallelization of the Global Extremum Searching Process*, Automation and Remote Control, Vol. 68, No. 5, pp. 787-798. (2007).

[3] R. Horst and P. M. Pardalos, *Handbook of Global Optimization*, Kluwer Academic Publishers, Dordrecht. (1995).

[4] E. Gourdin, B. Jaumard, and R. Ellaia , *Global Optimization of Hölder function*, J. of Global Optimization, Vol. 8, pp. 323-348. (1996).

[5] D. Lera and Ya. D. Sergeyev, *Global Minimization Algorithms for Hölder functions*, BIT, Vol 42, No. 1, pp. 119-133. (2002).

[6] M. Rahal and A. Ziadi, *A new extension of Piyavskii's method to Hölder functions of several variables*, Applied mathematics and Computation. Vol. 197, pp. 478-488. (2008).

[7] M. Rahal, *Extension de certaines méthodes de recouvrement en optimisation globale*. Thèse de Doctorat en sciences, Université Ferhat Abbas, Sétif, (2009).

[8] H. Sagan, *Space-Filling Curves*, Springer, New york, (1994).

[9] A. Ziadi, Y. Cherruault, and G. Mora, *The existance of α-dense Curves with minimal length in a metric space,* Kybernetes, Vol. 29, No. 2, pp. 219-230. (2000).

[10] A. Ziadi, and Y. Cherruault, *Generation of α-dense Curves in a cube of $\mathbb{R}^n$*, Kybernetes, Vol. 27, No. 4, pp. 416-425. (1998).

[11] A. Ziadi, and Y. Cherruault, *Generation of α-dense Curves and Applications to Global Optimization,* Kybernetes, Vol. 29, No.1, pp. 71-82. (2000).

[12] A. Ziadi, Y. Cherruault and G. Mora, *Global Optimization, a New Variant of the Alienor Method*, Comp. Math. Appl., Vol. 41, pp. 63-71. (2001).

[13] G. Mora and Y. Cherruault, *The theoretic calculation time associated with α-dense Curves*, Kybernetes, 27(8-9), pp. 919–39. (1998).

[14] S. K. Mishra, *Some new test functions for global optimization and performance of repulsive particle swarm method*, Munich Personal RePEc Archive, Paper No. 2718, (2006).

Extraction des connaissances et classification

Vers une modélisation booléenne des règles d'association

Abdelhak Mansoul et Baghdad Atmani,

Equipe de recherche SIF « Simulation, Intégration et Fouille de données »
Laboratoire d'Informatique d'Oran - LIO
Département Informatique, Faculté des Sciences, Université d'Oran
BP 1524, El M'Naouer, Es Senia, 31 000 Oran, Algérie
mans_abdel@yahoo.fr, atmani.baghdad@gmail.com

Résumé. L'extraction de règles d'association est une tâche populaire en fouille de données. L'une des premières méthodes utilisées est la méthode GUHA initiée par Hajek, Havel et Chytil [9], ensuite l'algorithme APRIORI, issu des travaux d'Agrawal, Imielinski, Swami et Srikant [6], [7]. Seulement ces méthodes produisent une trop grande masse de règles, ce qui induit un coût considérable en volume et en temps de traitement.
Dans cet article, nous proposons une extraction de règles d'association assistée par une modélisation booléenne des résultats obtenus, dans le but de parer aux inconvénients cités auparavant. Ceci, est dans la perspective de recherche d'un processus de génération de règles d'association par inférence, intégré dans un automate cellulaire.

Mots clés: Automate cellulaire, Fouille de données biologiques, Induction de règles, Règle d'association, modélisation booléenne.

1 Introduction

Parmi les travaux de recherche en fouille de données, l'extraction de règles d'association est sans doute la technique qui a attiré le plus l'attention des chercheurs et pour laquelle beaucoup de travaux ont été effectués [6], [7], [9], [10], [11]. Nous avons essayé de concevoir un processus assez novateur en se basant principalement sur le principe de représentation cellulaire des règles d'association afin de parer aux insuffisances liées à cette méthode de fouille de données.

Ce processus s'effectue en 3 étapes :

1. extraction de motifs fréquents et génération des règles d'association en utilisant l'algorithme Apriori [7];
2. modélisation booléenne des règles d'association par la machine CARI « Cellular Automaton for Rules Induction » ;
3. gestion des règles par le moteur d'inférence de CARI.

La structure des séquences biologiques. Les séquences biologiques expérimentales que nous utilisons ne sont pas dans leurs structures primaires à base de nucléotides

(ex : AAGTCGTTGCTGGC). Elles se présentent sous la forme de fichiers textes et dans des formats de données spécifiques (FASTA, STADEN, etc.) [2], [12] et contiennent des entités sémantiques (le gène, sa localisation, …..) (Fig. 1).

Nous exploiterons donc ce format pour définir un prétraitement spécifique et obtenir une structure bien appropriée à la fouille de données, où les entités sémantiques deviennent des descripteurs potentiels.

```
1: aac
 aminoglycoside 2-N-acetyltransferase [Mycobacterium tuberculosis CDC1551]
 Other Aliases: MT0275
 Annotation: NC_002755.2 (314424..314969, complement)
 GeneID: 923198
2: accD
 acetyl-CoA carboxylase, carboxyl transferase, beta subunit [Mycobacterium tuberculosis CDC1551]
 Other Aliases: MT0927
 Annotation: NC_002755.2 (1006705..1008192, complement)
 GeneID: 926242
……...
```

Fig. 1. Morceau d'une séquence génomique de la bactérie modèle [12].

Donc, la problématique abordée dans ce papier, est la recherche de règles d'association sur des données biologiques (séquences génomiques et protéiques), d'une bactérie modèle appelée : Mycobacterium Tuberculosis, avec post-traitement des résultats obtenus, par un automate cellulaire afin d'avoir une base de règles optimisée et des temps de traitements assez réduits.

2 Architecture du système BIODM

Notre système BIODM est composé de deux grands modules ERAB et CARI. Le module ERAB, acronyme d'Extraction de Règles d'Association à partir de données Biologiques, produit des règles d'association et les transmet au module cellulaire CARI pour générer et gérer les règles d'association booléennes.

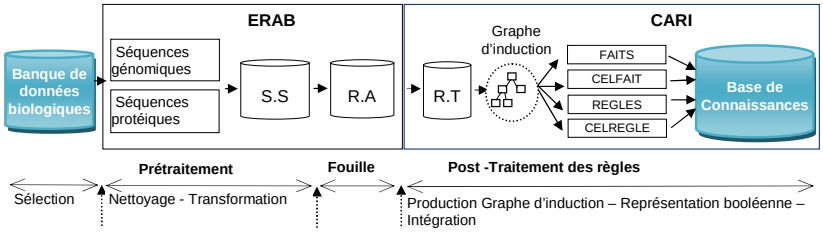


Fig. 2. Architecture générale du système BIODM (S.S : Séquences Structurées R.A : Règles d'Association R.T : Règles Transitoires).

2.1 Production des règles booléennes par la machine cellulaire CARI

CARI (Cellular Automaton for Rules Induction) est un automate cellulaire qui simule le principe de fonctionnement de base d'un moteur d'inférence classique en utilisant deux couches finies d'automates finis. La première couche, CELFAIT, pour la base des faits et la deuxième couche, CELREGLE, pour la base de règles. Chaque cellule au temps t+1 ne dépend que de l'état des ses voisines et du sien au temps t. Dans chaque couche, le contenu d'une cellule détermine si et comment elle participe à chaque étape d'inférence. A chaque étape, une cellule peut être active (1) ou passive (0), c'est-à-dire participe ou non à l'inférence. Le principe adopté est simple [1] :

- toute cellule i de la première couche CELFAIT est considérée comme fait établi si sa valeur est 1, sinon, elle est considérée comme fait à établir. Elle se présente sous trois états : état d'entrée (EF), état interne (IF) et état de sortie (SF) ;
- toute cellule j de la deuxième couche CELREGLE est considérée comme une règle candidate si sa valeur est 1, sinon, elle est considérée comme une règle qui ne doit pas participer à l'inférence. Elle se présente sous trois états : état d'entrée (ER), état interne (IR) et état de sortie (SR). Les matrices d'incidence R_E et R_S représentent la relation entrée/sortie des faits et sont utilisées en chaînage avant et en chaînage arrière en inversant leur ordre.

La dynamique de CARI pour simuler le fonctionnement d'un moteur d'inférence, utilise deux fonctions de transitions δ_{fact} et δ_{rule}, où δ_{fact} correspond à la phase d'évaluation, de sélection et de filtrage, et δ_{rule} correspond à la phase d'exécution.

- La fonction de transition δ_{fact} :

$$\delta_{fact}\,(EF, IF, SF, ER, IR, SR) = (EF, IF, EF, ER+(R_E^T.EF), IR, SR)\ ;$$

- La fonction de transition δ_{rule} :

$$\delta_{rule}\,(EF, IF, SF, ER, IR, SR) = (EF+(RS.ER), IF, SF, ER, IR, \hat{}ER)$$

où la matrice R_E^T désigne la transposée de R_E et ^ER désigne la négation du vecteur booléen ER.

2.2 Les étapes du processus adopté

Le processus de fouille de données adopté par notre système (ERAB + CARI) est composé de 4 étapes majeures :

1. sélection des données

A partir de la banque de données de NCBI [12], nous récupérons les séquences biologiques (table 1), qui vont servir à constituer la base de test expérimentale (table 2).

2. prétraitement

Un nettoyage et une transformation des données, du format original vers un formalisme adéquat, sont faits. Suivra alors une «binarisation», une opération nécessaire pour l'étape suivante.

3. fouille de données

Une recherche de règles d'association est faite par l'algorithme Apriori [7], avec calcul systématique du support et de la confiance pour ne retenir que les règles confiantes.

4. post-traitement des règles d'association

(a) transformation

Les règles d'association extraites sont transformées et représentées selon un formalisme transitoire aidant à la production d'un graphe d'induction. Ainsi la règle d'association R_i est traduite en une règle transitoire selon le principe suivant :

$$(R_i\ ,\ Antécédent_i\ ,\ Conséquent_i\ ,\ s,\ c) \rightarrow (R_i\ ,\ Prémisse_i\ (Antécédent_i\ )\ ,\ Conclusion_i\ (Conséquent_i\ )\ )\ ;$$

(b) production du graphe d'induction

Un algorithme utilisera en entrée R_i , les faits de $Premisse_i$ et de $Conclusion_i$, et en sortie il donnera un graphe d'induction où l'on aura : un sommet s_i qui désignera un nœud sur lequel on fait un test, avec des résultats possibles binaires ou à valeurs multiples ;

(c) modélisation booléenne

1. génération des règles cellulaires à partir du graphe d'induction sous la forme :

   $$R_i : Si\ \{\ Prémisse_i\ \}\ Alors\ \{\ Conclusion_i\ ,\ Sommet_i\ \}$$

   où $Prémisse_i$ est composée des items de $Antécédent_i$ et la $Conclusion_i$ est composée des items de $Conséquent_i$;

2. les règles générées (étape 4.c.1) sont représentées en couches cellulaires où :

   $$\{R_i\} \rightarrow \text{REGLES et } \{\ Prémisse_i\ ,\ Conclusion_i\ ,\ Sommet_i\ \} \rightarrow \text{FAITS}$$

3. intégration par la machine cellulaire, des règles générées, dans la base de connaissances pour les exploiter à travers différentes stratégies d'inférence.

La dynamique de la machine cellulaire est assurée par les deux fonctions de transition citées auparavant, δ_{fact} et δ_{rule} (2.1).

3 Exemple d'illustration de l'induction des règles cellulaires

Le processus général que notre système d'apprentissage applique à un échantillon est illustré par un exemple à partir de la 3e étape (2.2). Nous supposons avoir obtenu 2 règles d'association avec les gènes suivants : aceA-2, pstS-3, argC et phhB.

3e étape : fouille de données

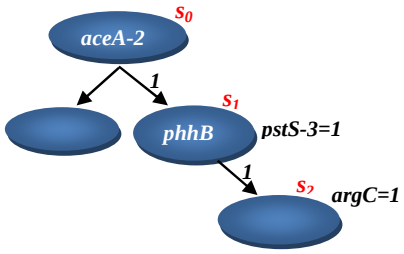
Règle	Antécédent	Conséquent	Support %	Confiance %
R1	aceA-2=1	pstS-3=1	45	77
R2	aceA-2=1, phhB=1	argC=1	45	70

4e étape : post-traitement des règles d'association

(a) transformation

Règle	Prémisse	Conclusion
R1	aceA-2=1	pstS-3=1
R2	aceA-2=1, phhB=1	argC=1

(b) production du graphe d'induction



(c) Modélisation booléenne

1. Génération des règles cellulaires à partir du graphe d'induction
   R1 : Si { s_0 } Alors { pstS-3=1, s_1 }
   R2 : Si { s_1 } Alors { argC=1, s_2 }

2. Représentation des règles en couches cellulaires
   Les règles booléennes produites R1 et R2 sont représentées par les couches CELFAIT (FAITS + CELFAIT) et CELREGLE (REGLES + CELREGLE) et les matrices d'entrée (R_E) et de sortie (R_S).

1. La couche CELFAIT

FAITS	CELFAIT		
	EF	IF	SF
s_0	1	0	0
pstS-3=1	0	1	0
s_1	0	0	0
argC=1	0	1	0
s_2	0	0	0

2. La couche CELREGLE

REGLES	CELREGLE		
	ER	IR	SR
R1	0	1	1
R2	0	1	1

3. La matrice d'incidence d'entrée R_E

R_E	R1	R2
s_0	**1**	0
pstS-3=1	0	0
s_1	0	**1**
argC=1	0	0
s_2	0	0

4. La matrice d'incidence de sortie R_S

R_S	R1	R2
s_0	0	0
pstS-3=1	**1**	0
s_1	**1**	0
argC=1	0	**1**
s_2	0	**1**

Fig. 3. Les couches cellulaires de l'automate cellulaire CARI.

3.1 La dynamique du moteur d'inférence cellulaire.

La dynamique de l'automate cellulaire CARI, pour simuler le fonctionnement d'un moteur d'inférence, utilise les deux fonctions de transitions $\boldsymbol{\delta_{fact}}$ et $\boldsymbol{\delta_{rule}}$, où $\boldsymbol{\delta_{fact}}$ correspond à la phase d'évaluation, de sélection et de filtrage, et $\boldsymbol{\delta_{rule}}$ correspond à la phase d'exécution.

1. Évaluation, sélection et filtrage (application de δ_{fact})
   $(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{fact}} (EF, IF, EF, ER + (R_E^T \cdot EF), IR, SR)$;
2. Exécution (application de δ_{rule})
   $(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{rule}} (EF + (RS.ER), IF, SF, ER, IR, \hat{}ER)$.

Nous montrons une simulation sur CELFAIT et CELREGLE de l'exemple illustré auparavant (Fig. 3), en considérant que $\mathbf{G_0}$ est la configuration initiale de l'automate cellulaire.

G_0

FAITS
s_0
pstS-3=1
s_1
argC=1
s_2

CELFAIT		
EF	IF	SF
1	0	0
0	1	0
0	0	0
0	1	0
0	0	0

REGLES
R1
R2

CELREGLE		
ER	IR	SR
0	1	1
0	1	1

1. Application de :

$$(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{fact}} (EF, IF, EF, ER + (R_E^T \cdot EF), IR, SR)$$

FAITS
s_0
pstS-3=1
s_1
argC=1
s_2

CELFAIT		
EF	IF	SF
1	0	**1**
0	1	0
0	0	0
0	1	0
0	0	0

REGLES
R1
R2

CELREGLE		
ER	IR	SR
1	1	1
0	1	1

2. Application de :

$$(EF, IF, SF, ER, IR, SR) \xrightarrow{\delta_{rule}} (EF + (RS.ER), IF, SF, ER, IR, \hat{}ER)$$

G_1

FAITS
s_0
pstS-3=1
s_1
argC=1
s_2

CELFAIT		
EF	IF	SF
1	0	1
1	1	0
1	0	0
0	1	0
0	0	0

REGLES
R1
R2

CELREGLE		
ER	IR	SR
1	1	**0**
0	1	1

3. Application de la fonction de transition globale : $\Delta = \delta_{rule} \circ \delta_{fact}$

La machine cellulaire passe de $\mathbf{G_1}$ à $\mathbf{G_2}$ avec $\Delta(G_1) = G_2$ si $G_1 \xrightarrow{\delta_{fact}} G'_1$ et $G'_1 \xrightarrow{\delta_{rule}} G_2$

$\mathbf{G_2}$

FAITS
s_0
pstS-3=1
s_1
argC=1
s_2

CELFAIT		
EF	IF	SF
1	0	1
1	1	1
1	0	1
1	1	0
1	0	0

REGLES
R1
R2

CELREGLE		
ER	IR	SR
1	1	0
1	1	**0**

Le cycle s'arrête car aucune règle n'est applicable.

4 Expérimentation

Pour examiner l'efficacité pratique de notre système, nous avons implémenté BIODM, et nous avons mené des tests expérimentaux sur une machine Intel Celeron CPU 540 à 186 GHz 512 Mo RAM, avec une base de test réelle et synthétique (table 2).

La base de test. Nous avons considéré les 4 souches échantillon du mycobacterium tuberculosis (table 1), et nous avons pris les 15 premiers gènes de chaque souche (table 2), avec la supposition que ces gènes soient assez représentatifs et distinctifs de chaque souche prise séparément.

Table 1. Base de données expérimentale [12].

N°	Souche	Nombre de Gènes
1	Mt CDC1551	4293
2	Mt F11	3998
3	Mt H37Ra	4084
4	Mt H37Rv	4048

Table 2. Base de test : échantillon de 15 gènes de chaque souche.

N°	Gènes
1	aac accD aceA-1 aceA-2 aceB aceE ackA acnA acp-1 acp-2 acpP acpS acs adh adk
2	aceE acpP acpS adk alaS alr argC argD argJ argS aroB aroE aroK aspS atpC
3	aac aao accA1 accA2 accA3 accD1 accD2 accD3 accD4 accD5 accD6 aceAa aceAb aceE acg
4	35kd_a aac aao accA1 accA2 accA3 accD1 accD2 accD3 accD4 accD5 accD6 aceAa aceAb aceE

4.1 Les résultats expérimentaux

Temps de traitement. En utilisant la base de test, le système BIODM donne des résultats intéressants et montre l'évolution exponentielle du temps d'exécution (table 3). Nous pouvons constater également que l'exécution de l'algorithme Apriori prend une part importante en temps d'exécution du système en totalité, i.e. dans ses phases les plus importantes de l'expérimentation à savoir : la génération des règles d'association par Apriori et la génération des règles booléennes par CARI.

Confiance %	Support %	Nombre de Gènes	Items générés	Nombre de règles	Temps d'exécution Apriori	Temps d'exécution global
10	70	15	41	69	0.00 s	0.00 s
30	50	15	41	147701	0.86 s	2.23 s
50	30	15	41	147687	0.86 s	2.23 s
70	10	15	41	835284	4.92 s	10.26 s

Table 3. Evolution du temps d'exécution (génération des règles d'association par Apriori).

Espace de stockage. Nous constatons qu'à titre indicatif pour un ensemble de 147687 règles d'association, le fichier occupe un espace de stockage de 8.65 MO alors que pour les règles cellulaires correspondantes, le fichier est de 6.13 MO.
Nous constatons qu'une représentation cellulaire est plus intéressante du point de vue espace de stockage et que sur un ensemble de règles encore plus conséquent, nous aurons un gain en espace de stockage assez significatif qui se répercutera positivement sur la performance du système, et mentionnons là aussi que ce ne sont que des résultats partiels qu'il faudra consolider avec un échantillon plus important.

Table 4. Evolution de l'espace de stockage.

Nombre de règles d'association produites	Espace de Stockage (règles d'association)	Espace de stockage (représentation booléenne)
69	1.78 KO	0. 81 KO
147701	8.65 MO	6.11 MO
147687	8.65 MO	6.12 MO
835284	48.8 MO	39.14 MO

5 Conclusion et perspectives

Dans ce papier, qui doit beaucoup aux travaux engagés dans le cadre des automates cellulaires [1], [5], [8], et après avoir évoqué les inconvénients des méthodes à base de règles en fouille de données, nous avons voulu utiliser une technique assez novatrice, dans la mesure où nous voulions adopter le principe des automates cellulaires.

Notre contribution est double : nous avons non seulement souhaité utiliser Apriori avec post-traitement des règles d'association, mais aussi exploiter les performances d'un moteur d'inférence cellulaire, dont nous exploiterons la méthode d'inférence qu'il utilise, dans un contexte de fouille de données.

De ce fait, deux objectifs nous ont guidés dans la proposition d'un automate cellulaire pour l'optimisation, la génération, la représentation et l'utilisation d'une base de règles d'association booléennes. Le premier, c'est d'avoir une base de règles optimisée et des temps de traitements assez réduits grâce à une modélisation cellulaires, et le deuxième c'est d'apporter une contribution à la construction des systèmes à base de connaissances en adoptant une nouvelle technique cellulaire.

Ainsi, les avantages de notre méthode basée sur la machine cellulaire CARI peuvent être récapitulés comme suit :

- un prétraitement simple et minimal de la base de règles d'association, pour sa transformation en matrice binaire selon le principe de couches cellulaires ;
- la facilité d'implémentation des fonctions de transitions δ_{fact} et δ_{rule} qui sont de basses complexités, efficaces et robustes et concernent des valeurs extrêmes et bien adaptées aux situations avec beaucoup de règles.

Tout ce travail s'inscrit dans la perspective de recherche d'un processus de fouille de données basée sur la génération de règles d'association par inférence, intégré dans un automate cellulaire.

Références

1. Atmani, B., Beldjilali, B.: Knowledge Discovery in Database : induction graph and cellular automaton. Computing and Informatics Journal, Vol. 26 N°2 171-197 (2007)
2. Carbonnelle, B., Dailloux, M., Lebrun, L., Maugein, J., Pernot, C.: Cahier de formation en biologie médicale N°29 (2003)

3. Guillaume, S.: Traitement des données volumineuses, mesures et algorithmes Évaluation et validation d'extraction de règles d'association et règles ordinales, Thèse de doctorat, Université de Nantes (2000)
4. Han, J., Kamber, M.: Data Mining : concepts and techniques. Morgan Kaufmann Publishers (2001)
5. Abdelouhab, F., Atmani, B.: Intégration automatique des données semi-structurées dans un entrepôt cellulaire, Troisième atelier sur les systèmes décisionnels, pp. 109-120. Mohammadia – Maroc 10 et 11 octobre (2008)
6. Agrawal, R., Imielinski, T., Swami, A.: Mining associations between sets of items in large databases, Proc. of the ACM SIGMOD Conf., Washington DC, USA (1993)
7. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), pp 487-499, Santiago, Chile (1994)
8. Benamina, B., Atmani, B.: WCSS : un système cellulaire d'extraction et de gestion des connaissances, Troisième atelier sur les systèmes décisionnels, 10 et 11 octobre 2008, Mohammadia – Maroc, pp. 223-234 (2008)
9. Hajek, P., Havel, I., Chytil, M.: The GUHA method of automatic hypotheses determination,Computing 1, pp. 293-308 (1966)
10. Hipp, J., Guntzer, U., Gholamreza, N.: Algorithms for association rule mining - a general survey and comparison. SIGKDD Explorations, vol. 2, 1, pp. 58-64 (2000)
11. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of associations rules. In Proceedings of 1997 ACM SIGMOD Int'l Conference on KDD and Data Mining, KDD'97, Newport Beach, Californie (1997)
12. National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov

Classification des images des dattes par SVM : contribution à l'amélioration du processus de tri

Djeffal Abdelhamid[1], Regueb Salah[1], Babahenini Mohamed Chaouki[1],Taleb Ahmed Abdelmalik[2],

[1]Département d'Informatique, Laboratoire LESIA, Université Mohamed Khider BP 145, Biskra, Algérie
[2]IUT GE2I, Laboratoire LAMIH, Université de Valenciennes, France
Abdelhamid_Djeffal@yahoo.fr, chaouki.babahenini@gmail.com, Abdelmalik.Taleb-Ahmed@univ-valenciennes.fr

Résumé. Nous proposons, dans ce travail, un système de classification automatique des dattes basé sur la technique d'apprentissage statistique dite Support Vector Machines. Ce système est utilisé pour le tri automatique des dattes dans les usines de conditionnement des dattes dans la région de Biskra. Le système analyse les images des dattes acquises par une caméra et construit une base de caractéristiques. La méthode SVM utilise cette base pour rechercher des hyperplans séparant les différents types de dattes dans l'espace de caractéristiques. Ces hyperplans serviront comme moyen pour classer les nouvelles images. Les Résultats obtenus démontrent la puissance de la méthode SVM pour la classification des images des dattes et valident le système proposé.

Mots clés : Classification d'image, sélection des dattes, support vector machine.

1. Introduction

Dans la région de Biskra située au sud est de l'Algérie, et connue par sa production des dattes, plusieurs usines souffrent du problème de la sélection manuelle des dattes, qui est très lente, marquée de son imprécision, et son coût élevé, ce qui influe sur la qualité du produit final. Les dattes sont collectées chaque automne des palmeraies et transférées vers les usines pour leur tri et emballage afin de les préparer à la vente dans le marché local ou leur exportation vers l'Europe. La sélection des dattes, une fois automatisée, peut contribuer efficacement dans l'amélioration de leur production en augmentant la vitesse de leur préparation et la qualité du produit final fourni au consommateur. L'objectif de notre travail est d'automatiser cette tache en se basant sur un système d'imagerie numérique utilisant la classification supervisée. Une étude minutieuse en se basant sur des recherches menés par des centres de recherche en agronomie de la région, et avec la collaboration des industriels et des agriculteurs concernés, nous a permis de déterminer les caractéristiques visuelles les plus importantes des dattes qui peuvent être utilisées pour leur classification. Dans le système proposé, les caractéristiques visuelles d'une datte sont extraites de son image

en utilisant des techniques de segmentation connues [4, 5]. Ces caractéristiques sont enregistrées dans une matrice avec leurs classes correspondantes afin de les utiliser pour apprendre un modèle de décision qui pourra guider la sélection des nouvelles images. L'apprentissage est effectué par la méthode SVM, introduite au début des années 90 par Vladimir Vapnik et qui connaît jusqu'à nous jours un très grand succès dans la reconnaissance des formes [1, 3]. Elle repose sur une théorie solide d'apprentissage statistique qui vise à trouver des hyperplans séparant les données dans un espace approprié des caractéristiques. Ce papier présente une contribution pour l'optimisation du processus de sélection des dattes. Dans la littérature, uniquement des solutions purement mécaniques basées sur le poids des dattes sont proposées en Emirats et Arabie Saoudite [13, 14], dont les résultats sont très faibles. Des solutions proposées par Compac [15] pour le tri d'autres produits tel que les pommes, les pommes de terres, les tomates… sont basées la couleur, le volume et les taches donnés par l'utilisateur. Le système proposé utilise l'apprentissage des qualités à partir des échantillons choisis par un expert pour construire des modèles de décision. Les résultats en termes de taux de reconnaissance sont satisfaisants en les comparants aux taux obtenus dans d'autres applications de classification d'images [4, 11].

2. Schéma général du système

Le système que nous proposons se base sur les éléments suivants (Fig 1):

- Un convoyeur qui permet de faire défiler les dattes sous une caméra numérique.
- Une caméra numérique qui prend les images des dattes et les envoie vers l'ordinateur.
- Un ordinateur qui reçoit les images, les analyse et prend la décision pour commander un système d'aiguillage,
- Un système d'aiguillage à la fin du convoyeur permettant son orientation vers une destination selon la décision prise par l'ordinateur.

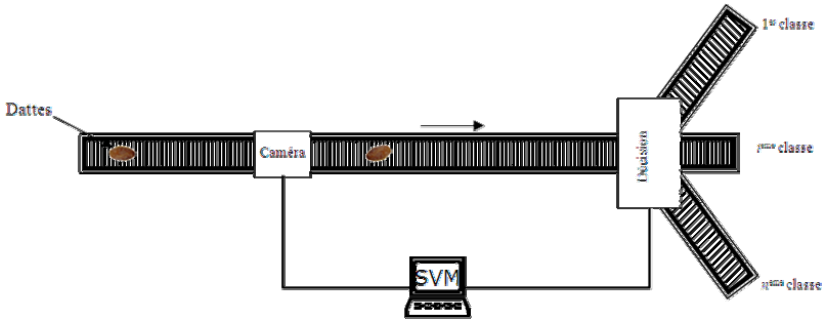


Fig 1. Schéma général du système

La partie convoyeur, caméra et système d'aiguillage n'est pas traitéé en détail dans ce travail et elle peut être l'objet de nouvelles recherches, autant plus, plusieurs systèmes

similaires existent en industrie et peuvent être utilisés [14]. Dans ce travail, on s'intéresse, plus, au problème de classification des dattes et l'apprentissage, c'est-à-dire depuis l'entrée de l'image au logiciel sur l'ordinateur jusqu'à trouver une décision par ce logiciel sur l'ordinateur.

En effet le logiciel est composé de deux parties : partie d'apprentissage et partie de sélection (d'utilisation). L'étape d'extraction des caractéristiques est commune entre les deux parties :

Le système proposé capture l'image d'une datte pour l'utiliser, selon le cas, dans l'un des deux modes : apprentissage ou classification. En mode apprentissage, le logiciel reçoit les images d'une classe donnée l'une après l'autre, chaque image est traitée et ses caractéristiques essentielles sont extraites (Fig 2) et stockées dans une table de vecteurs avec le libellé de la classe. Une fois l'acquisition des images de toutes les classes terminée, le logiciel utilise la méthode SVM pour trouver un modèle de décision qui permette de bien distinguer les types les uns des autres et enregistre ce modèle pour l'utiliser lors de la sélection. En mode sélection, les caractéristiques de l'image en question sont extraites puis exposées au modèle utilisé pour déterminer son type. Le type détecté est utilisé pour commander le système d'aiguillage afin d'orienter la datte vers la bonne direction c'est-à-dire la classer dans la bonne classe.

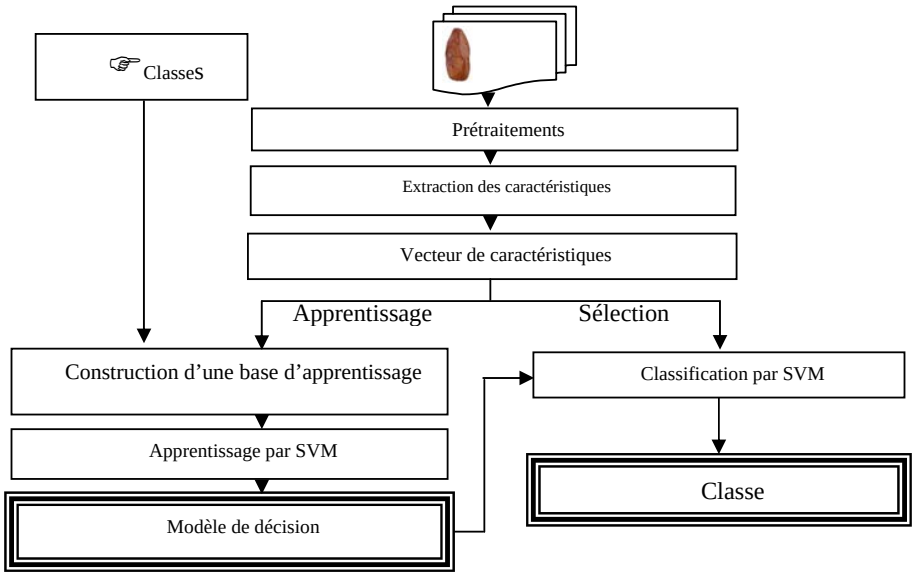


Fig 2. Etapes du logiciel

3. Extraction des caractéristiques

On commence, dans la phase d'apprentissage, par le traitement des images, l'extraction de leurs caractéristiques, puis l'enregistrement de ces caractéristiques avec les classes correspondantes dans une base de données. Dans ce travail, nous avons utilisé des images de dattes prises dans une usine de la région. Les images sont converties en niveaux de gris et filtrées pour éliminer les bruits éventuels dus à l'environnement d'acquisition [9, 5]. Une simple reconnaissance de forme est réalisée dans le but de reconnaître la datte et la distinguer de l'arrière plan. Une fois la datte localisée les différentes caractéristiques sont calculées. En se basant sur des recherches de l'INRAA [12] et selon la norme CEE-ONU DDP-08 concernant la commercialisation et le contrôle de la qualité commerciale des dattes entières et des interviews avec des concernés du domaine (agriculteurs, agronomes) de la région de Biskra, nous avons pu conclure que les caractéristiques les plus importantes qui permettent de distinguer le type d'une datte d'un autre sont :

1. Le calibre : représenté par le volume, la largeur et la longueur de la datte. On calcule premièrement le centre de gravité de la datte dans l'image et ses deux axes longitudinal et transversal, puis on calcule sa longueur et largeur mesurées en nombre de pixels puis on calcule son volume en nombre de pixels constituant la datte,
2. La couleur : représentée par la couleur moyenne des pixels de la datte,
3. L'homogénéité de la datte: représenté par le pourcentage des tâches calculé à base du nombre de pixels s'écartant plus de deux écarts type de la couleur moyenne.

Les informations extraites représentent un vecteur de caractéristiques de la datte (Fig 3).

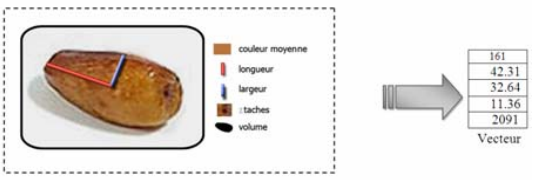


Fig 3. Extraction des vecteurs de caractéristiques

Support vector machines (SVM) [3, 6, 8, 10]

Parmi les méthodes à noyaux, inspirées de la théorie statistique de l'apprentissage de Vladimir Vapnik, les SVM constituent la forme la plus connue. SVM est une méthode de classification binaire par apprentissage supervisé, son but est de trouver un classificateur qui sépare les données d'apprentissage et maximiser la distance entre deux classes.

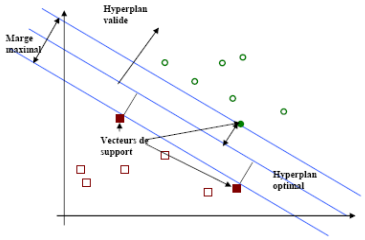


Fig 4. Exemple de marge maximale

Dans un espace de n attributs (dimensions) des données, le séparateur recherché est appelé hyperplan. Dans le schéma de la figure 4, on détermine un hyperplan qui sépare les deux ensembles de points (données). Les points les plus proches, qui seuls sont utilisés pour la détermination de l'hyperplan, sont appelés vecteurs de support. Il est évident qu'il existe une multitude d'hyperplans valides mais la propriété remarquable des SVM est que cet hyperplan doit être le plus loin possible des vecteurs supports. Le choix doit, donc, maximiser la « marge » entre l'hyperplan et les exemples d'apprentissage. Ce qui représente un problème de programmation quadratique convexe à contraintes linéaires et qui peut être résolu en introduisant les multiplicateurs de Lagrange :

$$\begin{cases} \max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{ij} \alpha_i \alpha_j y_i y_j x_i \cdot x_j \\ \forall i, 0 \le \alpha_i \le c; \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases} \quad d'où \qquad (1)$$

$$\begin{cases} \vec{w} = \sum_{i=1}^{i=N} \alpha_i y_i \vec{x_i} \\ \sum_{i=1}^{i=N} \alpha_i y_i = 0 \end{cases} \quad et \quad f(x) = Sgn\left(\sum_{i=1} \alpha^*_i y_i x_i \cdot x + b^* \right)$$

4.1. Cas non linéairement séparable :

Souvent, les données d'apprentissage ne sont pas linéairement séparables (Fig 5) c'est-à-dire qu'un hyperplan séparateur n'existe pas. Dans ce cas la méthode SVM fait recours à un changement d'espace pour aller à un nouvel espace où les données son linéairement séparables :

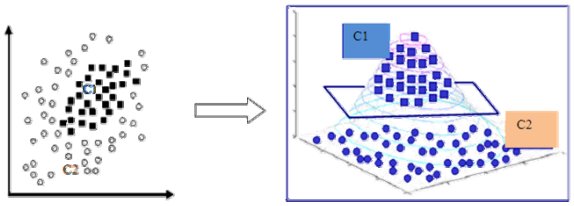


Fig 5. Changement d'espace dans le cas non linéairement séparable

Dans ce cas la fonction de décision devient :

$$f(x) = Sgn\left(\sum_{i=1}^{n} \alpha_i^* y_i \phi(x_i) \cdot \phi(x) + b^*\right) \qquad (2)$$

Et le problème et sa solution ne dépendent que du produit scalaire <ϕ(xi) , ϕ(x)> où ϕ représente la transformation d'espace utilisée. Pour faciliter les calculs, au lieu de choisir la transformation non-linéaire ϕ, on choisit une fonction réelle k(xi, x) appelée fonction noyau. Lorsque k est bien choisie, on n'a pas besoin de calculer la représentation des exemples dans cet espace pour calculer ϕ. Plusieurs noyaux sont utilisés dans la littérature tel que le noyau linéaire $k(x, x') = \langle x, x' \rangle$, le noyau Gaussien $k(x, x') = Exp\{(-||x - x'||^2) / 2\sigma^2\}$, … etc.

4.2 SVM multiclasses [7, 1]

La méthode SVM qu'on a vu jusqu'à maintenant ne concerne que le cas bi-classes. Dans le cas où les données appartiennent à plusieurs classes (>2), deux solutions sont principalement utilisées:

- Une contre reste (1vsR): On calcule pour chaque classe un hyperplan la séparant des autres. Lors de la phase de sélection, on prend la classe maximisant la fonction de décision (Fig 6.a).
- Une contre une (1vs1): On calcule pour chaque classe les hyperplans la séparant de chaque autre classe. Dans la phase de sélection, on prend la classe qui maximise le nombre d'appartenances par rapport aux autres classes (Fig 6.b).

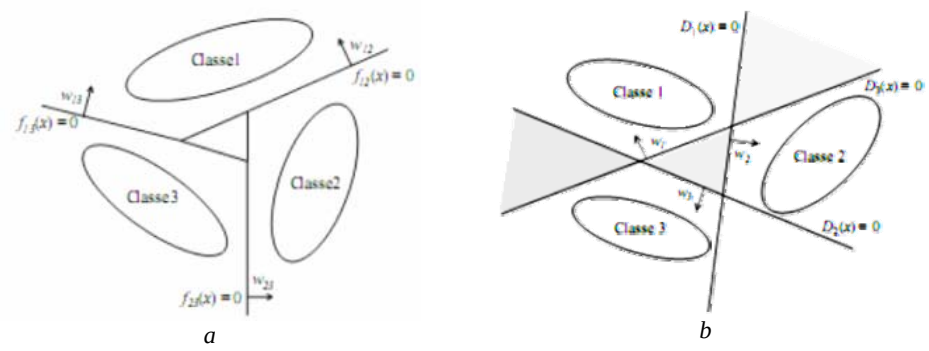


Fig 6. Méthodes: une contre une, une contre reste

5. Apprentissage et sélection

La méthode SVM travaille sur la base des caractéristiques et essaye de trouver un modèle de décision. L'algorithme SVM utilisé ici SMO [2] (Sequential Minimal Optimization) exige que la base utilisée soit normalisée entre -1 et +1, pour cela, il est nécessaire de convertir les valeurs de tous les vecteurs dans l'intervalle [-1, +1]. On calcule alors le maximum et le minimum de chaque attribut pour les enregistrer et les utiliser dans la phase de sélection. Le module SVM génère alors une fonction de décision pour chaque classe à partir de la base des caractéristiques normalisées. Dans la méthode une contre reste utilisée, on prend pour chaque classe la table normalisée et on met les valeurs de cette classe à +1 et toutes les autres à -1 puis on appelle le module SVM pour générer les paramètres de décision pour cette classe. Les paramètres générés par SVM pour chaque classe sont : les α_i différents de 0, les *bi* et les vecteurs des caractéristiques normalisés correspondants aux α_i différents de 0, c'est-à-dire les vecteurs supports. Le modèle de décision global obtenu contient en plus des paramètres des classes: les maximums et les minimums des cinq caractéristiques, le noyau utilisé, les paramètres du noyau et les libellés des classes. Dans la phase de sélection, on prend une image en entrée et on extrait ses caractéristiques de la même manière que lors de l'apprentissage, puis on applique, selon la méthode une contre reste, pour chaque classe du modèle la fonction de décision qui donne une valeur réelle : la datte appartient à la classe qui maximise la fonction de décision.

6. Tests et Résultats

6.1. Données utilisées

Pour tester notre système, nous avons pris des images d'une usine de la région de Biskra qui travaille sur six qualités différentes de dattes. Dans la table 1, sont présentées les classes utilisées pour les tests ainsi que le nombre d'échantillons et un exemple avec les valeurs du vecteur de caractéristiques extraits :

Table 1. Exemple de chaque qualité de dattes utilisées

Classe	*Nb Exmples*	*Exemple*	
		Image	*Vecteur (Lng, Lrg, Vol, Coul, Homog)*
Standard	**107**		**(34.06, 15, 1571, 4.77, 106)**
Fraza	**92**		**(29.15, 13, 1320, 7.21, 110)**
Petit fruit Standard	**44**		**(24.19, 12, 867, 6.8, 156)**
Petit fruit Fraza	**40**		**(21.59, 11.18, 684, 10.95, 95)**
Taché	**30**		**(28.35, 13, 1135, 19.11, 80)**
Boufarwa	**40**		**(32.56, 16.12, 1537, 7.89, 112)**

6.2. Choix des paramètres

Le premier paramètre à choisir est le noyau utilisé. Selon la plupart des références notamment [4, 11] le noyau Gaussien est celui le plus précis. Dans ce travail c'est le noyau utilisé, d'autres études seront menées pour comparer les différents noyaux dans le cas des images des dates.
Le noyau Gassien tel qu'il est définit utilise deux paramètres C et σ:

$$k(x, x') = e^{\frac{-\|x-x\|^2}{2\sigma^2}} \quad (3)$$

Ces paramètres sont choisis d'une façon empirique après plusieurs essais sur les échantillons de la table 1. Le résultat de ces essais est que les paramètres C et σ qui donnent les meilleurs résultats sont C=100 et σ=0.5. Les figures suivantes montrent graphiquement ce constat:

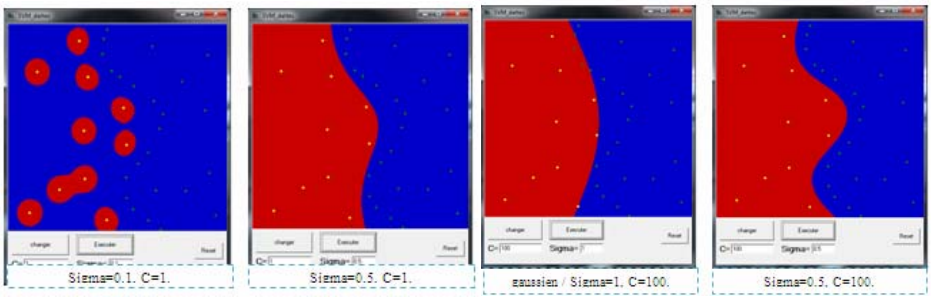


Fig 7. Choix des paramètres C et σ du noyau Gaussien

Après l'apprentissage du modèle on l'a testé sur les mêmes exemples utilisés par l'apprentissage le taux de reconnaissance était de 98.85%. Pour les nouveaux exemples qui n'appartiennent pas à la base d'apprentissage la matrice de confusion suivante montre les résultats obtenus sur dix images de chacune des six qualités déjà citées:

Table 2. Table de confusion des résultats obtenus

	Fraza	Petit St	Petit Fr	Boufarwa	Taché	Standard
Fraza	4	0	0	4	0	2
Petit St	0	7	3	0	0	0
Petit Fr	0	2	8	0	0	0
Boufarwa	5	0	0	3	1	1
Taché	2	0	0	0	7	1
Standard	1	0	0	2	1	6

Le tableau montre que dans la plupart des cas le taux est de 60 à 80%. Les cas de mis classifications remarquées (5 pour Boufarwa et 4 pour Fraza) sont dus à la grande ressemblance entre les deux classes. Ces résultats sont promoteurs en les comparants aux résultats obtenus par la sélection manuelle qui sont pratiquement faibles et très lents.

7. Conclusion et travaux futurs

Dans ce papier, une méthode de sélection automatique du fruit de dattes est proposée, elle se base sur la classification de leurs images par la méthode support vector machine. On a premièrement étudié les dattes et leurs caractéristiques visuelles qui peuvent être utilisées pour leur identification en se basant sur des recherches dans ce sujet. On a ensuite implémenté le module SVM et choisis d'une façon empirique ses

bons paramètres. Le système proposé peut améliorer le processus de sélection dans les usines de conditionnement des dattes. Dans la suite de ce travail, nous essayerons d'utiliser les images des différentes faces d'une datte pour construire son vecteur de caractéristiques plus précis ainsi que d'autres mesures tel que le poids. Nous essayerons aussi de comparer les deux méthodes 1v1 et 1vR de la méthode SVM et étudier l'utilisation d'autres noyaux. Enfin, Le système peut être utilisé aussi pour le tri d'autres fruits tel que les pommes, les oranges,...etc.

8. Bibliographie

1. D. Anguita, S. Ridella, and D. Sterpi. "A New Method for Multiclass Support Vector Machines". Proc. IEEE Int. Joint Conf. on Neural Networks, Budapest, Hungary, (2004).
2. J. C. Platt. "Fast training of support vector machines using sequential minimal optimization". In B.Schölkopf, C. J. C.Burges, and A.J. Smola, editors, Advances in Kernel Methods — Support Vector Learning, pages 185-208.Edition The MIT Press, (1999).
3. L.Wang, "Support Vector Machines: Theory and Applications", Springer 2005
4. Lei Wang, Xuchun Li, Ping Xue, and Kap Luk Chan, "A Novel Framework for SVM-based Image Retrieval on Larger Databases", 13th annual ACM International Conference on Multimedia (ACMMM), 2005.
5. Lei Zhang, Fuzong Lin, Bo Zhang, "Support vector machine learning for image retrieval", Tsinghua University, Beijing, 2001
6. N. Cristianini and J. Shawe-Taylor. "Introduction to Support Vector Machines and other kernel-based learning methods". Cambridge University Press, United Kingdom, (2000).
7. S. Har-Peled, D. Roth, and D. Zimak. "Constraint Classification for multiclass classification and ranking". Proc. Advances in Neural Information Processing Systems 15, pp. 785-792, (2003).
8. S.Abe, "Support Vector Machines for Pattern Classification", Springer 2005.
9. Simon Tong, Edward Chang,"Support vector machine active learning for image retrieval", Proceedings of the ninth ACM international conference on Multimedia, Vol 9, Pages: 107 – 118, 2001
10. V. N. Vapnik, "Statistical learning theory", Edition Wiley, New York, (1998).
11. Gidudu Anthony, Hulley Gregg and Marwala Tshilidzi, "Image Classification Using SVMs: One-against-One Vs One-against-All", Proccedings of the 28th Asian Conference on Remote Sensing, Nov 2007
12. S. ACOURENE, M. TAMA et B. TALEB, " Caractérisation, évaluation de la qualité de la datte et identification des cultivars rares de palmier dattier de la région des Zibans", Station INRAA Sidi-mehdi Touggourt, Algérie, 1997
13. www.kingdomdates.com
14. www.alriadh.com.sa
15. www.compacsort.com

Analyse de l'impact du changement : approche et étude de cas

M.K Abdi[*], H. Lounis[**]

**: Département d'Informatique, Université Es-Sénia d'Oran*
BP 1524, Oran, El M'naouer, Algérie
abdi.mustapha@univ-oran.dz

***: Département d'Informatique, Université du Québec à Montréal*
Case postale 8888, succursale Centre-ville, Montréal QC H3C 3P, Canada
lounis.hakim@uqam.ca

Résumé : Nous proposons dans cet article une approche et une démarche pour analyser et prédire les impacts des changements dans les systèmes à objets. La démarche que nous suivons consiste dans un premier temps, à choisir un modèle d'impact existant, pour ensuite l'adapter à notre contexte de travail. Une technique de calcul d'impact basée sur un méta-modèle est développée. Des données récoltées sur des systèmes réels sont envisagées pour étudier empiriquement des hypothèses de causalité entre d'une part, des attributs internes de logiciels, et d'autre part, l'impact de changement. Afin d'évaluer notre approche, une étude empirique a été menée sur un système réel dans laquelle une hypothèse de corrélation entre le couplage et l'impact de changement a été avancée. Un changement concret a été porté sur le système et des métriques de couplage ont été extraites de ce système. L'hypothèse a été étudiée par le biais de 3 techniques différentes d'apprentissage automatique. Les résultats ont montré que l'impact de changement peut être dépendant d'un type de couplage spécifique, e.g. celui d'importation.
Mots clés: Systèmes à objets, impact du changement, analyse, prédiction, métriques, apprentissage automatique.

1 Introduction

La modification des systèmes est une tâche à la fois difficile et porteuse de conséquence sur la suite de l'évolution de ces systèmes. Les effets des changements subis par le système doivent donc être pris en considération. La motivation de notre travail est d'améliorer la maintenance des systèmes orientés objet, et d'intervenir plus précisément dans la tâche de l'analyse de l'impact de changement. Nous visons principalement la réduction de l'effort ainsi que le coût de la maintenance. La réduction de cet effort peut être accomplie par la réduction du temps entre la proposition du changement, son implantation et sa réalisation, tout en assurant la qualité du système. L'effort peut être aussi réduit si on peut prédire le comportement du système face à d'éventuels changements. Notre travail se situe beaucoup plus dans cet axe de recherche. Plus l'analyse et la prédiction d'impact de changement est systématique, plus la réduction d'effort est à notre avis optimale. Ainsi les bonnes décisions peuvent être prises avant d'initier les changements. En identifiant l'impact potentiel d'une modification, on réduit le risque de s'embarquer dans des changements coûteux et imprévisibles. Plus un changement affecte de classes, plus le coût de sa réalisation est élevé. L'analyse des impacts de changement permettra ainsi d'évaluer le coût d'un changement et de faire un compromis entre les différents changements suggérés.

La section 2 résume l'état de l'art des différents travaux faits autour de l'analyse d'impact de changement. Notre proposition fait l'objet de la troisième section. Nous expliquons les points fondamentaux sur lesquels se basent notre approche ainsi que les étapes générales de la démarche adoptée. Ensuite, nous présentons le modèle d'impact de changement choisi pour mener nos travaux, suivi de son adaptation à Java. La technique de calcul d'expressions d'impact de changement basée sur une approche par méta-modèle (PTIDEJ) [GUÉ 03] finira cette section. La section 4 est réservée à l'étude empirique et à la discussion des résultats trouvés. Les perspectives de notre travail sont discutées en conclusion.

2 Travaux connexes

Plusieurs études ont été menées pour valider les métriques et les relier à certaines propriétés de la maintenabilité. Li et Henry [LI 93] ont pris cinq métriques de Chidamber & Kemerer [CHI 94] et ont ajouté trois métriques propres à eux pour montrer qu'il existe une forte relation entre ces métriques et l'effort de maintenance (exprimé par le nombre de lignes-codes changées). Lounis & al [LOU 97] ont proposé une suite de 24 métriques de code pour générer des modèles prédictifs liés à la propension d'erreurs (fault proneness) dans un système orienté objet. Enfin, dans [BRI 01], les auteurs ont étudié aussi les relations entre la majorité des métriques de couplage, de cohésion et d'héritage et la propension d'erreurs des classes de systèmes orientés objets.

Moins de travaux ont été menés concernant l'impact de changement. Dans [ANT 99], les auteurs ont prédit la taille des systèmes orientés objets évoluant dans le temps en se basant sur l'analyse des classes impactées par une demande de changement. Kung & al [KUN 95] intéressés par les tests de régression, ont développé un modèle d'impact de changements basé sur les trois liens : héritage, association et agrégation. Li et Offutt [LEE 96] et [LEE 98] pour examiner les effets d'encapsulation, d'héritage et de polymorphisme sur l'impact de changements, ont proposé des algorithmes pour calculer l'impact complet de changements faits dans une classe donnée. Dans [CAN 01], l'analyse d'impact a été faite avec l'objectif de réduire les coûts et la durée des tests de régression. L'analyse a été faite à partir d'un graphe de dépendance. Briand & al dans [BRI 99], ont essayé de voir si les mesures de couplage capturant toutes sortes de collaborations entre classes, peuvent aider à faire l'analyse d'impact de changement. La stratégie adoptée dans cette étude est différente des autres stratégies dans la mesure où elle est purement empirique. Dans [CHA 98] et [KAB 02], un modèle de changements et d'impact de changements, a été défini au niveau conceptuel pour étudier la changeabilité des systèmes à objets. Selon une perspective différente, Sahraoui & al. ont étudié dans [SAH 00], l'impact du refactoring sur la structure et donc sur les métriques structurelles. Cette étude a permis de déterminer quels sont les refactorings qui peuvent améliorer ou détériorer certaines propriétés structurelles.

En résumé, les études faites dans [BRI 99] et [WIL 99] sont des exemples d'approches purement empiriques. Les travaux [KUN 95], [LEE 96], [CAN 01] et [LEE 98] relèvent de la catégorie d'approches basées principalement sur des modèles qui sont des graphes de dépendance et éventuellement enrichis de certains formalismes. Les études [CHA 98] et [KAB 02] proposent un modèle différent. Nous en parlerons par la suite. Nous avons remarqué à travers cette synthèse qu'il y a plus de travaux à base de graphe de dépendance que de travaux à base d'autres modèles ou purement empiriques, ce qui explique qu'il vaudrait mieux à notre avis explorer d'autres voies de recherche. D'autre part, le plus souvent, l'impact n'est pas calculé d'une façon systématique et enfin, nous soulignons le fait qu'il y a eu beaucoup plus d'expérimentations sur des petits systèmes que sur des systèmes réels de taille industrielle, ce qui par conséquent, empêche la généralisation des résultats trouvés (règles, relations, lois, etc.). Dans la section suivante, nous présentons notre proposition en commençant par expliquer l'approche globale puis la démarche adoptée.

3 Proposition

3.1 Approche globale

Nous avons décidé dés le début que notre approche ne sera pas purement empirique pour la simple raison que nous visons des résultats (règles, relations de cause à effet, etc.) plus au moins généraux, ou à la limite qui peuvent s'appliquer à des domaines d'application larges. Cela ne réduit pas l'importance et la nécessité de la voie empirique dans notre approche; elle est à la fois analytique et expérimentale. L'étude des changements et de l'analyse de leurs impacts doivent se faire, à notre avis, d'abord à un niveau plus haut d'abstraction. Les résultats trouvés à ce niveau doivent être nécessairement testés et vérifiés par la suite par le biais d'études empiriques. Une remise en cause des modèles utilisés au niveau abstrait (conceptuel) est tout à fait possible dans le cas où les études empiriques n'aboutissent pas aux résultats proposés par les modèles, sinon une explication doit être donnée aux exceptions des résultats trouvés. Suite à notre revue de la littérature du domaine, nous avons remarqué qu'il y a très peu de travaux qui proposent un modèle d'impact de changements plus au moins complet, dans le sens où le modèle tient compte des principaux liens qu'on peut trouver dans une conception orientée objet (à savoir l'association, l'agrégation, l'invocation et l'héritage) [ANT 99].

Dans notre cas, nous avons repris le modèle d'impact défini dans le projet SPOOL[1] [SCH 01], [CHA 99] et [KAB 02] vu que ce modèle est l'un des plus généraux et surtout du fait qu'il permet de calculer l'impact d'une façon systématique, ce qui est à notre avis, un facteur important dans la réduction de l'effort ainsi que celle du coût de la maintenance. Le projet SPOOL avait comme objectif principal la compréhension des propriétés de conception des systèmes industriels et de leurs influences sur la maintenance et l'évolution de ces derniers.

Dans notre travail, nous utilisons ce modèle pour mener nos expérimentations. Ces dernières doivent être, à notre avis, orientées dans la mesure où elles doivent vérifier des hypothèses énoncées auparavant. Ces hypothèses manquent évidemment de preuve. Par analogie à d'autres travaux faits dans le domaine [LOU 97] et [BRI 01], les hypothèses sont en général des relations entre certaines caractéristiques de conception (ou propriétés architecturales : cohésion, couplage, etc.) d'un système et l'impact de changement dans notre cas. Ces caractéristiques de conception sont mesurées par des métriques. Le choix de ces métriques fait partie de l'orientation des études empiriques.

La vérification de ces hypothèses peut être faite de différentes manières, comme par exemple, par le biais de techniques à base de modèles statistiques [KAB 02]. Dans notre travail, nous allons opter pour des techniques de l'intelligence artificielle, plus précisément celles de l'apprentissage automatique (machine- learning) pour deux raisons principales. D'une part, ces techniques n'ont pas encore été utilisées dans des travaux antérieurs traitant de l'analyse d'impact de changement. D'autre part, le résultat de l'utilisation de ces techniques est un ensemble de connaissances représentées selon un formalisme, qui peut être exploité par un système de décision basé sur les connaissances. Enfin, nous signalons que comme nous visons les systèmes logiciels codés en Java, une opération d'adaptation du modèle utilisé (modèle défini au niveau conceptuel) à ce langage s'avère nécessaire afin de pouvoir calculer l'impact de tout changement atomique possible en Java. La section suivante résume les points essentiels de notre approche globale et détaille la démarche adoptée.

3.2 Démarche

Les principales actions à entreprendre dans le cadre de notre approche sont les suivantes :
1. Choisir un modèle défini au niveau conceptuel.
2. Adapter ce modèle à Java.
3. Prendre des systèmes réels et appliquer concrètement un ou des changements (au niveau du code).
4. Définir l'expression ou les expressions d'impact de changement.
5. Formuler des hypothèses de causalité reliant des caractéristiques internes des applications et l'impact de changement.
6. Dériver les métriques à partir des hypothèses formulées.
7. Vérifier les hypothèses par des techniques d'apprentissage automatique.

Notons que le retour depuis l'étape 7 aux trois premières étapes est tout à fait possible dans un but de remise en cause.

Afin de concrétiser cette démarche, nous avons besoin de systèmes réels de taille industrielle pour faire nos expérimentations. La diversité des domaines d'applications est souhaitable dans notre travail. Nous avons besoin aussi d'outils (efficaces) qui permettent d'analyser le code du système sous test. La programmation de l'expression (ou des expressions) calculant l'impact de changement déduit par le modèle au niveau conceptuel dépend d'une part du (des) changement (s) considéré (s), et d'autre part, de l'outil d'analyse utilisé. Le calcul des métriques choisies ou leur programmation en cas de nouvelles métriques est une tâche qui peut être réalisée dans le cadre de l'outil utilisé, et sera ainsi une partie intégrante de l'outil, comme elle peut lui être totalement indépendante. Enfin, le choix de techniques d'apprentissage automatique dans un but de vérification d'hypothèses dépend de leurs performances.

3.3 Modèle d'Impact de Changement

[1] SPOOL: "Spreading Desirable Properties into the design of Object-Oriented Large-scale software systems". Ce projet SPOOL a été organisé par CSER (Consortium for Software Engineering Research), et subventionné par BELL Canada, NSERC (Natural Sciences and Research Council of Canada) et NRC (National Research Council Canada).

3.3.1 Objectifs

Nous nous concentrons sur comment les divers liens entre des classes influencent en fait l'impact de changement. Cela permet de s'assurer que le système s'exécutera toujours correctement après que le changement soit mis en œuvre. Notre intérêt est concentré sur comment le système réagit à un changement (en général). Notons qu'un système absorbe facilement un changement si le nombre de composants impactés est petit.

3.3.2 Modèle Conceptuel

Un système est vu comme un ensemble de classes connectées par différents liens. Une classe est définie comme un groupe de méthodes qui servent comme interface publique ou pour des opérations internes, et une section de variables qui définissent l'état des instances de la classe.

3.3.2.1 Changements

Nous définissons un changement à un système comme un changement qui peut s'appliquer à un composant. Un composant se réfère à une classe, une méthode, ou bien une variable. Comme exemples de changement, on peut avoir la suppression d'une variable, le changement de la portée d'une méthode, de "public" à "protected", ou le déplacement du lien entre une classe et son parent.
La table 1 consigne les principaux changements aux systèmes orientés objets, au niveau conception. Ils sont définis puis classifiés selon le composant qu'ils affectent et un total de 13 changements est identifié.

Composant	***Description du Changement***
Variable	Changement de type de variable
	Changement de portée de variable
	Ajout de variable
	Suppression de variable
Méthode	Changement de type de retour de méthode
	Changement d'implémentation de méthode
	Changement de signature de méthode
	Changement de portée de méthode
	Ajout de méthode
	Suppression de méthode
Classe	Changement de structure d'héritage de classe
	Ajout de classe
	Suppression de classe

Table 1. *Principaux changements au niveau conceptuel*

3.3.2.2 Liens

Une fois qu'un composant donné est soumis à un changement, une partie spécifique peut être affectée, dans le cas où elle est liée au composant changé via un lien. Ces liens sont parmi les quatre types suivants :
S (association) : une classe fait référence aux variables d'une autre classe, **G** (agrégation) : la définition d'une classe implique des objets d'une autre classe, **H** (héritage) : une classe hérite les particularités définies dans une autre classe parente),
I (invocation) : les méthodes d'une classe invoquent des méthodes définies dans une autre classe.

Nous considérons aussi une notation spéciale généralement utilisée dans l'algèbre booléenne : L'absence d'un opérateur entre 2 liens signifie une ***intersection***. L'opérateur "**+**" signifie une ***union***. L'opérateur "**~**" avant un lien signifie la ***négation***, *c'est-à-dire l'*ensemble des classes non associées par ce lien spécial, par exemple, **~G** signifie l'ensemble des classes qui ne sont pas liées à la classe indiquée par le lien d'agrégation. Les liens sont indépendants les uns des autres et nous pouvons trouver n'importe quel nombre et type de liens entre deux classes. Un changement d'une classe peut aussi avoir un impact dans la même classe. Le pseudo-lien **L** (local) est introduit pour exprimer ceci.

3.3.2.3 Impact

Nous appelons impact d'un changement l'ensemble des classes qui exigent une correction suite à ce changement. Il dépend de deux facteurs : l'un est le type de changement. Par exemple, un changement de type de variable a un impact sur toutes les classes faisant référence à cette variable, tandis que l'ajout d'une variable n'a aucun impact sur ces classes. Étant donné un type de changement, l'autre facteur est la nature des liens impliqués. Si, par exemple, la portée d'une méthode est changée de "public" à "protected", les classes qui invoquent la méthode seront impactées, à l'exception des classes dérivées. Nous notons que plus qu'un type de lien entre la classe changée et une classe impactée peut être impliqué dans le calcul de l'impact. Ainsi, pour un changement donné ch_i dans la classe cl_j, l'ensemble des classes impactées est exprimé par une expression booléenne dans laquelle les variables représentent les liens. Par exemple, la formule d'impact pour un changement hypothétique peut être donnée par :

$$\text{Impact } (cl_j, ch_i) = \mathbf{S\sim H+G}$$

Cette expression signifie que les classes qui sont en association (S) avec la classe changée cl_j et non dérivées (~H) de cette classe, ou les classes qui sont en agrégation (G) avec cl_j : sont impactées.
Dans notre travail, nous nous intéressons seulement aux changements qui ont un impact syntaxique. Un changement donné est caractérisé par une transformation du code quelque part dans le système. Si le système est recompilé avec succès, alors il n'y a aucun impact. Sinon, nous sommes face à un impact, c'est-à-dire, des modifications du code qui doivent être faites ailleurs dans le système pour obtenir un code syntaxiquement correct qui se recompilera.

3.4 Adaptation du modèle à Java

Nous signalons que ce modèle défini au niveau conceptuel a été déjà adapté à C++. Cela représentait une contrainte du partenaire industriel du projet SPOOL [CHA 98], [CHA 99] et [SCH 01], projet dans lequel a été défini ce modèle d'impact. Comme dans notre travail, nous visons les systèmes logiciels codés en Java, une opération d'adaptation de ce modèle à ce langage s'avère nécessaire. Nous avons examiné l'adaptation déjà faite dans [CHA 98] et [KAB 02]. Nous avons remarqué qu'il y a certains changements communs aux deux langages. Nous citons comme exemples, le changement de type de variable, changement de signature de méthode, changement de la structure d'héritage d'une classe, etc. Par contre, il y a d'autres changements qui sont propres au langage C++. Ces derniers concernent principalement les concepts de "virtual" (méthode virtuelle ou classe virtuelle) et "friendship" (classe amie). Le concept de "virtual" est introduit en C++ pour gérer les appels dynamiques. Le rôle joué par ce concept en C++ est assuré en Java par le biais de sa machine virtuelle. Le concept d'amitié (friendship) n'existe pas en Java. En résumé, les changements propres à C++ viennent des concepts qui lui sont propres. Nous tenons à souligner à ce niveau, que notre intérêt dans cette étude est de voir les changements qui peuvent être appliqués en Java. Donc, mis à part les changements touchant à ces deux concepts, le reste des changements raffinés en C++ sont tout à fait possible dans le langage Java. La liste finale contient un total de 52 changements, comprenant 12 changements pour la variable, 25 pour la méthode et 15 pour la classe. Dans la section suivante, nous parlons de l'outil utilisé ainsi que de son extension pour répondre à notre objectif de calcul d'impact.

3.5 Outil utilisé : PTIDEJ

Pour nos expérimentations, nous avons opté pour PTIDEJ[2] [GUÉ 03] afin d'analyser le code des programmes (systèmes) considérés. Guéhéneuc propose et décrit dans sa thèse des modèles et des algorithmes pour garantir la traçabilité des motifs de conception[3] entre les phases d'implantation et de rétroconception des programmes. Il fait cela par l'identification semi-automatique de micro-architectures similaires à ces motifs dans le code source des programmes. Ces modèles et ces algorithmes forment un cadre pour la traçabilité des motifs de conception. La suite d'outils PTIDEJ est une implantation en Java de ces modèles et algorithmes. Elle est intégrée à l'environnement de développement (*EDI*) Eclipse [OTI 01]. Elle permet la modélisation des programmes Java, l'identification et la traçabilité des relations

[2] Ptidej : Pattern Trace Identification, Detection, and Enhancement in Java.

[3] Un motif de conception est la solution d'un patron de conception.

interclasses (association, agrégation et composition) et des motifs de conception entre les phases d'implantation et de rétroconception.

Elle inclut le métamodèle PADL[4], dérivé du métamodèle PDL[5] défini dans une autre thèse [ALB 03], pour modéliser les relations interclasses, les motifs de conception et les programmes Java. Le métamodèle PADL dispose d'un ensemble de constituants nécessaires à la description des modèles d'un programme aux niveaux implémentation, idiomatique[6] et conception. Le métamodèle PADL est utilisé aussi pour modéliser un motif de conception au niveau idiomatique. Cette modélisation consiste à décrire ses participants et leurs relations avec les constituants du métamodèle PADL, et fournit par la suite un modèle abstrait du motif. L'implantation du patron de conception Visiteur intègre l'interface Walker qui joue le rôle de visiteur. Cela permet de parcourir tous les constituants d'un modèle de programme exprimé avec PADL. Ce mécanisme de visiteur est utilisé par exemple pour calculer des métriques sur les modèles de programmes, ou générer des informations comme la liste globale des entités du modèle. Pour plus de détails sur l'outil PTIDEJ et ses métamodèles, nous orientons le lecteur vers [GUÉ 03] et [ALB 03].

En ce qui nous concerne, notre intervention réside à ce niveau du modèle PADL pour l'étendre afin qu'il puisse répondre à nos besoins de calcul d'expressions d'impact de changements. Nous avons implémenté, pour le moment, les classes permettant de déduire les impacts de changements dont l'expression résultat fait appel aux liens d'agrégation, d'association et d'héritage. Le lien d'invocation (de méthodes) n'a pas été pris en considération dans notre travail pour la simple raison que la version du méta-modèle utilisée ne l'offrait pas. Nous signalons que nous prenons en charge ce lien dans une perspective à court terme.

4. Étude empirique

4.1. Objectif

Comme les liens inter-classes sont censés être plus responsables de la propagation de la modification que les liens intra-classe, nous allons nous concentrer sur la propriété de couplage, et voir s'il y a des relations de cause à effet entre cette propriété architecturale du système et l'impact de changement. Nous proposons l'hypothèse suivante : *"Le couplage influence l'impact de changement dans un système objets"* Nous avons cité dans la section 2 plusieurs travaux autour de cette propriété architecturale mais l'objectif dans cette expérimentation est de voir quels types de couplage influence le plus l'impact de changement.

4.2 Système considéré

Nous avons choisi pour cette étude empirique, un système disponible à notre niveau. Il s'agit de BOAP (Boîte à Outils pour l'Analyse de Programmes) développé au Centre de Recherche Informatique de Montréal (CRIM) [ELH 02]. C'est un ensemble d'outils logiciels intégrés, qui permet à un expert d'évaluer rapidement le niveau de qualité d'un logiciel (faiblesses conceptuelles ou structurelles, instructions trop complexes, etc.). Le système BOAP (version 1.1.0) que nous avons considéré contient en tout 394 classes.

4.3. Changements considérés et métriques sélectionnées

Nous avons choisi comme changement : le changement de type de variable. A titre d'exemple, nous avons déterminé par le biais de la technique de calcul élaborée une classe qui présente un nombre important de liens d'associations avec les autres classes pour avoir un impact assez considérable sur le reste du système selon le changement envisagé. Ensuite, nous avons sélectionné une variable puis porté notre changement. Nous obtenons:

Classe considérée : dbClass (du package : DBLMR)

Variable choisie : sizeInBytes

[4] PADL : Pattern and Abstract-level Description Language.

[5] PDL : Pattern Description Language.

[6] Niveau idiomatique : niveau d'abstraction défini entre les niveaux implémentation et conception.

Changement : de type "**long**" au type "**integer**"
L'expression d'impact de ce changement est : S + L
Ce qui signifie qu'il y a impact de changement localement d'abord (dans la classe changée elle même) et aussi dans toutes les classes du système qui sont en association avec la classe changée "dbClass".
La technique de calcul d'impact de changement nous retourne un résultat de : 42 classes. Il y a donc 42 classes impactées suite à ce changement.
Nous avons procédé de la même façon pour le reste des classes du système. Notons qu'il peut s'agir d'autres changements de types de variables (pas nécessairement de "**long**" à "**integer**"), comme il est tout à fait possible qu'un changement ne crée aucun impact (impact nul).
Ensuite, nous avons extrait de notre système de test BOAP un ensemble de métriques reliées toutes à la propriété de couplage. Elles sont présentées dans la table 2. Nous les avons calculées par le biais de l'outil développé dans [CHE 04].

Métriques	Définition
RFC	Response For a Class : nombre de méthodes invoquées en réponse à un message.
MPC	Message Passing Coupling : nombre de messages envoyés par une classe en direction des autres classes du système.
CBOU	CBO Using : se réfère aux classes utilisées par la classe cible.
CBOIUB	CBO Is Used By : se réfère aux classes utilisant la classe cible.
CBO	Coupling Between Object : nombre de classes avec lesquelles une classe est couplée
CBONA	CBO No Ancestors : CBO sans considérer les classes ancêtres.
AMMIC	Ancestors Method–Method Import Coupling : nombre de classes parentes avec lesquelles une classe a une interaction de type méthode-méthode et un couplage de type IC.
OMMIC	Others Method–Method Import Coupling : nombre de classes (autres que les super-classes et les sous-classes) avec lesquelles une classe a une interaction de type méthode-méthode et un couplage de type IC.
DMMEC	Descendants Method – Method Export Coupling : nombre de sous-classes avec lesquelles une classe a une interaction de type méthode-méthode et un couplage de type EC.
OMMEC	Others Method – Method Export Coupling : nombre de classes (autres que les superclasses et les sous-classes) avec lesquelles une classe a une interaction de type méthode-méthode et un couplage de type EC.

Table 2. *Métriques sélectionnées*

4.4. Étude de l'hypothèse

Comme déjà signalé, nous abordons cette étude par le biais de techniques d'apprentissage automatique. Nous avons fait appel à l'environnement Weka (Waikato Environment for Knowledge Analysis) [WIT 00] pour atteindre cet objectif. Weka est un ensemble d'outils permettant de manipuler et d'analyser des fichiers de données, implémentant la plupart des algorithmes d'apprentissage automatique, dont les arbres de décision et les réseaux de neurones. Il est écrit en java, est "open source" et disponible sur le web[7]. Nous avons voulu lors de cette expérimentation, utiliser plusieurs algorithmes d'apprentissage (J48, PART et NBTree) afin de trouver diverses relations de cause à effets entre les métriques de couplage et l'impact de changement. Dans ce travail, le choix de ces algorithmes a été basé sur trois critères, à savoir la facilité de l'interprétation des modèles trouvés, la complémentarité et la précision des résultats. Rappelons que notre système contient 394 classes, que l'impact dans notre cas est quantifié par le nombre de classes impactées, et que suite à un changement (changement de type de variable) toutes les classes où il y a eu propagation de modification, ont été considérées, même s'il ne s'agit que d'impact local (ou encore impact ≥ 1). Nos données d'apprentissage regroupent 11 variables (10 variables indépendantes + la variable dépendante). Les variables indépendantes représentent les métriques de couplage que nous avons extrait du système testé. La variable dépendante représente l'impact de changement. Toutes les variables indépendantes sont numériques. Par contre, la variable dépendante est nominale. Initialement, celle-ci était numérique parce que nous l'avons calculé par le biais de notre technique de calcul, et elle est le résultat de l'expression d'impact S+L (voir section 4.3). Il était

[7] www.cs.waikato.ac.nz/ml/weka

nécessaire de la transformer en variable nominale pour pouvoir utiliser efficacement les 3 algorithmes, en particulier J48. Pour cela, nous avons divisé l'ensemble des valeurs d'impact en 5 tranches, chacune correspondante à une valeur nominale d'impact, variant de "très-faible" à la valeur " très-fort".

4.5 Résultats et discussion

J48

J48 est une implémentation de l'algorithme bien connu C4.5 [QUI 93]. C'est un algorithme d'apprentissage supervisé qui induit un modèle de classification sous la forme d'un arbre de décision ou de règles, et ce, à partir d'un ensemble d'exemples. L'étape clé de ce type d'algorithmes est le choix du "meilleur" attribut à tester afin d'obtenir des arbres de décision compacts et possédant la meilleure capacité prédictive possible. Des heuristiques basées sur la notion d'entropie ont montré leur efficacité pour réaliser ce type de choix. Lors de l'exécution de cet algorithme sur notre ensemble de données, nous avons choisi le mode de test validation croisée (cross-validation). C'est une technique dans laquelle l'ensemble des données disponibles est divisé en N blocs. Elle consiste à créer un modèle (apprendre une hypothèse) sur N-1 blocs, puis à tester ce modèle sur le bloc restant. L'algorithme refait la même chose pour chacun des N blocs de données; ainsi l'opération entière est faite N fois. Le taux de succès obtenu (73,85%) est assez intéressant, c'est-à-dire que sur 394 instances, 291 ont été correctement classifiées. Par contre, nous trouvons que l'arbre de décision généré est trop grand (taille : nombre de nœuds=67). Par conséquent, il est difficile de tirer des règles de causalité depuis cet arbre.

Dans un objectif d'avoir des résultats plus clairs, notamment un arbre de décision plus compact, nous avons opté pour un prétraitement des données. Nous avons ainsi retenu un ensemble réduit d'attributs (ou variables indépendantes), les plus pertinents, au lieu de considérer l'ensemble de tous les attributs. Pour cela, nous avons sélectionné cet ensemble réduit d'attributs par le biais de l'algorithme "CfsSubsetEval" dans l'environnement Weka toujours, avec la méthode de recherche "BestFirst", paramétrée en recherche descendante. Il s'agit d'un algorithme simple de filtrage qui range des sous-ensembles d'attributs selon une corrélation basée sur une fonction heuristique d'évaluation. Les attributs non pertinents devraient être ignorés parce qu'ils auront une faible corrélation avec la variable à prédire. Les attributs qui ont été sélectionnés sont : MPC, CBOU, CBONA, AMMIC, et OMMIC (voir table 2). Nous avons re-exécuté l'algorithme J48 sur notre ensemble de données ainsi réduit. Le taux de succès obtenu (73,30 %) est très proche du précèdent. Par contre, l'arbre de décision obtenu est bien plus compact (nombre de nœuds = 31). Il contient 16 feuilles. Chaque chemin de la racine à une feuille donnée est une règle de causalité. Nous avons donc un ensemble de 16 règles. La figure 1 présente certaines règles choisies de cet ensemble.

Règle 1 : *MPC ≤ 21*
OMMIC ≤ 4
AMMIC = 0
→ impact: faible (119.0/51.0)

Règle 2 : *MPC ≤ 21*
OMMIC ≤ 4
AMMIC > 3
→ impact: faible (32.0)

Règle 11 : *MPC ≤ 21*
OMMIC > 4
CBOU ≤ 7
→ impact: faible (68.0/12.0)

Règle 12 : *MPC ≤ 21*
OMMIC > 4
CBOU > 7
→ impact: moyen (9.0/1.0)

Règle 15 : *MPC > 36*
CBOU > *14*
AMMIC ≤ 5
→ impact: moyen (4.0/1.0)

Règle 16 : *MPC > 36*
CBOU > *14*
AMMIC > 5
→ impact: très-fort (2.0)

Figure 1. *Règles de causalité (J48)*

La première remarque à faire sur cet ensemble de règles est qu'il y a 14 règles sur 16, où les métriques de couplage d'importation sont impliquées. Cela montre bien l'influence de cette propriété particulière de couplage sur l'impact de changement. En observant bien ce sous ensemble de 14 règles, on peut encore distinguer 3 sous-ensembles particuliers. Le premier sous-ensemble est formé des 10 premières règles, où figurent dans la partie gauche les deux métriques de couplage d'importation OMMIC et AMMIC. Le deuxième est formé des règles 11 et 12, où figurent seulement la métrique OMMIC. Le troisième est formé des règles 15 et 16, où figurent seulement la métrique AMMIC. Dans le premier sous ensemble, on remarque que dans la plupart des cas, l'impact est faible ou très-faible pour les classes qui présentent un faible couplage d'importation Méthode-Méthode-Autres (OMMIC ≤ 4, la moyenne est 9.26) et un faible/moyen couplage d'importation Méthode-Méthode-Ancêtres (AMMIC entre 0 et 3, la moyenne est

2.15). Dans le second sous-ensemble, la métrique OMMIC n'est pas déterminante mais elle représente un élément important à considérer dans la prédiction de l'impact. Ce dernier est faible ou moyen selon que le nombre de classes utilisées par la classe cible est moyen ou grand (CBOU est ≤ 7 ou > 7, la moyenne est 3.57). Cela est valable pour les classes qui disposent d'un nombre d'invocations statiques de méthodes pas très grand (MPC ≤ 21, la moyenne est 11.34) et d'un couplage d'importation Méthode-Méthode-Autres pas trop petit (OMMIC > 4, la moyenne est 9.26). Enfin, dans le troisième sous-ensemble, les règles expriment que pour les classes dont le nombre d'invocations statiques de méthodes est grand (MPC > 36) et dont le nombre de classes utilisées par la classe cible est grand aussi (CBOU>14), le couplage d'importation Méthode-Méthode-Ancêtres est déterminant dans le sens où l'impact pour ces classes sera moyen ou très-fort selon la valeur de cette propriété architecturale, quelle soit moyenne (autour de 2.5 et ≤ 5) ou grande (>5). Cela représente un résultat important de cette expérimentation.

PART

PART [FRA 98] permet d'inférer des règles par la génération itérative d'arbres de décision partiels en combinant deux paradigmes majeurs : les arbres de décision et la technique d'apprentissage des règles "diviser pour régner". La combinaison de ces deux paradigmes ajoute de la flexibilité et de la vitesse. En effet, il est inutile de construire un arbre de décision plein pour obtenir une règle simple. Le processus peut être accéléré sensiblement sans sacrifier les avantages des deux approches. L'idée principale est de construire un arbre de décision *partiel* au lieu d'un arbre entièrement exploré. PART fournit des résultats aussi précis que ceux de l'algorithme J48. Il permet d'éviter l'optimisation globale de ce dernier et fournit un ensemble de règles compactes et exactes.

Suite à l'exécution de l'algorithme PART sur notre ensemble de données, le taux de succès obtenu (65.48 %) semble assez bon, tout en étant plus faible que celui obtenu par J48, et l'ensemble résultat est formé de 25 règles. La première constatation à faire sur cet ensemble est que sur les 25 règles, il y a 16 règles où les métriques de couplage d'importation sont impliquées. Cela confirme la remarque faite auparavant (l'influence de cette propriété particulière de couplage sur l'impact de changement). D'autre part, plusieurs règles sont similaires à certaines trouvées par J48. Nous citons à titre d'exemples les règles 3 et 4 au niveau de PART, et 2 et 11 au niveau de J48. Nous tenons à signaler que pour les deux algorithmes, les règles 15 sont identiques. Cela confirme partiellement le résultat important trouvé par J48 (voir la fin de la section précédente). La figure 2 montre quelques règles choisies parmi l'ensemble des règles générées par PART.

Règle 3 : MPC <= 13
AMMIC > 3
→Impact: faible (33.0/1.0)

Règle 4 : MPC <= 13
OMMIC > 4
CBOU <= 1
→Impact: faible (6.0)

Règle 15 : MPC > 36
CBOU > 14
AMMIC <= 5
→Impact: moyen (4.0/1.0)

Figure 2. *Règles de causalité (PART)*

NBTree (Naïve-Bayes decision-Trees)

Kohavi [KOH 96] propose NBTree comme une approche hybride combinant le classificateur Bayésien naïf et le classificateur à base d'arbre de décision. Ce classificateur hybride obtient fréquemment une très haute précision par rapport au classificateur Bayésien naïf ou au classificateur de type arbre de décision. Il utilise une structure arborescente pour diviser l'espace d'instances en sous-espaces et générer un classificateur Bayésien naïf pour chaque sous-espace. Dans un arbre de décision conventionnel, chaque feuille est marquée avec une seule classe et prédit cette classe pour les instances qui atteignent la feuille, alors qu'un arbre Bayésien naïf utilise un classificateur Bayésien naïf local pour prédire les classes de ces instances. Suite à l'exécution de l'algorithme NBTree sur notre ensemble de données, le taux de succès obtenu est assez intéressant (66.75%). L'arbre de décision généré est compact (nombre de nœuds = 17). Il contient 9 feuilles, contenant chacune un classificateur Bayésien permettant de déduire la classe à prédire avec plus de précision. Chaque chemin de la racine à une feuille donnée est une règle de causalité. Nous avons donc un ensemble de 9 règles. La figure 3 présente quelques règles de cet ensemble. Notons qu'au niveau de la conclusion de chacune des règles, on trouve le numéro du

classificateur Bayésien naïf (Naïve-Bayes), soit NB3 pour la règle 1, suivi de la classe à prédire, qui est en fait la classe qui a la probabilité la plus élevée. Cette probabilité est mentionnée entre parenthèses. Les résultats de cet algorithme affirment qu'en plus du couplage d'importation, l'impact est aussi influencé par le couplage mesuré par les métriques CBONA et CBOU vu que ces dernières figurent dans la plupart des règles trouvées. Les règles 1 et 9 (voir figure 3) montrent bien que l'impact est très-faible ou fort selon les valeurs de ces métriques qu'elles soient petites ou grandes. Enfin, les règles 2 et 3 expriment que l'impact devient de plus en plus faible si le couplage d'importation Méthode-Méthode-Ancêtres (AMMIC) augmente. Cela confirme bien un résultat déjà trouvé par J48.

Règle 1 : CBONA ≤ 3.5
CBOU ≤ 0.5
→ NB3 : impact: très-faible (0.46)

Règle 2 : CBONA ≤ 3.5
CBOU $\in$ $]0.5,1.5]$
$AMMIC \leq 0.5$
→ NB5 : impact: faible (0.54)

Règle 3 : CBONA ≤ 3.5
CBOU $\in$ $]0.5,1.5]$
$AMMIC > 0.5$
→ NB6 : impact: très-faible (0.76)

Règle 9 : CBONA > 3.5
CBOU > 36.5
→ NB16 : impact: fort (0.48)

Figure 3. *Règles de causalité (NBTree)*

5 Conclusion

Nous avons proposé dans cet article une approche et défini une démarche afin de répondre à la problématique d'analyse et de prédiction de l'impact des changements dans un système à objets. Une étude approfondie et une synthèse générale des différents travaux antérieurs traitant de ce sujet étaient indispensables. Afin de concrétiser notre approche, nous avons choisi un modèle d'impact existant et nous l'avons adapté au langage Java. Par la suite, nous avons proposé une technique de calcul d'expressions d'impact de changement en utilisant une approche par méta-modèle (PTIDEJ). Cette technique prend en charge tout changement dont l'expression d'impact fait appel aux liens d'association, d'agrégation et d'héritage. Pour vérifier notre approche, nous avons réalisé une étude empirique dans laquelle nous avons avancé une hypothèse de corrélation entre le couplage et l'impact de changement. L'expérimentation a été faite sur le système BOAP (système développé au CRIM). Il contient en tout 394 classes. Par la suite, un changement a été concrètement porté sur ce système, il s'agit du changement de type de variable. L'impact de ce changement est déduit par le modèle utilisé puis calculé par le biais de la technique proposée. Un ensemble de métriques reliées à la propriété de couplage a été extrait du système BOAP. Nous avons utilisé 3 algorithmes d'apprentissage de l'environnement Weka pour vérifier notre hypothèse.

Les taux de succès obtenus pour les 3 algorithmes semblent assez intéressants. Les résultats trouvés par J48 puis confirmés par PART, expriment que le couplage d'importation influence beaucoup plus l'impact de changement que les autres types de couplage vu que dans la plupart des cas (règles obtenues), l'impact est principalement lié à ce type de couplage. Aussi, comme autre résultat important de cette expérimentation, il s'avère que pour les classes dont le nombre d'invocations statiques de méthodes ainsi que le nombre de classes utilisées par la classe cible sont grands, le couplage d'importation (mesurée par la métrique AMMIC) décide de l'impact de changement. Ce résultat a été trouvé par J48 puis partiellement confirmé par PART. Enfin, les résultats de NBTree ont ajouté plus de précisions aux résultats déjà trouvés par J48 et PART, et ont montré qu'en plus du couplage d'importation, l'impact est aussi influencé par le couplage mesuré par les métriques CBONA et CBOU.

Comme travaux futurs, nous envisageons d'autres expérimentations sur d'autres systèmes afin de confirmer encore plus ces résultats. Nous nous intéressons aussi à d'autres mesures de couplage, ainsi qu'à d'autres types de propriétés architecturales expliquant mieux les mécanismes les plus communs aux propagations de modification (ripple-effect) à travers les systèmes à objets. Il serait intéressant, à notre avis, de comparer l'impact de changements pour des systèmes différents et trouver ainsi des résultats applicables à une large catégorie de systèmes.

Références

[ALB 03]: ALBIN-AMIOT H., "Idiomes et Patterns Java : Application à la Synthèse de Code et à la Détection". Thèse de doctorat, université de Nantes, février 2003.

[ANT 99]: ANTONIOL G., CANFORA G., LUCIA A. D., "Estimating the size of changes for evolving object OrientedSystems: a Case Study" in Proceedings of the 6th International Software Metrics Symposium, pages 250-258, Boca Raton, Florida, Nov 1999

[BRI 99]: BRIAND L. C., WÜST J., LOUNIS H., "Using Coupling Measurement for Impact Analysis in Object-Oriented Systems" in proceedings of the International Conference on Software Maintenance ICSM'99, Oxford, England, August 30 – September 3, 1999.

[BRI 01]: BRIAND L. C., WUST J., H. LOUNIS H., "Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs". In Empirical Software Engineering, an International Journal, 6 (1):11-58, March 2001, Kluwer Academic Publishers.

[CHA 98]: CHAUMUN M. A., "Change Impact Analysis in Object-Oriented Systems: Conceptual Model and Application to C++". Master's thesis, Université de Montréal, Canada, November 1998.

[CAN 01]: CANTAVE R., "Abstractions via un modèle générique d'application orientée objet", Master's thesis, Université Laval, Canada, Avril 2001

[CHA 99]: CHAUMUN M. A., KABAILI H., KELLER R. K., LUSTMAN F., "A Change Impact Model for Changeability Assessment in Object-Oriented Software Systems". In Proceedings of the Third Euromicro Working Conference on Software Maintenance and Reengineering CSMR'99, pages 130-138, Amsterdam, The Netherlands, March 1999.

[CHI 94]: CHIDAMBER S. R., KEMERER C. F., "A Metrics Suite for Object Oriented Design" in IEEE Transactions on Software Engineering, Vol. 20, No. 6, pages 476-493, June 1994.

[CHE 04]: CHEÏKHI L., "Estimation de l'impact du changement dans les programmes à Objets", Master's thesis, Université de Montréal, Canada, November 2004.

[ELH 02] : EL HACHEMI A., SNOUSSI H., "BOAP 1.1.0 : Manuel d'utilisation", CRIM, Janvier 2002.

[FRA 98]: FRANK E., WITTEN I.H., "*Generating Accurate Rule Sets Without Global* Optimization" in Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers, San Francisco, CA, 1998.

[GUÉ 03]: GUÉHÉNEUC Y., "Un cadre pour la traçabilité des motifs de conception", Thèse dedoctorat de l'université de Nantes, École Nationale Supérieure des Techniques Industrielles et des Mines de Nantes, juin 2003.

[KAB 02]: KABAILI H., "Changeabilité des logiciels orientés objet : propriétés architecturales et indicateurs de qualité", PhD thesis, Université de Montréal, Canada, Janvier, 2002.

[KOH 96]: KOHAVI R., "*Scaling up the accuracy of naive-Bayes classifiers: a decision tree hybrid*" in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, (1996).

[KUN 95]: KUNG D. C., GAO J., HSIA P., LIN J., TOYOSHIMA Y., "Class firewall, test order, and regression testing of object-oriented programs" in Journal of Object-Oriented Programming, Vol. 8, No. 2, pages 51-65, May 1995.

[LEE 96]: LEE M., OFFUTT A. J., "Algorithmic Analysis of the Impact of Changes to Object-Oriented Software" in ICSM96, pages 171-184, 1996.

[LEE 98]: LEE M., "Change Impact Analysis for Object-Oriented Software". PhD thesis, George Mason University, Virginia, USA, 1998

[LI 93]: LI W., HENRY S., " Object-Oriented Metrics that Predict Maintainability" in Journal of Systems and Software, Vol. 23, pages 111-122, 1993

[LOU 97]: LOUNIS H., SAHRAOUI H. A., MELO W. L., "Defining, Measuring and Using Coupling metrics in Object-Oriented Environment" in SIGPLAN OOPSLA'97 Workshop on Object-Oriented Product Metrics, 1997, Atlanta, Georgia, USA, 1997.

[OTI 01]: Object Technology International, Inc. / IBM. Eclipse platform – A universal tool platform, July 2001.

[QUI 93]: QUINLAN J.R., "C4.5: Programs for Machine Learning". Morgan Kaufmann Publishers, Sao Mateo, CA, 1993.

[SAH 00] SAHRAOUI H. A., GODIN R., MICELI T., "Can metrics help to bridge the gap between the improvement of OO design quality and its automation ?", *in Proceedings of the International Conference on Software Maintenance (ICSM'00)*, 2000

[SCH 01]: SCHAUER R, KELLER R. K., LAGUË B., KNAPEN G., ROBITAILLE S., SAINT-DENIS G., "The SPOOL Design Repository: Architecture, Schema, and Mechanisms. In Hakan Erdogmus and Oryal Tanir, ditors, Advances in Software Engineering. Topics in Evolution, Comprehension, and Evaluation. Springer-Verlag, 2001.

[WIL 99]: WILKIE F. G., KITCHENHAM B. A., "Coupling Measures and Change Ripples inC++ Application Software", published in the proceedings of EASE'99, University of Keele, UK, 1999.

[WIT 00]: WITTEN I. H., FRANK E., "Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation", © 2000 Morgan Kaufmann Publishers.

Optimisation II

$FH2(P2, P2)$ hybrid flow shop scheduling with recirculation of jobs

Nadjat Meziani[1] and Mourad Boudhar[2]

[1]University of Abderrahmane Mira Bejaia, Algeria
[2]USTHB University Algiers, Algeria
ro_nadjet07@yahoo.fr and mboudhar@yahoo.fr

Abstract. In this paper, two hybrid flow shop scheduling problems to minimize the makespan are presented. The first is the hybrid flow shop on two stages with only one machine on each stage and recirculation of jobs. This problem is polynomial. The second one is the hybrid flow shop on two stages such as each one contains two identical parallel machines and every job recirculates a finite number of times. This problem is NP-hard in general. Linear mathematical formulation and heuristics are also presented with numerical experimentations.

Key words: Hybrid flow shop, complexity, recirculation, makespan

1 Introduction

Problems of the flood production with recirculation marked the interest of the researchers these last years. In [1], Bertel and Billaut consider the hybrid flow shop scheduling problem with recirculation of jobs and suggest a genetic algorithm to minimize the weighted number of late jobs. To minimize the maximum lateness in the hybrid flow shop scheduling problem with recirculation, Choi et al. [3] propose several lists of scheduling algorithms . An exact method and heuristics were presented in [4], to solve the problem on two stages with recirculation to minimize the makespan under the maximum dues dates. Another type of the hybrid flow shop problem on two stages with recirculation of jobs, to minimize the maximum completion processing times of jobs, was studied in [2] where the authors presented two problems. The first problem is a flow shop with two machines with recirculation of jobs which is polynomial. The second is of the same type, except that the first stage consists of only one machine and the second of two identical parallel machines. This problem is proved NP-hard. In our work, we interest to the hybrid flow shop scheduling problem on two stages with recirculation of jobs with the objective to minimize the makespan.

The problem of jobs recirculation is drawn from a practical application and appears in the workshops painting of the metallic doors, bicycles, cars or other finite or semi-finite processes where each product has to pass through several operations to complete the painting process. Thus any product passes by two stages. On the first stage, each product undergoes a test or a control of quality

and compliance on machines for several times and noncomplying products are rejected. The product retaines, will pass on the second stage whose the first operation is the anti-rust treatment, then the product must return once more on the machine to carry out the following operation, which is the actual painting. By recirculating one final time on the same machine, the operation of brightness is applied to each product. The time of drying that separates two operations can be included in the processing time.

In this paper, two scheduling problems to minimize the makespan were studied. The first is the hybrid flow shop on two stages with only one machine on each stage and recirculation of jobs. This problem is polynomial. The second one is the hybrid flow shop on two stages such as each one contains two identical parallel machines and every job recirculates a finite number of times. This problem is NP-hard in general. Linear mathematical formulation and heuristics are also presented with numerical experimentations.

2 Complexity

The two stage hybrid flow shop scheduling problem with m_1 identical parallel machines on the first stage and m_2 identical parallel machines at the second one in order to minimize the makespan (C_{max}), noted by $FH2(Pm_1, Pm_2)//C_{max}$, was studied by Gupta [5]. He has shown that the problem is NP-hard in the strong sens as soon as a stage contains more than one machine whereas its opposite problem was tackled in [6] by Gupta and Tunc. Hoogeveen et al. [8] proved that the same problem with two machines at the first stage and only one machine at the second stage, $FH2(P2, 1)//C_{max}$, is NP-hard in the ordinary sens. They have also proved that this latter problem with preemtion of jobs noted by $FH2(P2, 1)/pmtn/\ C_{max}$ is NP-hard in the strong sens.

The following theorem is a generalization of the theorem of equivalence of the hybrid flow shop scheduling problem with two stages and its reverse stated in [7].

Theorem 1. *The hybrid flow shop problem on two stages with recirculation of jobs on the second stage* $FH2(Pm_1, Pm_2)/recr(2)/C_{max}$ *and its reverse* $FH2$ $(Pm_2, Pm_1)/recr(1)/C_{max}$ *with recirculation of jobs on the first stage are equivalent.*

3 Polynomial subproblem: one machine at each stage

We consider the flow shop scheduling problem on two machines with recirculation of jobs on the two machines, $F2/recr(1, 2)/C_{max}$, which the objective is to minimize the makespan C_{max} of the jobs. Let n independent jobs to schedule on these machines. Each job J_i must be processed a finite number of times n_{i1} on the first machine with the processing time p_{1ij} and a finite number n_{i2} on the second machine with the processing time p_{2il}. The objective is to minimize the makespan. For the resolution of the problem, we propose the following algorithm

which is based on the Johnson's rule [9] and provided an optimal solution.

Algorithm $F2RC(1,2)$

1. Transform the considered problem to $F2//C_{max}$ as follows:
   the processing time on the first and the second machine are given respectively by: $p'_{1i} = \sum_{j=1}^{n_{i1}} p_{1ij}$, $p'_{2i} = \sum_{l=1}^{n_{i2}} p_{2il}$
2. Solve the problem $F2//C_{max}$ with Johnson's algorithm.
3. Build the solution of the initial problem $F2/recr(1,2)/C_{max}$ by subdividing the processing time of each job on the two machines of each stage in processing time of its initial elementary operations.

Theorem 2. *The algorithm F2RC(1,2) provides an optimal solution to the problem* $F2/recr(1,2)/C_{max}$ *in* $O(max\{nlogn, N\})$ *where* $N = \sum_{i=1}^{n}(n_{i1} + n_{i2})$.

Proof. Let us suppose that in an optimal solution of the problem $F2/recr(1,2)/C_{max}$, we have at least two unspecified operations of the same job which are not processed successively on one or two machines (see Fig. 1).

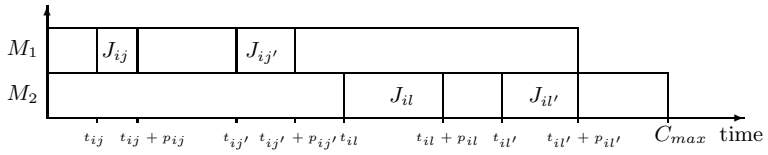


Fig. 1. An example of jobs scheduling

By scheduling on the first machine M_1 the operation J_{ij} at the date $t_{ij'} - p_{ij}$, the completion time of jobs doesn't changed because the operations processed between jobs J_{ij} and $J_{ij'}$ will be moved in the worst of the cases with p_{ij} units of time towards the left, therefore processed before $t_{ij'} + p_{ij'}$. In this manner, the dates of the beginning operations process on the second machine are respected.

By scheduling on the second machine the operation $J_{il'}$ at the date $t_{il} + p_{il}$ the solution remains feasible and C_{max} does not change a value because the treated operations between J_{il} and $J_{il'}$ will be moved in the worst of the cases with $p_{il'}$ units of time towards the right, therefore treated before $t_{il'} + p_{il'}$. By repeating these two operations a finite number of times, we obtain a solution where all the operations of the same job are successively scheduled on the two machines (one after the other without idle time). The first step of the algorithm requires $O(N)$ operations and the algorithm of Johnson turns in $O(nlogn)$. Therefore the problem $F2/recr(1,2)/C_{max}$ is polynomial.

4 Problem $FH2(P2,P2)/recr(1,2)/C_{max}$

Let n independent jobs to schedule on two stages. The first stage consists of two identical parallel machines M_{11}, M_{12} and the second stage is composed of two others identical parallel machines M_{21}, M_{22}. The workshop is of hybrid flow shop type on two stages. Each job J_i must be processed a finite number of times n_{i1} and n_{i2} with the processing times p_{1ij} and p_{2il} on the first and the second stages respectively. The objective is the minimization of the makespan (C_{max}). The problem is denoted by $FH2(P2,P2)/recr(1,2)/C_{max}$.

The summing up of the processing times operations of the same job and their successive processing on the same machine of the second stage does not always give the best solution. We give here the following counter example:

J_i	J_1	J_2	J_3
nb_{i1}	1	2	1
p_{1i1}	2	1	2
p_{1i2}	/	1	/
nb_{i2}	2	1	2
p_{2i1}	1	3	2
p_{2i2}	2	/	1

Table 1. Processing time of jobs

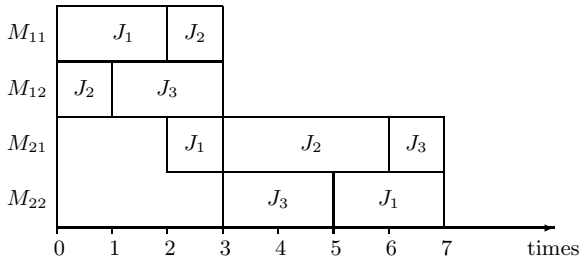


Fig. 2. Scheduling of jobs

4.1 Mathematical modeling

The problem $FH2(P2,P2)/recr(1,2)/C_{max}$ is formulated in a linear mathematical program with real and binary variables.

Variables
Let's consider the following variables s_{ijk}, r_{ilh}, X_{ijk} , $\alpha_{ij}^{i'j'}$, Y_{ilh} and $\beta_{il}^{i'l'}$:

- s_{ijk} : the starting time of the operation j of the job J_i on the machine M_k at the first stage, for $i=\overline{1,n};\ j=\overline{1,n_{i1}};\ k=1,2$;
- r_{ilh} : the starting time of the operation l of the job J_i on the machine M_h at the second stage, for $i=\overline{1,n};\ l=\overline{1,n_{i2}};\ h=3,4$;
- $X_{ijk}=\begin{cases}1, \text{if the operation } j \text{ of the job } J_i \text{ processed on the machine } k\\ \quad \text{at the first stage;}\\ 0, \text{else.}\end{cases}$
  for $i=\overline{1,n};\quad j=\overline{1,n_{i1}};\quad k=1,2$;
- $\alpha_{ij}^{i'j'}=\begin{cases}1, \text{if the starting time of the operation } j \text{ of the job } J_i\\ \quad \leq \text{at starting time of the operation } j' \text{ of the job } J_{i'};\\ 0, \text{else.}\end{cases}$
- $Y_{ilh}=\begin{cases}1, \text{if the operation } l \text{ of the job } J_i \text{ processed on the machine } h\\ \quad \text{at the second stage ;}\\ 0, \text{else.}\end{cases}$
  for $i=\overline{1,n};\quad l=\overline{1,n_{i2}};\quad h=3,4$;
- $\beta_{il}^{i'l'}=\begin{cases}1, \text{if the starting time of the operation } l \text{ of the job } J_i\\ \quad \leq \text{at starting time of the operation } l' \text{ of the job } J_{i'};\\ 0, \text{else.}\end{cases}$
  for $i,i'=\overline{1,n};\quad l,l'=\overline{1,n_{i2}};$

Z : the completion time of all the jobs.

Constraints related to the first stage

– An operation of a job is assigned only to one machine:
$\sum_{k=1}^{2} X_{ijk}=1\ ; \qquad i=\overline{1,n}; \qquad j=\overline{1,n_{i1}}\ ;$

– For any pair of operations we have:
$\alpha_{ij}^{i'j'}+\alpha_{i'j'}^{ij}=1\ ; \quad i,i'=\overline{1,n}; \quad j=\overline{1,n_{i1}}; \quad j'=\overline{1,n_{i'1}}; \quad j\neq j' \ if \ i=i';$

– On the same machine, the processing of an operation of a job starts only if the processing of the operation which precedes it is over:

$s_{ijk}+p_{1ij}-s_{i'j'k}\leq M_1\cdot(1-\alpha_{ij}^{i'j'}+2-X_{ijk}-X_{i'j'k})\ ; \quad i,i'=\overline{1,n};$
$j=\overline{1,n_{i1}}\ ; j'=\overline{1,n_{i'1}}\ ; j\neq j' \quad \text{if} \quad i=i'\ ; \quad k=1,2;$
$s_{i'j'k}+p_{1i'j'}-s_{ijk}\leq M_1\cdot(\alpha_{ij}^{i'j'}+2-X_{ijk}-X_{i'j'k})\ ; \quad i,i'=\overline{1,n}; \quad j=$
$\overline{1,n_{i1}}\ ; \ j'=\overline{1,n_{i'1}}\ \ ; \ j\neq j' \quad \text{if} \quad i=i'\ ; \quad k=1,2;$

where M_1 is a very large value which may be equal to $\sum_{i=1}^{n}\sum_{j=1}^{n_{i1}} p_{1ij}$.

– Two operations of the same job cannot be processed at the same time on two different machines:

$s_{ijk} + p_{1ij} - s_{ij'k'} \leq M_1 \cdot (1 - \alpha_{ij}^{ij'} + 2 - X_{ijk} - X_{ij'k'})$; $i = \overline{1,n}$; $j, j' = \overline{1,n_{i1}}$; $k, k' = 1,2$; $j \neq j'$; $k \neq k'$;

$s_{ij'k} + p_{1ij'} - s_{ijk'} \leq M_1 \cdot (\alpha_{ij}^{ij'} + 2 - X_{ij'k} - X_{ijk'})$; $i = \overline{1,n}$; $j, j' = \overline{1,n_{i1}}$;$k, k' = 1,2$; $j \neq j'$; $k \neq k'$;

where M_1 is a very large value which may be equal to $\sum_{i=1}^{n}\sum_{j=1}^{n_{i1}} p_{1ij}$.

Constraints related to the second stage

– An operation of a job is assigned only to one machine:
$\sum_{h=3}^{4} Y_{ilh} = 1$; $i = \overline{1,n}$; $l = \overline{1,n_{i2}}$;

– For any pair of operations we have:
$\beta_{il}^{i'l'} + \beta_{i'l'}^{il} = 1$; $i, i' = \overline{1,n}$; $l = \overline{1,n_{i2}}$; $l' = \overline{1,n_{i'2}}$; $l \neq l'$ if $i = i'$;

– On the same machine, the processing of an operation of a job starts only if the processing of the operation which precedes it is over:

$r_{ilh} + p_{2il} - r_{i'l'h} \leq M_2 \cdot (1 - \beta_{il}^{i'l'} + 2 - Y_{ilh} - Y_{i'l'h})$; $i, i' = \overline{1,n}$; $l = \overline{1,n_{i2}}$; $l' = \overline{1,n_{i'2}}$; $l \neq l'$ if $i = i'$; $h = 3,4$;
$r_{i'l'h} + p_{2i'l'} - r_{ilh} \leq M_2 \cdot (\beta_{il}^{i'l'} + 2 - Y_{ilh} - Y_{i'l'h})$; $i, i' = \overline{1,n}$; $l = \overline{1,n_{i2}}$; $l' = \overline{1,n_{i'2}}$; $l \neq l'$ if $i = i'$; $h = 3,4$;
where M_2 is a very large value which may be equal to $\sum_{i=1}^{n}\sum_{l=1}^{n_{i2}} p_{2il}$.

– Two operations of the same job cannot be processed at the same time on two different machines:

$r_{ilh} + p_{2il} - r_{il'h'} \leq M_2 \cdot (1 - \beta_{il}^{il'} + 2 - Y_{ilh} - Y_{il'h'})$; $i = \overline{1,n}$; $l, l' = \overline{1,n_{i2}}$; $h, h' = 3,4$; $l \neq l'$; $h \neq h'$;
$r_{ij'k} + p_{2ij'} - r_{ijk'} \leq M_2 \cdot (\beta_{il}^{il'} + 2 - Y_{il'h} - Y_{ilh'})$; $i = \overline{1,n}$; $l, l' = \overline{1,n_{i2}}$; $h, h' = 3,4$; $l \neq l'$; $h \neq h'$;
where M_2 is a very large value which may be equal to $\sum_{i=1}^{n}\sum_{l=1}^{n_{i2}} p_{2il}$.

Constraints binding the starting times of the jobs of the first stage with the second stage

– If no sequence is imposed for the operations:

- The processing of a job starts on the second stage only if the processing of its last operation on the first stage is over:
  $s_{ijk}+p_{1ij} \leq r_{ilh}$; $\quad i=\overline{1,n}$; $\quad j=\overline{1,n_{i1}}$; $\quad k=1,2$; $l=\overline{1,n_{i2}}$; $\quad h=3,4$;

- The completion time of a job on the second stage is lower than, or equal to, Z :
  $r_{ilh}+p_{2il} \leq Z$; $\quad i=\overline{1,n}$; $\quad l=\overline{1,n_{i2}}$; $\quad h=3,4$;

– If an operation sequence is imposed:

- The execution of an operation of a job J_i can start only if the processing of the preceding operation is over:
  $s_{i1k}+p_{1i1} \leq s_{i2k}$; $\quad i=\overline{1,n}$; $\quad k=1,2$;

  $\vdots$

  $s_{i(n_{i1}-1)k}+p_{1i(n_{i1}-1)} \leq s_{in_{i1}k}$; $\quad i=\overline{1,n}$; $\quad k=1,2$;

  $s_{in_{i1}k}+p_{1in_{i1}} \leq r_{i1h}$; $\quad i=\overline{1,n}$; $\quad k=1,2 \quad h=3,4$;

  $r_{i1h}+p_{2i1} \leq r_{i2h}$; $\quad i=\overline{1,n}$; $\quad k=1,2$; $\quad h=3,4$;

  $\vdots$

  $r_{i(n_{i2}-1)h}+p_{2i(n_{i2}-1)} \leq r_{in_{i2}h}$; $\quad i=\overline{1,n}$; $\quad k=1,2$; $\quad h=3,4$;

- The completion time of a job on the second stage is lower than, or equal to, Z :
  $r_{in_{i2}k}+p_{2in_{i2}} \leq Z$; $\quad i=\overline{1,n}$; $\quad k=1,2$; $\quad h=3,4$;

The objective function is: $min(Z)$.

4.2 Lower bounds

In what follows, two lower bounds on the makespan C_{max} are proposed: LB_1 and LB_2.

Proposition 1. $LB_1 = \min\limits_{1\leq i\leq n}\left\{\sum\limits_{j=1}^{n_{i1}} p_{1ij}\right\} + \left\lceil \frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}\right\rceil$ *is a lower bound on the makespan.*

Proof. $\left\lceil \frac{1}{2}\sum\limits_{i=1}^{n}\sum\limits_{l=1}^{n_{i2}} p_{2il}\right\rceil$ is a lower bound for the total completion time of the operations in the second stage if their processing begins at $t=0$. It is deduced from the problem $P2//C_{max}$, and the processing of the operations in the second stage can begin only if at least one operation of the same job is completed in the first stage.

Proposition 2. $LB_2 = \left\lceil \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n_{i1}} p_{1ij} \right\rceil + \min_{1 \leq i \leq n} \left\{ \sum_{l=1}^{n_{i2}} p_{2il} \right\}$ *is a lower bound on the makespan.*

Proof. $\left\lceil \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n_{i1}} p_{1ij} \right\rceil$ a lower bound for the total completion time of the operations in the first stage. It is deduced from the problem $P2//C_{max}$. Also, the processing of the operations in the second stage can begin only if at least one operation of the same job is completed in the first stage.

Consequently $LB = max\{LB_1, LB_2\}$ is also a lower bound.

4.3 Heuristics

For the resolution of the $FH2(P2, P2)/recr(1, 2)/C_{max}$ problem, we present three heuristics.

Heuristic1 The heuristic $H1\_J$ is based on the algorithm $F2RC(1, 2)$ which determines the sequence of jobs to schedule on the first stage according to the Johnson rule[9], then jobs are assigned on the second stage according to the First Available Machine (FAM) rule.

H1_J heuristic

1. Transform the problem $FH2(P2, P2)/recr(1, 2)/C_{max}$ into the problem $F2//C_{max}$, by considering only one machine on each stage, and by summing up the processing times of the operations of each job on the two stages.
2. Solve the obtained problem, $F2//C_{max}$, by Johnson algorithm's.
3. Process the jobs on the two stages according to their order obtained at the preceding stage.
4. Return to the initial problem.

Heuristic2 In the second heuristic $H2\_SPT$, we add initially, the processing time of the operations of each job on the two stages. Then, to obtain the sequence of the jobs to be followed, we applie the Shortest Processing Time (SPT) rule according the processing time of the jobs of the first stage for then affecting them on the two stages according to the third phase of this heuristic.

H2_SPT heuristic

Step 1: Add the processing time of the operations of each job on the two stages.
Step 2: Apply the SPT (Shortest Processing Time) rule over processing times of the new jobs obtained on the first stage.
Step 3: Assignment of jobs:

- For i:=1 to n do
  Let r^1 and r^2 be the smallest times of availabilities of the machines of the first and second stages

  - If $r^1 + p_{1i} \geq r^2$ then
  - The job J_i is assigned on the machine realizing r^1.
  - On the second stage, it's assigned on the machine which minimizes its completion time. In the event of the multiple choice, take that which is free latest.
  - Else the job J_i is assigned on the machine of the second stage realizing r^2. On the first stage, it's assigned on the machine which minimize its waiting.
  - Endif
- Update r^1 et r^2.
- Enddo.
- Return to the initial problem.

Heuristic3 The only difference from the $H2\_SPT$ heuristic, is that on stage two the Longest Processing Time (LPT) rule is applied unstead of Shortest Processing Time (SPT) rule.

H3_LPT heuristic
Step 1: Add the processing time of the operations of each job on the two stages.
Step 2: Apply the LPT (Longest Processing Time) rule over processing times of the new jobs obtained on the first stage.
Step 3: Assignment of jobs:

- For i:=1 to n do
  Let r^1 and r^2 be the smallest times of availabilities of the machines of the first and second stages
  - If $r^1 + p_{1i} \geq r^2$ then
  - The job J_i is assigned on the machine realizing r^1.
  - On the second stage, it's assigned on the machine which minimizes its completion time. In the event of the multiple choice, take that which is free latest.
  - Else the job J_i is assigned on the machine of the second stage realizing r^2. On the first stage, it's assigned on the machine which minimizes its waiting.
  - Endif
- Update r^1 et r^2.
- Enddo.
- Return over to the initial problem.

5 Experimentations

Several instances are randomly generated according to the uniform law on which we applied the three heuristics cited above. For each instance, the number n of jobs take its values on the set $[10, 20, 50, 100, 250, 500, 1000]$. Processing times of the jobs will be taken in the intervals $[1, 50]$. Operation numbers of the jobs

will be taken in the intervals $[1, 5]$ and $[1, 10]$. We have coded our algorithms in Delphi (Version7.0) and have run them on a Pentium III 1.0GHz Personal Computer with 256MB RAM.

The results obtained are given in a table below. For each value of n, 100 instances are generated. The first line indicated the percentage where the solution found by the heuristic is better compared with other solutions, the second line indicated the average execution time of each heuristic (in milliseconds).

		$p_{1ij}, p_{2il} \in [1,50]$ $n_{i1}, n_{i2} \in [1,5]$			$p_{1ij}, p_{2il} \in [1,50]$ $n_{i1}, n_{i2} \in [1,10]$		
		$H1\_J$	$H2\_SPT$	$H3\_LPT$	$H1\_J$	$H2\_SPT$	$H3\_LPT$
$n = 10$	C_{max}	61	11	28	66	16	18
	Avr-time	2.7	1.52	3	2.1	1.9	3.6
$n = 20$	C_{max}	68	11	21	68	6	26
	Avr-time	2.8	2.2	6.32	3.01	2.6	6.5
$n = 50$	C_{max}	70	7	23	68	4	28
	Avr-time	5.5	4.4	14.83	5.2	5.5	17.12
$n = 100$	C_{max}	73	3	24	74	3	23
	Avr-time	8.51	12.23	26.23	7.91	10.51	33.64
$n = 250$	C_{max}	76	2	22	76	2	22
	Avr-time	19.52	28.11	69.84	18.61	27.72	84.91
$n = 500$	C_{max}	80	5	15	80	1	19
	Avr-time	41.76	62.91	145.98	43.27	64.9	185.28
$n = 1000$	C_{max}	84	4	12	84	1	15
	Avr-time	107.7	150.62	336.37	110.88	152.81	414.58

According to the results obtained, we note that the heuristics $H1\_J$ works better for any number of jobs and a number of operations with the average execution time lower than the other two heuristics if the number of jobs is large.

6 Conclusion

We have presented two recirculate hybrid flow shop scheduling problems with two stages to minimize the maximum completion time. The first problem is the scheduling on two machines with recirculation of jobs on these latter. This problem is polynomial and an algorithm for its resolution is proposed. The second problem consists of scheduling on two stages, with two identical parallel machines on every one. Jobs can be treated a finite number of times on the two stages. This problem is NP-hard and it is formulated as a linear mathematical program in real and binary variables. We also proposed lower bounds and heuristics, that we have tested, for the second problem. Like prospect for our work, we plan to use other methods of resolution such as the exact methods and metaheuristics.

References

1. Billaut, J.C., Bertel, S.: A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation. European J. Oper. Res. 159, 651–662 (2004)
2. Boudhar, M., Meziani, N.: Two-stage hybrid flow shop with recirculation. Int. Trans. Oper. Res. 17(2), 239–255, (2010)
3. Choi S.W., Kim Y.D., Lee G.C.: Minimizing total tardiness of orders with reentrant lots in a hybrid flow shop. Int. J. Prod. Res. 43, 2149–2167 (2005)
4. Choi H.S., Kim H.W., Lee D.H., Yoon J., Yun Y.C. and Chae K.B.: Scheduling algorithms for two-stage reentrant hybrid flow shops: minimizing makespan under the maximum allowable due dates. The Int. J. Advanced Man. Tech. 42 (2009)
5. Gupta J.N.D.: Two-stage, hybrid flow shop Scheduling problem. J. Oper. Res. Soc. 39, 359–364 (1988)
6. Gupta J.N.D., Tunc E.A.: Schedules for a two-stage hybrid flowshop with parallel machines at the second stage. Int. J. Prod. Res. 29(7), 1489–1502 (1991)
7. Gupta J.N.D., Hariri A.M.A., Potts C.N.: Scheduling a two-stage hybrid flow shop with parallel machines at the first stage, Annals Oper. Res. 69, 171–191(1997)
8. Hoogeveen J.A., Lenstra J.K., Veltman B.: Preemptive scheduling in a two stage multiprocessor flow shop is NP-hard. Euro. J. Oper. Res. 89, 172–175 (1996)
9. Johnson S.M.: Optimal two and three stage production schedules with setup time included. Nav. Res. log. Quar. 1, 61–67 (1954)

Ordonnancement sur machines identiques en présence d'ouvriers spécialisés

Mourad Boudhar et Wafaa Labbi

Faculté de Mathématiques, Université USTHB,
BP 32 Bab-Ezzouar, El-Alia 16111, Alger, Algérie
mboudhar@usthb.dz et fawalab@yahoo.fr

Résumé On s'intéresse au problème d'ordonnancement de tâches non préemptibles et indépendantes sur machines parallèles identiques en présence d'ouvriers spécialisés. Chaque tâche doit subir, avant d'être exécuter sur une machine, un traitement particulier par un ouvrier spécialisé. Les machines identiques ainsi que les ouvriers spécialisés ne peuvent traiter qu'une seule tâche à la fois. Nous montrons que le problème général est NP-difficile et nous donnons quelques sous problèmes polynomiaux. Une méthode exacte basée sur la modélisation mathématique du problème et des heuristiques sont aussi présentées avec des résultats expérimentaux. Basées sur des instances générées aléatoirement, ces expérimentations nous permettent d'apprécier l'efficacité des méthodes proposées.

Mots clés Ordonnancement, machines identiques, serveurs, makespan.

1 Introduction

Dans la littérature, de nombreuses recherches traitent des problèmes d'ordonnancement à machines parallèles identiques. Fruits de ce vaste travail, plusieurs contraintes ont été considérées. Sans contraintes de préparation, le problème de minimisation de la date de fin de traitement C_{max} a été largement étudié, ce dernier est NP-difficile [7]. Plusieurs cas de ce problème ont été également étudiés avec différents types de contraintes, pour plus de détails le lecteur peut se référer à [2,3,4,5,6,10].

Nous traitons dans ce papier un problème d'ordonnancement à machines parallèles identiques. Il s'agit d'ordonnancer un ensemble de n tâches non préemptibles et indépendantes $T_1, T_2, ..., T_n$ sur un ensemble de m machines identiques $M_1, M_2, ..., M_m$ pour optimiser le makespan qui correspond à la date de fin de traitement de l'ensemble des tâches. Chaque tâche T_i (i=$\overline{1,n}$) nécessite un temps de traitement p_i et un temps de préparation S_i. Pour cette préparation, nous disposons de k ouvriers spécialisés, il est donc impossible de faire plus de k préparations en même temps. Ce problème sera noté : $Pm/S_i, k/C_{max}$.

Pour les problèmes à machines parallèles identiques en présence d'un seul serveur pour la préparation des tâches, Abdelkhodaee et Wirth [1] ont étudiés le problème à deux machines parallèles identiques dont l'objectif est de minimiser la date de fin de traitement des tâches. Ils ont montrés que le problème général est NP-difficile au sens fort et ils ont proposés une formulation mathématique en

nombres entiers, deux cas particuliers sont étudiés ainsi que deux heuristiques en $O(nlogn)$ sont présentés avec des expérimentations numériques. Koulamas [8] a traité le problème d'ordonnancement de deux machines parallèles Semi-automatiques pour minimiser le temps mort résultant de l'indisponibilité du robot avec la condition que ces deux machines partagent le même serveur (robot) pour la préparation des tâches, il a démontré que ce dernier est NP-difficile au sens fort, aussi il a développé une procédure de réduction pour le transformer en un problème plus petit. Kravchenko et Werner [9] ont traités le problème d'ordonnancement de m machines parallèles en présence d'un seul serveur dont l'objectif est de minimiser le makespan, ils ont présentés un algorithme pseudo-polynomial pour le cas de deux machines où les temps de préparation sont égaux à 1, ils ont aussi montrés que le problème général avec un nombre arbitraire de machines est NP-difficile. Zouba et al. [11] ont étudiés le problème d'ordonnancement non préemptif sur deux machines parallèles identiques en présence d'un seule opérateur pour minimiser le makespan. Le problème consiste à déterminer des intervalles de temps d'utilisation des différentes affectations de l'opérateur pour ordonnancer les tâches sur les deux machines.

$Pm/S_i, k/C_{max}$ peut trouver des applications dans les opérations d'assemblage de petites usines où un robot est monté entre deux lignes d'assemblage de sorte qu'il peut servir les deux lignes. Ce dernier peut trouver aussi des applications dans les systèmes de fabrication flexibles (FMSs) tel qu'un atelier flexible est un environnement de fabrication souple et automatisé, il est composé généralement de trois éléments principaux : un système de fabrication, un système de manutention (ou de transport) et un système d'information. Il peut être rencontré aussi dans le cas d'un atelier de fabrication cellulaire où on a une ou plusieurs cellules constituées de machines, et entre les cellules, un ou plusieurs robots sont chargés de transporter les tâches.

Cet article est organisé comme suit. Dans la section 2, nous proposons une modélisation mathématique et dans la section 3, nous donnons une analyse de la complexité du problème. Une borne inférieure ainsi qu'un sous problème polynomial sont identifiés dans la section 4. La section 5 est consacrée aux heuristiques de résolution. Des tests numériques sont réalisés à la section 6. Une conclusion prendra forme dans la dernière section.

2 Modélisation mathématique

Dans cette section, une modélisation mathématique est proposée. Pour ce faire, considérons les variables bivalentes X_{ijt} telles que :

$X_{ijt}=\begin{cases} 1 \text{ si la tâche } T_i \text{ débute sa préparation sur la machin } M_j \text{ à l'instant t} \\ 0 \text{ sinon.} \end{cases}$

pour $i = \overline{1,n}$; $j = \overline{1,m}$ et $t = \overline{0,H-1}$ avec H un temps limite qu'on peut estimer à la borne supérieure.

On veut minimiser la date de fin de traitement de l'ensemble des tâches notée μ. Le modèle mathématique ainsi développé s'écrit :

Min μ

$$\begin{cases} \sum_{j=1}^{m}\sum_{t=0}^{H-1} X_{ijt} = 1 & i = \overline{1,n} \quad (1) \\ \sum_{y=t}^{t+S_i+p_i-1} X_{ijy} \leq 1 & j = \overline{1,m};\ i = \overline{1,n};\ t = \overline{0,H-1} \quad (2) \\ \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{y=max\{0,t-S_i+1\}}^{t} X_{ijy} \leq k & t = \overline{0,H-1} \quad (3) \\ (\sum_{j=1}^{m}\sum_{t=0}^{H-1} tX_{ijt}) + S_i + p_i \leq \mu & i = \overline{1,n} \quad (4) \\ X_{ijt} \in \{0,1\} & j = \overline{1,m};\ i = \overline{1,n};\ t = \overline{0,H-1} \quad (5) \\ \mu \in IR & \quad (6) \end{cases}$$

La première contrainte assure que l'on assigne une tâche à une machine et une seule. La deuxième contraintes indique que dans l'intervalle du temps [t,t+S_i+p_i-1] on ne peut traiter qu'au plus une seule tâche. La troisième contraintes signifie qu'on ne peut pas faire plus de k préparation à un instant t donné. Et la dernière contrainte indique que la date de fin de traitement de chaque tâche doit être inférieure ou égale à μ. Le modèle mathématique associé requiert $nmH + 1$ variables et $2n + (mnH) + H$ contraintes. Par exemple : pour $n = 3$, $m = 2$ et $H = 5$, on a 31 variables et 41 contraintes.

Les tests réalisés en utilisant un solveur de programmation linéaire mixte sur des instances de petites tailles, confirme l'efficacité de notre modélisation.

3 Analyse de la complexité

Le problème à deux machines $P2/S_i, k/C_{max}$ est NP-difficile, car le problème $P2//C_{max}$ en est un cas particulier en prenant $S_i = 0$.

Théorème 1. *Le problème $P2/p_i = p, S_i, k = 2/C_{max}$ est NP-difficile.*

Démonstration. Considérons le problème de décision NP-complet suivant connu sous le nom de 2-partition : étant donnés n entiers positifs $a_1, a_2, ..., a_n$. Existe-t-il un sous-ensemble $A_1 \subseteq A$ ($A = \{a_1, a_2, ..., a_n\}$) tel que : $\sum_{a_i \in A_1} a_i = \sum_{a_i \in A \setminus A_1} a_i$?

Montrons que ce problème se réduit polynomialement au problème de décision suivant : Etant données n tâches T_1, T_2, ..., T_n indépendantes et non morcelables avec des temps de traitement $p_i = p = 0$, le temps nécessaire pour la préparation d'une tâche est $S_i = a_i$, $k = 2$ et $Y = \frac{1}{2}\sum_{i=1}^{n} a_i$. Existe-t-il un ordonnancement de ces n tâches sur deux machines de durée inférieure ou égale à Y ?

Ce problème appartient à la classe NP, car on peut vérifier en un temps polynomial qu'une affectation des tâches aux machines vérifie toutes les contraintes. Il est clair que la réduction précédente est polynomiale ($O(n)$). Nous prouvons que le problème 2-partition a une solution si et seulement si le problème d'ordonnancement a une solution.

Si le problème 2-partition a une solution, alors il existe un sous-ensemble $A_1 \subseteq A$ avec la propriété désirée : $\sum_{a_i \in A_1} a_i = \sum_{a_i \in A \setminus A_1} a_i$. Nous construisons une solution pour le problème d'ordonnancement comme suit : on prépare les tâches qui appartiennent à l'ensemble A_1 sur la machine M_1 et les tâches restantes, on les prépare sur la machine M_2. Donc, la durée de l'ordonnancement est égale à $Y = \frac{1}{2}\sum_{i=1}^{n} a_i$.

Si le problème d'ordonnancement a une solution de durée inférieure ou égale à Y, alors, chaque tâche n'est préparée que sur l'une des deux machines. Donc le problème 2-partition a une solution, en posant A_1 comme étant l'ensemble de tâches préparées sur la machine M_1 et $A \setminus A_1$ l'ensemble de tâches préparées sur la machine M_2. □

Théorème 2. *Le problème $P2/S_i = s, k = 2/C_{max}$ est NP-difficile.*

Démonstration. Soit le problème de décision 2-partition : étant donné un ensemble $A = \{a_1, a_2, ..., a_n\}$ d'entiers positifs. Existe-t-il un sous-ensemble $A_1 \subseteq A$ tel que : $\sum_{a_i \in A_1} a_i = \sum_{a_i \in A \setminus A_1} a_i$?

Montrons que ce problème se réduit polynomialement au problème à deux machines parallèles identiques, étant données n tâches $T_1, T_2, ..., T_n$ indépendantes et non morcelables avec des temps de préparation $S_i = s = 0$ et $k = 2$, chaque tâche T_i a un temps de traitement $p_i = a_i$, et un nombre $Y = \frac{1}{2}\sum_{i=1}^{n} a_i$. Existe-t-il un ordonnancement de ces n tâches sur deux machines de durée inférieure ou égale à Y ?

Ce problème appartient à la classe NP, car on peut vérifier en un temps polynomial qu'une affectation des tâches aux machines vérifie toutes les contraintes. Il est clair que la réduction précédente est polynomiale ($O(n)$). Nous prouvons que le problème 2-partition a une solution si et seulement si le problème d'ordonnancement a une solution.

Supposons que le problème 2-partition a une solution, donc il existe un sous-ensemble $A_1 \subseteq A$ tel que $\sum_{a_i \in A_1} a_i = \sum_{a_i \in A \setminus A_1} a_i$. Ainsi, on peut construire un ordonnancement pour le problème P2/S_i = s, k=2 /C_{max} en traitant les $|A_1|$ tâches sur la machine M_1 et les $|A \setminus A_1|$ tâches restantes sur la machine M_2. Donc, la durée de l'ordonnancement est égale à Y.

Supposons maintenant que le problème d'ordonnancement a une solution de durée inférieure ou égale à Y, alors chaque tâche n'est affectée qu'à l'une des deux machines de telle sorte que A_1 soit l'ensemble des tâches affectées à la machine M_1 et $A \setminus A_1$ l'ensemble des tâches affectées à la machine M_2. Donc, le problème 2-partition a une solution. □

Théorème 3. *Le problème $P2/S_i, k = 1/C_{max}$ est NP-difficile au sens fort.*

Démonstration. La preuve de ce théorème a été donnée dans [1]. Nous proposons une preuve plus facile : il suffit de constater que ce problème est équivalent au problème défini dans [8] par C. P. Koulamas où il cherche à minimiser le temps mort résultant de l'indisponibilité du robot. Or, minimiser le temps mort revient à minimiser le makespan. □

4 Borne inférieure et étude d'un sous problème polynomial

4.1 Bornes inférieure et supérieure

Nous proposons une borne inférieure pour le C_{max}.

Théorème 4. $\overline{M} = \max\{\lceil \frac{1}{m} \sum_{i=1}^{n}(p_i + S_i)\rceil, \max_{i=1}^{n}\{p_i + S_i\}\}$ *est une borne inférieure.*

Démonstration. Par l'absurde : Si $\overline{M} < \max_{i=1}^{n}\{p_i + S_i\}$, la tâche ayant le plus grand temps nécessaire pour son traitement $(p_i + S_i)$ sera traitée sur au moins deux machines simultanément, contradiction avec les hypothèses qu' à chaque instant, une tâche ne peut être exécutée que par une seule machine au plus, et à chaque instant, une machine ne peut exécuter qu'une seule tâche à la fois. D'où $\overline{M}$ est une borne inférieure pour le C_{max}.

Si $\overline{M} < \max\{\lceil \frac{1}{m} \sum_{i=1}^{n}(p_i + S_i)\rceil$, contradiction du fait que le C_{max} ne peut pas prendre une valeur inférieure à la somme des temps de traitement et les temps de préparation des tâches divisée par le nombre de machines. □

Si toutes les tâches sont traitées sur la même machine, on aura comme borne supérieure $\overline{S}$, tel que : $\overline{S} = \sum_{i=1}^{n}(p_i + S_i)$.

4.2 Problème $P2/p_i = p, S_i = s, k \leq 2/C_{max}$

Pour ce problème avec des temps de traitement et de préparation constants, nous proposons un algorithme polynomial pour le résoudre.

Algorithm A ;
début

- $\overline{M'}$= $\begin{cases} \frac{n(p+S)}{2} & \text{si } n \text{ est pair et } k = 2\,; \\ \frac{n(p+S)}{2} + S & \text{si } n \text{ est pair et } k = 1 \text{ avec } S \leq p\,; \\ n.S + p & \text{si } n \text{ est pair et } k = 1 \text{ avec } S > p\,; \\ \frac{(n+1)(p+S)}{2} & \text{si } n \text{ est impair et } k = 2\,; \\ \frac{(n+1)(p+S)}{2} & \text{si } n \text{ est impair et k} = 1 \text{ avec } S \leq p\,; \\ n.S + p & \text{si } n \text{ est impair et } k = 1 \text{ avec } S > p\,; \end{cases}$

- $t := 0$; $i := 1$; $j := 1$; $l := 1$;
- **répéter**
  si $(t + p_i + S_i) \leq \overline{M'}$ **alors**
  - Affecter la préparation de la tâche T_i à l'ouvrier l et la traiter sur la machine M_j à l'instant t ;
  - $t := t + p_i + S_i$; $i := i + 1$

  sinon **si** $k = 1$ **alors** $t := S$; $j := j + 1$
  sinon $t := 0$; $l := l + 1$; $j := j + 1$;
  fsi ;

  fsi ;

jusqu'à $i = n$;
fin.

Théorème 5. *L'algorithme A résout le problème* $P2/p_i = p, S_i = s, k \leq 2/C_{max}$ *en* $O(n)$.

Démonstration. La valeur $\overline{M'}$ constitue une borne inférieure pour la solution optimale. Comme les temps de traitement et les temps de préparation sont constants et égaux à p et s respectivement, on a :

Pour $k = 2$ et d'après la borne inférieure énoncée dans la proposition 3.1 précédente, nous avons $\overline{M'}$=max$\{\lceil \frac{1}{2}n(p+S)\rceil, p+S\}$=$\lceil \frac{1}{2}n(p+S)\rceil$=$\frac{n(p+S)}{2}$ si n est pair. Si n est impair, alors $C_{max} = \frac{(n-1)(p+S)}{2} + (p+S) = \frac{(n+1)(p+S)}{2}$ qui vient du fait que la préemption des tâches n'est pas autorisée. Il suffit donc de considérer le cas pair et d'ajouter le traitement de la n-ème tâche à la fin de l'ordonnancement.

Pour $k = 1$, nous avons un seul ouvrier. Donc la préparation simultanée de deux tâches n'est pas possible. Ainsi, la seconde machine sera libre pendant S unités de temps au début de l'ordonnancement.

Si le temps de préparation est supérieur au temps de traitement ($S \geq p$), alors la préparation des différentes tâches se fera en alterné sur les deux machines et chaque tâche sera traitée entre deux préparations successives. On obtient donc $n.S$ comme temps totale de préparation de l'ensemble des tâches. Finalement, on obtient $C_{max} = n.S + p$.

Si le temps de préparation est inférieur au temps de traitement ($S < p$), alors les tâches seront traitées comme dans le cas où $k = 1$ avec un décalage de S unités de temps sur la deuxième machine (la préparation de la deuxième tâche commence à $t = S$ sur la deuxième machine). On obtient donc, $C_{max} = \frac{n(p+S)}{2} + S$ dans le cas ou n est pair et $C_{max} = \frac{(n+1)(p+S)}{2}$ dans le cas où n est impair (la deuxième machine sera libre pendant p unités de temps à la fin de l'ordonnancement). □

5 Méthodes approchés

Dans cette partie, nous proposons six heuristiques pour résoudre le problème considéré. Toutes ces heuristiques ont un point en commun qui est une procédure A1 dont le rôle est d'ordonnancer les tâches préalablement rangées.

La procédure $A1$ se déroule de la manière suivante : nous supposons qu'à l'instant $t = 0$ tous les ouvriers et toutes les machines sont disponibles, et tant que l'ensemble des tâches n'est pas encore vide, on sélectionne une tâche T_i de cet ensemble et on l'affecte au premier ouvrier libre pour la traiter sur la première machine disponible à l'instant $t = 0$. Ensuite, on élimine cette tâche de l'ensemble des tâches et la préparation et le traitement de la tâche qui la suit commence à $t := \max\{\min\{O_l\}, \min\{C_j\}\}$.

L'objectif de cette procédure est donc de déterminer à quel instant, par quel ouvrier et sur quelle machine on lance la préparation et le traitement d'une tâche.

La procédure $A1$ s'écrit alors :

Procédure A1 ;
début
- $t := 0$; $O_l := 0$; $C_j := 0$;
- **tantque** $T \neq \emptyset$
  faire
  - Prendre la première tâche T_i de la liste et l'affecter au premier ouvrier libre (soit l) pour la traiter sur la première machine (soit M_j) disponible à l'instant t ;
  - $O_l := t + S_i$; $C_j := O_l + p_i$;
  - $T := T \setminus \{T_i\}$; $t := \max\{\min\{O_l\}, \min\{C_j\}\}$;

  fait ;
fin.

O_l représente la date de disponibilité de l'ouvrier l et C_j représente la date de disponibilité de la machine j

La complexité de cette procédure est de $O(n)$.

Description des heuristiques Hi : La première phase consiste à ranger les tâches selon un certain ordre, ensuite, faire appel à la procédure $A1$.

Algorithme Hi ;
début - Ranger les tâches suivant la règle **Ri** ;
- Appliquer la procédure $A1$;

fin.

Nous avons définis 6 règles de rangement :

R1 : Ranger les tâches selon l'ordre LPT des temps de traitement (notée LPT_{p_i})

R2 : Ranger les tâches selon l'ordre SPT des temps de traitement (notée SPT_{p_i})

R3 : Ranger les tâches dans l'ordre décroissant de leurs temps de préparation (notée LPT_{S_i})

R4 : Ranger les tâches dans l'ordre croissant de leurs temps de préparation (notée SPT_{S_i})

R5 : Ranger les tâches dans l'ordre décroissant de leurs temps de traitement et de préparation (notée $LPT_{S_i+p_i}$)

R6 : règle qui range les tâches dans l'ordre croissant de leurs temps de traitement et de préparation (notée $SPT_{S_i+p_i}$)

Toutes ces règles s'exécute en $O(nlogn)$.

6 Résultats expérimentaux

Afin de tester les heuristiques développées nous avons généré plusieurs instances possédant différentes caractéristiques. Par ailleurs, chaque instance est

caractérisée par un triplet (machines, tâches, ouvriers) tel que le nombre de machines $m \in \{2, 3, 5, 10\}$, le nombre de tâches $n \in \{10, 20, 50, 100, 1000\}$ et le nombre d'ouvriers à $k \in \{1, ..., m\}$.

En ce qui concerne les temps de traitement des tâches et les temps de préparation, ils sont générés aléatoirement suivant la loi uniforme en prenant plusieurs intervalles de temps.

Pour m, n et k fixés, nous avons généré et testé 100 instances différentes. Pour chaque combinaison de nombre de machines, nombre de tâches et nombre d'ouvriers on compte le nombre de fois où l'heuristique H donne de meilleurs solutions, aussi le nombre de fois où elle donne une solution égale à la borne inférieure. La table 1 illustre le résumé de ces tests.

Tab. 1. Résumé des tests

Tests		H1	H2	H3	H4	H5	H6
$p_i \in [1, 10]$	H. meilleure	5138	349	794	556	7515	106
$S_i \in [1, 5]$	$C_{max} = \overline{M}$	1111	240	580	315	2167	106
$p_i \in [1, 20]$	H. meilleure	4745	278	762	534	7101	100
$S_i \in [1, 10]$	$C_{max} = \overline{M}$	1001	207	537	262	1761	100
$p_i \in [1, 50]$	H. meilleure	4764	210	687	540	6806	100
$S_i \in [1, 20]$	$C_{max} = \overline{M}$	878	167	462	250	1446	100
$p_i \in [1, 100]$	H. meilleure	4548	233	726	441	6666	100
$S_i \in [1, 50]$	$C_{max} = \overline{M}$	807	177	459	196	1331	100

D'après cette expérimentation, il s'avère que pour toutes les instances, les heuristiques $H1$ et $H5$ donnent de meilleures solutions. De plus, l'heuristique $H5$ est généralement meilleure que l'heuristique $H1$. En effet, sur 10000 instances générées (selon la loi uniforme) avec les p_i et les S_i uniformement distribués dans [1, 10] et [1, 5] respectivement, l'heuristique $H5$ donne une meilleure solution dans 7515 cas et une solution optimale dans 2167 cas.

En ce qui concerne la deuxième série des tests où les $p_i \in [1, 20]$ et $S_i \in [1, 10]$, l'heuristique $H5$ est meilleure dans 7101 cas contre 4745 cas pour $H1$. $H5$ donne la solution optimale dans 1761 cas contre 1001 cas pour $H1$.

Pour la troisième série des tests où $p_i \in [1, 50]$ et $S_i \in [1, 20]$, l'heuristique $H5$ est optimale dans 1446 cas sur 6806. Et pour la dernière série de tests il y a 1331 cas où elle donne des solutions optimales sur les 6666 cas où elle est meilleure.

Finalement, d'après cette expérimentation, il apparaît que l'heuristique $H5$ qui range les tâches selon la règle $LPT_{S_i+p_i}$ est très intéressante que les autres, de plus elle apporte la solution en moins de 16 ms pour les problèmes de petite taille et en moins de 141 ms pour les problèmes de grande taille. Pour une meilleure lisibilité, la figure 1 transcrit quelques résultats trouvés par l'heuristique $H5$.

Pour évaluer les six heuristiques, nous avons utilisé l'indicateur de performance de chaque heuristique par rapport à la borne inférieure. Ce dernier appelé aussi déviation et est donné par le quotient $\frac{(H-BI)}{BI}$ avec H la valeur du critère

donnée par l'heuristique et BI la borne inférieure. Pour avoir des résultats exploitables, on génère aléatoirement 100 instances et on calcule la moyenne de $\frac{(H-BI)}{BI}$.

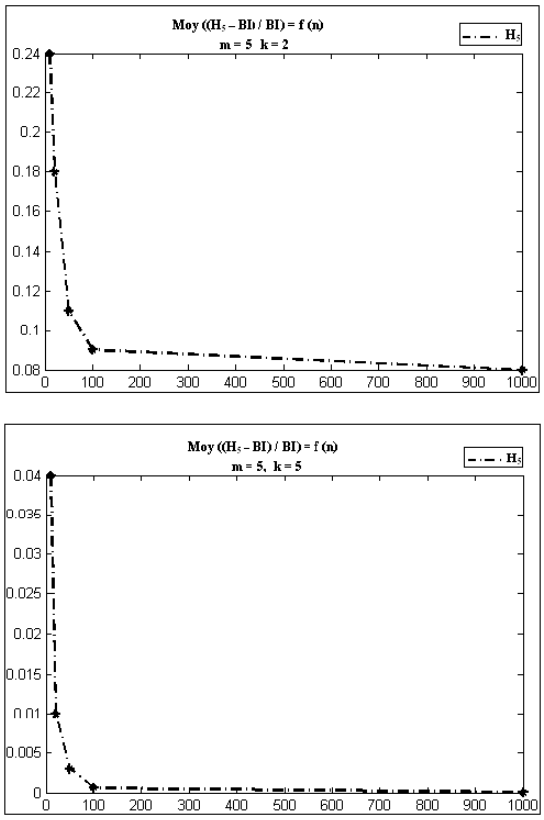


Fig. 1. Comportement de l'heuristique H5 par rapport à la borne inférieure pour $p_i \in [1, 10]$ et $S_i \in [1, 5]$

7 Conclusion

Dans cet article, nous avons traité un problème d'ordonnancement qui se pose dans un environnement à machines parallèles identiques en tenant compte des temps de préparation des tâches. L'objectif a été de déterminer des méthodes de résolution pour ce problème.

Nous avons, au cours de cet article, présenté une modélisation mathématique pour le problème et nous avons conclu que le problème général est NP-difficile.

Une borne inférieure, ainsi qu'un sous problème polynomial ont été proposés, comme nous avons développés six heuristiques.

Finalement, différents tests expérimentaux ont été réalisé sur des instances générées aléatoirement selon la loi uniforme. Après évaluation de ces heuristiques par rapport au nombre de fois où une heuristique l'emporte sur les autres, et le nombre de fois où la solution trouvée est égale à la borne inférieure, nous concluons que l'heuristique basée sur le rangement des tâches selon la règle $LPT_{S_i+p_i}$ est assez performante.

Le rapport de performance $\frac{4}{3} - \frac{1}{3m}$ de Graham pour $P2//C_{max}$ (règle LPT) reste valable pour le problème $Pm/S_i, k/C_{max}$ pour $k \geq m$ (règle $LPT_{S_i+p_i}$).

Dans les perspectives de nos études nous projetons d'étudier quelques extensions de ce problème, nous envisageons aussi d'améliorer les heuristiques et de résoudre le problème avec d'autres approches.

Références

1. Abdelkhodaee, AH., Wirth, A. : Scheduling parallel machines with a single server : some solvable cases and heuristics, Computers and Operations Research 29(3), 295-315 (2002).
2. Bettayeb, B., Kacem, I., Adjallah, K.H. : Ordonnancement sur machines parallèles identiques avec temps de préparation par famille : Application à la gestion des tâches de maintenance préventive. Université de Technologie de Troyes, $6^{ième}$ conférence froncophone de Modélisation et Simulation - MOSIM'06 (2006).
3. Boustta, M. : Minimisation du temps de fabrication sur une machine à injection avec réglages multiples, Thèse, University Laval. Québec (2003).
4. Cochran, J., Horng, SM., Fowler, J. : A multi- population genetic algorithm to solve multi-objective scheduling problems for parallel machines. Computer and Operations Research 30, 1087-1102 (2003).
5. Dell'Amico, M., Iori, M., Martello, S. : Heuristic Algorithms and Scatter Search for the Cardinality Constrained P//Cmax problem, journal of heuristics 10, 169-204 (2004).
6. Ghosh Jay, B. : Batch scheduling to minimize total completion time. Operations Research Letters 16, 271-275 (1994).
7. Karp, R. : Complexity of computer computations. R.E. Miller, J.W. Thatcher (eds.), Plenum Press, New-York (1972).
8. Koulamas, CP. : Scheduling two parallel semiautomatic machines to minimize machine interference. Computers and Operations Research 23(10), 945-956 (1996).
9. Kravchenko, SA. Werner, F. : Parallel machine scheduling problem with a single server. Mathematical and Computer Modelling 26(12), 1-11 (1997).
10. Mellouli, R., Sadfi, C., Kacem, I., Chu, C. : Ordonnancement sur machine parallèles avec contraintes d'indiponibilité. Université de Technologie de Troyes, $6^{ième}$ conférence froncophone de Modélisation et Simulation - MOSIM'06 (2006).
11. Zouba, M., Baptiste, P., Rebaine, D. : Ordonnancement de jobs sur deux machines parallèles en presence d'un seul opérateur, $6^{ième}$ conférence froncophone de Modélisation et Simulation - MOSIM'06 (2006).

Scheduling problem subject to compatibility constraints

Mohamed Bendraouche[1] and Mourad Boudhar[2]

[1] Faculty of Sciences, Saad Dahleb University,
Route de Soumaa, BP 270, Blida, Algeria
mbendraouche@yahoo.fr
[2] Faculty of Mathematics, USTHB University,
BP 32 El-Alia, Bab-Ezzouar, Algiers, Algeria
mboudhar@yahoo.fr

Abstract. The problem studied in this paper consists in finding the minimum makespan in a problem of scheduling jobs on identical parallel processors subject to compatibility constraints that some jobs cannot be scheduled simultaneously in any time interval. These constraints are modeled by a graph in which compatible jobs are represented by adjacent vertices. We study the complexity of this problem for bipartite graphs and their complements. We propose polynomial heuristics which are experimentally evaluated and compared.

Keywords. scheduling, compatibility graph, complexity, heuristics.

1 Introduction

In this work we study the problem of scheduling n independent jobs $J_1, J_2, ..., J_n$ non-preemptively on m identical parallel processors. Each job J_i has a processing time p_i and a release time r_i. Two jobs are compatible if they can be scheduled simultaneously. We suppose that there exist compatibility constraints between the jobs such that non-compatible jobs cannot be scheduled simultaneously in any time interval. These constraints are represented by a graph $G = (V, E)$ where V is the jobs set and $\{J_i, J_j\} \in E$ if and only if J_i and J_j are compatible. The graph G is called the compatibility graph. The aim is to minimize the makespan.subject to the compatibility constraints. Following the three parameters notation introduced in [1] our problem is denoted by $P/G = (V, E), r_i/C_{\max}$ and is sometimes referred to as the general problem. When m is fixed the problem is denoted by $Pm/G = (V, E), r_i/C_{\max}$.
This problem arises in the resource-constrained scheduling when the resources are non-sharable. Applications of this problem include the one of Baker and Coffman [2] presented for balancing the load in a parallel computation, others are mentioned in traffic intersection control, frequency assignment in cellular networks and session management in local area networks (see Halldorsson et al.[3]). Bodlaender and Jansen [4] have described an application derived from a problem of assigning operations to processors, where the operations are given

in a flow graph. Our interest to this problem initially comes from the following problem. **The Workshop Resource-Constrained** problem (W.R.C in short): there are n tasks $J_1, J_2, .., J_n$ to be executed by m workers in a workshop. Each task J_i requires a time p_i for its treatment and a subset of resources $R_i \subseteq R$ where R is the set of the available resources. The objective is to execute these tasks in a minimum time. If we regard the workers as processors and associate the graph $G = (V, E)$ in which the tasks correspond to the jobs set and that two jobs are compatible if they use no resources in common, one can verify that the W.R.C problem can be modeled as the problem $P/G = (V, E)/C_{\max}$.

Example 1. $n = 5, m = 2$, $R = \{res_1, res_2\}$. The resource requirements and the processing times are given in Table 1, the compatibility graph is represented on Fig. 1 (a). Fig. 1 (b) is the Gantt diagram representing a feasible schedule for this example, which is also optimal.

Table 1. Processing times and the resources requirements

J_i	J_1	J_2	J_3	J_4	J_5
p_i	1	2	1	3	1
R_i	$\{res_1, res_2\}$	$\{res_1\}$	$\{res_2\}$	$\{res_1\}$	$\emptyset$

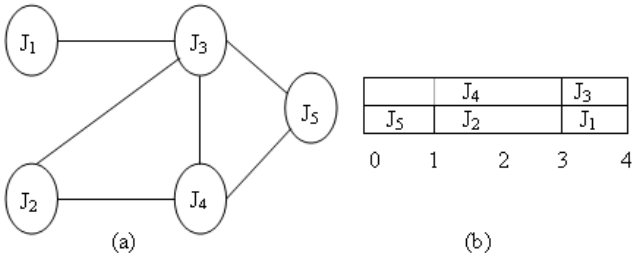


Fig. 1. (a)The compatibility graph of Example1 (b) The Gantt diagram of Example1

2 Related work and previous results

Among the related problems are:
1-The Mutual Exclusion Scheduling problem, M.E.S in short [2]: n unit-processing times jobs have to be scheduled on m processors in a minimum time subject to constraints represented by a conflict graph G such that adjacent jobs in G must be scheduled in disjoint time intervals.

2-Scheduling With Conflicts problem, S.W.C in short: Recently in [5] Guy Even, Magnus M. Halldorsson, Lotem Kaplan and Dana Ron have considered the problem of Scheduling With Conflicts (S.W.C) which consists in finding a minimum makespan on identical machines where conflicting jobs cannot be scheduled concurrently. The relationship between these problems and ours is as follows: The S.W.C problem with an arbitrary conflict graph $G = (V, E)$ is equivalent to the problem $Pm/\overline{G} = (V, \overline{E})/C_{\max}$ for which the compatibility graph is $\overline{G}$, the complement of G.
The M.E.S problem with fixed m is a special case of the S.W.C problem in which the processing times of the jobs are equal to 1. The M.E.S problem with conflict graph $G = (V, E)$ is equivalent to the problem $P/\overline{G} = (V, \overline{E}), p_i = 1/C_{\max}$ where the compatibility graph is $\overline{G}$, the complement of G. As far as the previous results are concerned, the problem $P2/G = (V, E), p_i = 1/C_{\max}$ can be reduced the maximum matching problem in G. The authors of [5] have established that the S.W.C problem can polynomially be solved when $m = 2$ and $p_i \in \{1, 2\}$, we deduce that the problem $P2/G = (V, E), p_i \in \{1, 2\}/C_{\max}$ is polynomial. By a reduction from Partition ([6]) the problem $P2/G = (V, E)/C_{\max}$ is NP-hard even if the compatibility graph is complete. The problem $Pm/G = (V, E), p_i = 1/C_{\max}$ is NP-hard for any $m \geq 3$ due to a result of B.S. Baker and E.G. Coffman [2]. The problem $Pm/G = (V, E), p_i \in \{1, 2, 3, 4\}/C_{\max}$ is APX-hard [5].

3 Bipartite graphs

In this section, we suppose that $m = 2$ and that the compatibility graph G is bipartite which is denoted by $G = (S_1 \cup S_2, E)$.

3.1 Case when $p_i \in \{1, 2\}, r_i \in \{0, r\}$

Theorem 1. *The problem $P2/G = (S_1 \cup S_2, E), p_i \in \{1, 2\}, r_i \in \{0, r\}/C_{\max}$ is NP-hard.*

Proof. Let Dbipartite be the decision problem associated with the above problem. We make a reduction from the 3- Dimentional Matching problem(3-DM). Let an arbitrary instance of 3-DM be given. We construct an instance of Dbipartite as follows: the jobs set is $V = V_M \cup V_Y \cup V_Z \cup V_D$ such that the jobs of V_M are in correspondence with the elements of M. Thus to any triplet $(x, y, z) \in M$ corresponds a job $J(x, y, z)$ in V_M. The elements of V_Y and V_Z are in correspondence with the elements of the sets Y and Z respectively. Thus each element $y \in Y$ corresponds to a job J_y of V_Y and each element $z \in Z$ corresponds to a job J_z of V_Z. Note that $|V_M| = |M|$, $|V_Y| = |V_Y| = q$. Suppose that $X = \{x_1, x_2, ..., x_q\}$. For each i ($i = 1, 2, ..., q$) let $M_i = \{(x, y, z) \in M : x = x_i\}$ so that $\bigcup_{i=1}^{q} M_i = M$ and let V_{M_i} be the subset of V_M corresponding to M_i. Note that $|V_{M_i}| = |M_i|$. To each job set V_{M_i} corresponds a set V_{D_i} of dummy jobs such that $|V_{D_i}| = |M_i| - 1$

$(i = 1, 2, ..., q)$. Let $V_D = \bigcup_{i=1}^{q} V_{D_i}$. One can see that $|V_D| = |M| - q$. The compatibility graph G is defined in such a way that each of the sets V_M, V_Y, V_Z, V_D form an independent set of vertices in G.

Every job $J(x, y, z)$ of V_M is compatible with both of the jobs J_y and J_z belonging to V_Y and V_Z respectively. For each $i = 1, 2, ..., q$, the jobs of V_{M_i} are joined to all of the jobs of V_{D_i}. Let G_M, G_Y, G_Z and G_D denote the subgraphs of G induced by V_M, V_Y, V_Z and V_D respectively. The processing times of the jobs of G_M and G_D are equal to 2 and those of G_Y and G_Z are equal to 1. The release times of the jobs of G are zero except those of G_D which are equal to $2q$, see $Fig.$ 2.

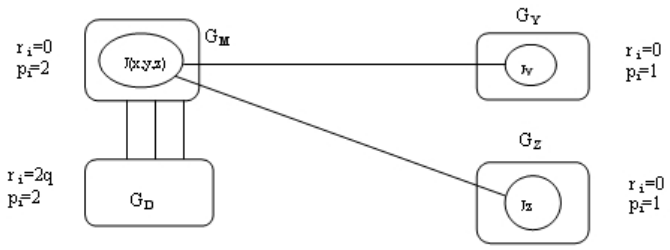


Fig. 2. The compatibility graph $G = (V, E)$

The graph G is bipartite since it can be written $G = (S_1 \cup S_2, E)$ where $S_1 = V_M$ and $S_2 = V_Y \cup V_Z \cup V_D$ and that §1 and S_2 are independent sets of vertices in G. Suppose that 3-DM has a solution. Consider the schedule σ defined as follows: let $\{J_1, J_2, .., J_q\}$ be the set of jobs of V_M corresponding to M' such that $J_i \in V_{M_i}$ $(i = 1, 2, ..., q)$. We schedule these jobs as well as their corresponding jobs of the graphs G_Y and G_Z in the time interval $[0, 2q]$ as indicated in Fig. 3.

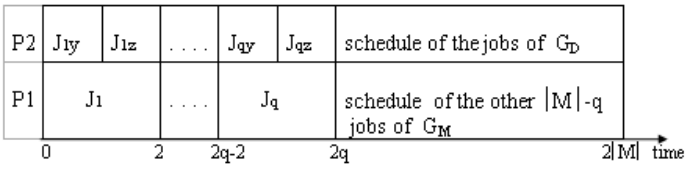


Fig. 3. Passage from a feasible solution of 3-DM to the one of Dbipartite

We schedule the remaining $|M| - q$ jobs of G_M in the interval $[2q, 2|M|]$ on the processor $P1$ as follows: we schedule these jobs into groups: the jobs of $V_{M_1} \backslash \{J_1\}$ first then the ones of $V_{M_2} \backslash \{J_2\}$ and so on till the jobs of $V_{M_q} - \{J_q\}$. We then schedule the jobs of G_D on $P2$, within the same interval in such a way that the jobs of $V_{M_i} - \{J_i\}$ are opposite to their corresponding jobs of V_{D_i} $(i = 1, 2, ..., q)$, see Fig. 3. The schedule that has just been obtained, say σ is therefore a feasible schedule for the problem Dbipartite with a makespan $C_{\max}(\sigma) \leq 2|M|$. Conversely, assume that there is a feasible schedule σ for the problem Dbipartite with a makespan $C_{\max}(\sigma) \leq 2|M|$. The jobs of G_D form an independent set of vertices of order $|M| - q$ in the compatibility graph G. Having each a processing time equal to 2 and a release time equal to $2q$, they must have been executed in the time interval $[2q, 2|M|]$. Without loss of generality we may suppose that they have been executed on $P2$. The remaining jobs include the ones of the graph G_M and those of the graphs G_Y and G_Z and their number is equal to $|M| + 2q$.

The remaining space equals $4|M| - (2|M| - 2q) = 2|M| + 2q$. Thus this space must entirely have been occupied by the $|M| + 2q$ remaining jobs of the graphs G_M, G_Y and G_Z. By construction of the graph G, q jobs of G_M must have been processed in the time interval $[0, 2q]$ along with their corresponding jobs of G_Y and G_Z respectively as indicated in Fig. 3. This corresponds to a set $M' \subseteq M$ with q elements, producing a solution of 3-DM. It is straightforward to see that the transformation used is polynomial and that DBipartite $\in$ NP, which completes the proof of Theorem 1. □

3.2 Unit processing times for S_1 and arbitrary for S_2

Next we consider the case when the processing times of the jobs of S_1 are equal to unity and the ones of S_2 are arbitrary. This problem is denoted by $P2/G = (S_1 \cup S_2, E)$, $p_{S_1} = 1/C_{\max}$. Suppose that $S_1 = \{T_1, T_2, ..., T_{|S_1|}\}$ and $S_2 = \{J_1, J_2, ..., J_{|S_2|}\}$.

Algorithm 1
Begin

1: Construct the network R=(V,U,c) as follows :
$V = \{s, p\} \cup S_1 \cup S_2$, $U = U_1 \cup U_2 \cup U_3 \cup \{u_r\}$ where $u_r = (p, s)$ is a return arc, $U_1 = \{(s, T_i), T_i \in S_1\}$, $U_2 = \{(T_i, J_j) \in S_1 \times S_2 : \{T_i, J_j\} \in E\}, U_3 = \{(J_j, p), J_j \in S_2\}$

2: Construct the arc capacity function c as follows:
if $(s, T_i) \in U_1 \Longrightarrow c(s, T_i) = 1$, if $(T_i, J_j) \in U_2 \Longrightarrow c(T_i, J_j) = 1$,
$c(u_r) = +\infty$, if $(J_j, p) \in U_3 \Longrightarrow c(J_j, p) = p_j$.

3: Determine a maximum feasible flow f^* in the network R.

4: Schedule the jobs of S_2 successively in the time interval $[0, \sum_{i=1}^{|S_2|} p_i]$ on $P1$.

5: Schedule the jobs T_i of S_1 such that $f^*(T_i, J_1) = 1$ opposite to the job J_1 in the time interval $[0, p_1]$, and the jobs T_i of S_1 such that $f^*(T_i, J_2) = 1$ opposite to the job J_2 in the time interval $[p_1, p_1 + p_2]$ and so on.

6: Schedule the remaining jobs of S_1 successively in the time interval $[\sum_{i=1}^{|S_2|} p_i, \sum_{i=1}^{|S_2|} p_i + |S_1| - f^*(u_r)\ ]$.

end

Remark 1. 1- If u is an arc of R, and f is a flow in R then $f(u)$ represents the value of the flow f on the arc u in R.
2- In step 3 of the above Algorithm we use J. Cherian and S.N. Maheshwari 'Algorithm [7] for obtaining a maximum flow.

Theorem 2. *Algorithm 1 solves the problem* $P2/G = (S_1 \cup S_2, E), p_{S_1} = 1/C_{\max}$ *polynomially in* $O(n^3)$.

Proof. One can easily see that Algorithm 1 returns a feasible schedule σ^* for the problem $P2/G = (S_1 \cup S_2, E), p_{S_1} = 1/C_{\max}$ with a makespan $C_{\max}(\sigma^*) = \sum_{i=1}^{|S_2|} p_i + |S_1| - f^*(u_r)$. Let us now show that for every feasible schedule σ we have: $C_{\max}(\sigma) \geq C_{\max}(\sigma^*)$. For let σ be an arbitrary feasible schedule. We shift the jobs of S_2 to the leftmost side of the Gantt diagram representing σ, as well as the jobs of S_1 that are scheduled opposite to the jobs of S_2 so that all these jobs would be scheduled in the time interval $[0, \sum_{i=1}^{|S_2|} p_i]$. By shifting and removing all the idle times in the Gantt diagram and by scheduling successively the remaining jobs of S_1 after the instant $\sum_{i=1}^{|S_2|} p_i$, we get a feasible schedule σ_1 and a feasible flow f_1 corresponding to σ_1 such that $C_{\max}(\sigma) \geq C_{\max}(\sigma_1) = \sum_{i=1}^{|S_2|} p_i + |S_1| - f_1(u_r)$. But since f^* is a maximum flow in R then $f_1(u_r) \leq f^*(u_r)$ and thus for every feasible schedule σ we have: $C_{\max}(\sigma) \geq \sum_{i=1}^{|S_2|} p_i + |S_1| - f^*(u_r)$. Since $\sum_{i=1}^{|S_2|} p_i + |S_1| - f^*(u_r) = C_{\max}(\sigma^*)$ (proved previously) we deduce that the schedule σ^* obtained by Algorithm 1 is optimal . The construction of the network R can be achieved in at most $O(|S_1|\,|S_2|) = O(n^2)$ time. By using J. Cherian and S.N. Maheshwari' Algorithm, step2 can be performed in $O(n^2\sqrt{|U|})$. As $|U| = O(|S_1|\,|S_2|) = O(n^2)$, step 2 can be achieved in $O(n^3)$. We deduce that the time complexity of Algorithm 1 equals $O(n^3)$. □

4 Complement of bipartite graphs

Now we consider the case when the compatibility graph G is the complement of a bipartite graph noted $G = (K_1, K_2; E)$ where K_1, K_2 are the cliques of G forming a partition of V.

H.L. Bodlaender, K. Jansen [4] have established that the problem $P/G = (K_1, K_2, E), p_i = 1/C_{\max}$ is NP-hard. We prove that the problem becomes polynomial if we restrict to odd r_i s. Consider the following greedy algorithm in which

for any job J_i, t_i represents its starting time and suppose $K_1 = \{J_1, J_2\ , .., J_s\ \}$ and $K_2 = \{J_{s+1}, J_{s+2}\ , .., J_n\}$.

Algorithm 2
Begin

1: **for** $i := 1$ to s **do**
2: $t_i := r_i$
3: **end for**
4: **for** $k := s + 1$ to n **do**
5: **if** (all the jobs of K_1 scheduled at time $t = r_k$ are compatible with J_k) **then**
6: $t_k := r_k$
7: **end if**
8: $t_k := r_k + 1$
9: **end for**

end

Theorem 3. *The problem $P/G = (K_1, K_2; E), p_i = 1, r_i$ odd $/C_{\max}$ with $m \geq n$ can polynomially be solved in $O(n^2)$ time.*

Proof. Let τ be the schedule obtained by Algorithm 2. First we prove that τ is an optimal feasible schedule. It is clear that τ is feasible. Let $L = max_{1 \leq i \leq n}\{t_i\}$, thus $C_{\max}(\tau) = L+1$. Two possibilities may happen: case1: L is even. Let J_k be a job satisfying $L = t_k$. Since t_k is even, by construction of the algorithm $J_k \in K_2$. On the other hand J_k must have been scheduled at step 8 of the algorithm and we have $t_k = r_k + 1$. Also, there must exist a job $J_i \in K_1$ scheduled at time $t_i = r_k$ which is is not compatible with J_k. Since $J_i \in K_1$ then $t_i = r_i$ and hence $r_i = r_k$. Since the jobs J_i and J_k are not compatible, with the condition $r_i = r_k$ it follows that $C^*_{\max} \geq r_k + 2 = t_k + 1 = L + 1 = C_{\max}(\tau)$.
case2: L is odd . Let J_p be a job satisfying $L = t_p$. t_p is odd , so J_p has not been scheduled at step 8 of the algorithm and hence $t_p = r_p$. This implies that $C^*_{\max} \geq r_p + 1 = L + 1 = C_{\max}(\tau)$. We deduce that the schedule τ is optimal.
Time complexity : the for-loop through steps 1-3 can be implemented in $O(n)$ time at worse, the for-loop through steps 4-9 requires at most $O(n^2)$ iterations. Then the time complexity equals $O(n^2)$. □

5 Heuristics for the problem $P/G = (V, E)/C_{\max}$

In this section, we propose some heuristics for the problem $P/G = (V, E)/C_{\max}$. The approach used is based on the list scheduling one used by R.L. Graham [8] for the problem $P//C_{\max}$ with some modifications.

5.1 Lower bounds on the optimal makespan

Let $Opt(G)$ denote the optimal makespan of the problem $P\,/G = (V, E)/\,C_{\max}$. A trivial lower bound on $Opt(G)$ is $LB_0 = Max\{\lceil(\sum_{j=1}^{n} p_j)/m\rceil, Max_{1 \leq j \leq n}\{p_j\}\}$.

On the other hand in [9] S. Sakai, M. Togasaki, K. Yamazaki have studied the Maximum Weighted Independent Set problem and have elaborated three greedy algorithms for this problem namely GWMIN, GWMAX and GWMIN2. Since non-adjacent jobs must be scheduled in disjoint time intervals for any feasible schedule, we deduce three lower bounds on $Opt(G)$ based on the three algorithms previously cited, say LB_1, LB_2 and LB_3 respectively. From the implementation we have observed that the second lower bound is weaker and therefore the overall lower bound on $Opt(G)$ is given by: $LB = Max\{LB_0, LB_1, LB_3\}$.

5.2 Definition of nine job lists

First we define the compatibility number (C.N for short) of a job to be the number of jobs which are compatible with it. Next we define nine job lists. If the jobs are sorted in order of increasing compatibility numbers, the list obtained is called $List1$. If the jobs are arranged in a decreasing order of their compatibility numbers, the list obtained is called $List2$. The following algorithm constructs $List3$.

Algorithm 3
Begin
1: Find a job J_1 from V with maximum C.N in G
2: **for** $j := 2$ to n **do**
3: $G' :=$ the subgraph of G induced by $V\backslash\{J_1, J_2, ..., J_{j-1}\}$.
4: Find a job J_j from $V\backslash\{J_1, ..., J_{j-1}\}$ with maximum C.N in G'.
5: **end for**
6: $List3 := (J_1, ..., J_n)$
end

If in the for-loop of Algorithm 3 we choose a job J_j with a minimum compatibility number in G' we obtain $List4$. $List5$ is a random permutation of $(1, 2, 3, ..., n)$. The next algorithm produces $List6$.

Algorithm 4
Begin
1: determine a job J_1 from V with maximum C.N
2: $j := 1$
3: **for** $j = 2$ to n **do**
4: **for** all $J_k \in V\backslash\{J_1, J_2, ..., J_{j-1}\}$ **do**
5: n_{J_k} :=the number of jobs in $V\backslash\{J_1, J_2, ..., J_{j-1}\}$ that are compatible with J_i
6: **end for**
7: determine among these jobs J_k a job J_j such that $n_{J_j} = max(n_{J_k})$
8: **end for**
9: $List6 := (J_1, ..., J_n)$
end

$List7$ is obtained by a similar algorithm to Algorithm 4 except that at step 7 we choose J_j satisfying $n_{J_j} = \min(n_{J_k})$. In $List8$ the jobs are sorted according

to the Shortest Processing Time rule called the SPT list. Finally $List9$ is based on the Longest Processing Time rule and is called the LPT list.

5.3 The proposed heuristics

Suppose (H) is a heuristic and that some jobs have already been scheduled by (H). Let $sch(H)$ denote the subset of jobs already scheduled by (H). If J_j is a job to be scheduled by (H) at some instant t let $\Psi(t, J_j)$ denote the set of jobs of $sch(H)$ having a part of processing in the time interval $[t, t+p_j]$. We say that the job J_j is ready for processing at time t if all the jobs of $\Psi(t, J_j)$ are compatible with J_j, otherwise J_j is not ready for processing at time t.
Next we present the Heuristic scheme that generates all the proposed heuristics. For any job J_j let p_j and c_j will represent the processing and the completion time of the job J_j respectively. For any processor Pi, s_i will denote the earliest time at which Pi becomes free.

Algorithm $HeuristicScheme$
Begin

choose a list L, say $L = (J_1, J_2, ..., J_n)$; $t := 0$
for all $i = \overline{1, m}$ **do**
  $s_i := 0$
end for
schedule job J_1 on processor $P1$ at time t
for $j = 2$ to n **do**
  while (J_j is not scheduled) **do**
    $schedule := false$
    find the first free processor Pk and its s_k; $t := s_k$
    for $r = j$ to n **do**
      if (J_r is unscheduled and ready at time t) **then**
        schedule J_r on Pk at t
        schedule $schedule := true$
        break
      end if
    end for
    if ($schedule := false$) **then**
      determine the set $\Psi(t, J_j)$
      $t := max\{c_i : J_i \in \Psi(t, J_j)$ and J_i is not compatible with $J_j\}$
    end if
  end while
end for

end

Each choice of L at step1 in this scheme induces a heuristic for the problem $P/G = (V, E)/C_{\max}$. Thus we derive nine heuristics that are called $H1, H2, ..., H9$ respectively.

Remark 2. 1- The break instruction in the above scheme makes the program exit from the for-loop in which it is contained.
2- The heuristic $H9$ corresponds to the LPT-list and is also referred to as the LPT-heuristic

6 Experimental results

All the heuristics have been coded with Matlab 7.0 and tested on a pentium IV PC computer with a 3.4 GHZ and 2 GB Ram. We have used two classes of randomly generated instances: instances with variable processing times and instances with unit processing times. Besides the parameters m and n we have used the parameter d which is the density (in percentage) of the compatibility graph. For both classes the different combinations of the parameters m, n and d are as follows: $m \in \{2, 5, 10, 20\}$, the values $m = 2$ and $m = 5$ are each associated with seven values of n such that $n \in \{10, 20, 50, 100, 250, 500, 1000\}$, the value $m = 10$ corresponds to seven values of n where $n \in \{10, 20, 50, 100, 250, 500, 1000\}$ and the last value $m = 20$ is associated with seven values of n such that $n \in \{30, 50, 100, 250, 350\ , 500, 1000\}$. The different values used for d are $10, 20, ..., 90$. For both classes the instances are generated as follows: for each triplet (m, n, d) for which n is different from 1000 we have generated 100 randomly generated instances according to the uniform distribution in which 25 instances are generated with $p_i \in \{1, 2, ..., 10\}$, 25 instances with $p_i \in \{1, 2, ..., 20\}$, 25 instances with $p_i \in \{50, 51, ..., 100\}$ and 25 instances with $p_i \in \{1, 2, ..., 100\}$. For $n = 1000$, we have done the same except that we have used 40 instances (rather than 100 instances) involving 4 sets of 10 instances each rather than of 25 instances as previously done. The generated instances have been grouped into three groups : low density, medium density and high density and these correspond to instances whose compatibility graph densities belong to $\{10, 20, 30\}, \{40, 50, 60\}$and $\{70, 80, 90\}$ respectively. In total we have used 22940 instances for each class.

After extensive experiments the first observation is that the three heuristics $H1, H4$ and $H9$ (that is the LPT-heuristic) are considerably better than the others in the case of variable processing times and that the heuristics $H1$ and $H4$ are the best in the unit processing times case. Therefore we only have compared these three heuristics in the case of variable processing times and the heuristics $H1$ and $H4$ in the unit processing times case. We have obtained four tables corresponding to $m \in \{2, 5, 10, 20\}$. The results for the case $m = 5$, variable processing times are represented in table 3. In each table the first row represents the number of times (in percentage) for which the corresponding heuristic is best. The MD and the AD rows represent the maximum and the average deviations from the lower bound respectively. The last row AT represents the average CPU time in seconds for each heuristic. The main observations are: in the case of variable processing times the LPT-heuristic is in general considerably better than both heuristics $H1$ and $H4$. In the unit processing times case is concerned we have observed that $H4$ is best relatively to $H1$. The deviations of the LPT-

heuristic (v.p.t case) and those of the heuristic $H4$ (u.p.t case) are summarized in table 2 lines 1-3, lines 4-8 respectively .
Analysis of the results obtained: In the case of variable processing times the LPT-heuristic is expected to lead to better solutions compared to the others since it takes into account the processing times of the jobs and in the same checks the compatibility, in contrast to the others which use only the compatibility. This heuristic is somewhat a compromise between the processing times and the compatibility. However, in the unit processing times case the compatibility has an effect since the processing times are equal. The superiority of the heuristics H1 and H4 can be explained by the fact that both of them give the priority to the jobs with less compatibility numbers to pass first since it is more likely that the schedule of the jobs with higher compatibility numbers keeps the maximum completion time reduced.

Table 2. Deviations of the LPT-heuristic(v.p.t) and heuristic $H4$ (u.p.t)

Low den.	$n \leq 100, m \leq 20$ Av-dev ≤ 0.664, Max-dev $= 1.387$
Med. den.	$n \leq 1000, m \leq 10$ Av-dev ≤ 0.972, Max-dev $= 1.634$
High den.	$n \leq 1000, m \leq 20$ Av-dev ≤ 0.987, Max-dev $= 1.980$
Low den.	$n \leq 1000, m \leq 10$ Av-dev ≤ 1.368, Max-dev $= 1.964$
Med. and High.den.	$n \leq 1000, m \leq 10$ Av-dev ≤ 0.789, Max-dev $= 1.4$
Low.den.	$m = 20, n \leq 100$ Av-dev ≤ 0.684, Max-dev $= 1.308$
Med.den.	$m = 20, n \leq 100$ Av-dev ≤ 1.029, Max-dev $= 1.5$
High.den.	$m = 20, n \leq 1000$ Av-dev ≤ 0.702, Max-dev $= 1.6$

7 Conclusion

In this paper we have studied the problem of scheduling jobs non-preemptively on identical parallel processors and the aim is to minimize the makespan subject to the compatibility constraints. We have studied the complexity of the problem for bipartite graphs and their complements. In addition we have devised several polynomial time heuristics with acceptable performances for the problem without release times. The effectiveness of such heuristics have been evaluated by extensive experiments on randomly instances, showing that the LPT-heuristic outperforms all the proposed ones when the processing times are variable and that $H4$ is the best in the case of unit processing times.

References

1. Graham R.L., Lawler E.L., Lenstra J.K. and Rinnooy Kan A.H.G. Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discrete Math 5: 287–326 (1979).

Table 3. Experimental results m=5, v.p.t case

	Low dens.			**Med. dens.**			**High dens.**		
n	**H1**	**H4**	**H9**	**H1**	**H4**	**H9**	**H1**	**H4**	**H9**
n=10 Best	42.333	73.333	76.333	41.000	62.000	59.333	78.333	77..000	69.333
MD	0.657	0.543	0.355	0.625	0.667	0.696	0.500	0.500	0.684
AD	0.105	0.052	0.048	0.150	0.110	0.118	0.078	0.080	0.115
AT	0.008	0.007	0.009	0.008	0.008	0.008	0.005	0.004	0.005
n=20 Best	14.000	38.667	64.333	26.667	40.667	46.333	46.667	40.333	33.667
MD	0.614	0.471	0.568	1.089	0712	1.100	0.875	0.600	0.716
AD	0.200	0.144	0.116	0.308	0.272	0.266	0.167	0.181	0.211
AT	0.020	0.018	0.022	0.014	0.020	0.015	0.009	0.013	0.013
n=50 Best	4.333	28.000	68.000	29.667	38.667	37.333	37.000	35.000	39.000
MD	1.169	1.003	0.991	1.000	0.868	0.825	0.286	0.337	0.349
AD	0.488	0.420	0.372	0.473	0.450	0.458	0.074	0.075	0.091
AT	0.047	0.047	0.041	0.035	0.035	0.034	0.019	0.023	0.025
n=100 Best	3.333	18.667	78.667	34.000	40.333	29.333	24.333	33.000	51.333
MD	0.719	0.708	0.668	0.109	0.094	0.131	0.109	0.094	0.029
AD	0757	0.697	0.628	0.291	0.282	0.299	0.032	0.028	0.029
AT	0.166	0.174	0.154	0.115	0.110	0.090	0.029	0.023	0.027
n=250 Best	4.333	13.000	82.667	42.000	54.667	12.000	14.333	13.333	75.000
MD	1.428	1.384	1.324	0.321	0.287	0.305	0.044	0.027	0.054
AD	0.838	0.796	0.723	0.088	0.083	0.113	0.012	0.011	0.006
AT	1.146	1.216	1.059	0.538	0.517	0.416	0.066	0.054	0.061
n=500 Best	6.000	15.667	79.000	37.667	51.333	18.333	12.000	9.000	81.000
MD	1.275	1.196	1.070	0.132	0.122	0.159	0.020	0.013	0.021
AD	0.682	0.659	0.600	0.025	0.023	0.046	0.006	0.006	0.002
AT	5.696	6.042	5.105	1.673	1.587	1.350	0.167	0.135	0.151
n=1000 Best	13.333	26.667	60.833	38.333	35.833	37.500	5.000	7.500	88.333
MD	0.974	0.961	0.851	0.027	0.037	0.070	0.0080	0.006	0.010
AD	0.483	0.472	0.436	0.005	0.005	0.015	0.003	0.003	0.001
AT	31.039	32.808	26.965	5.126	4.884	4.536	0.488	0.392	0.416

2. Baker B.S., Coffman E.G. Mutual Exclusion Scheduling. Theoretical Computer Science 162: 225–243 (1996).
3. Halldorson M M., Kortsarz G., Proskurowski A., Salman R., Shachnai H., and Telle J.A. Multicoloring trees. Informationand Computation 180(2), 113-129 (2003).
4. Bodlaender H.L., Jansen K. Restrictions of graph partition problems part I. Theoretical Computer Science 148: 93–109 (1995).
5. Even G., Halldorson M M., Kaplan L. and Ron D. Scheduling with conflicts: online and offline algorithms. J. Sch. 12: 199-224 (2009).
6. Lenstra J.K., Rinnooy Kan A.H.G. Computational complexity of discrete optimization. In Lenstra JK and AHG Rinnooy Kan and P Van Emde Boas(eds), Interfaces Between Computer Science and Operations Research, Proceedings of a symposium held at the Matematisch Centrum, Amsterdam: 64–85 (1979).
7. Cherian J., Maheshwari S.N. Analysis of preflow push algorithms for maximum network flow. SIAM Journal on Computing 18: 1057–1086 (1989).
8. Graham R.L. Bounds for certain multiprocessing anomalies. Bell System Technical Journal 45:1563–1581 (1966).
9. Sakai S., Togasaki M, Yamazaki K. A. note on greedy algorithms for maximum weighted independent set problem. Discrete Applied Mathematics 126: 313–322 (2003).

Séparation et Evaluation pour le problème d'ordonnancement avec blocage.

Abdelhakim Ait Zai[1], Abdelkader Bentahar[1], Hamza Bennoui[1],

Mourad Boudhar[2] et Yazid Mati[3]

[1] Faculté d'Electronique et d'Informatique, Département d'Informatique, USTHB, BP 32 El Alia Alger, Algérie.

[2]Faculté de Mathématiques, Département de Recherche Opérationnelle, USTHB, BP 32 El Alia Alger, Algérie.

[3]Université d'Elkassime Arabie Saoudite.

Résumé : Nous traitons, dans ce papier, du problème d'ordonnancement Job shop avec blocage connu pour être NP-difficile. Après une présentation du problème posé, nous proposons de le résoudre d'une manière exacte en utilisant une méthode par séparation et évaluation SEP. Pour obtenir une solution optimale en un temps raisonnable, nous avons amélioré la méthode en utilisant une technique de séparation originale basée sur les graphes alternatifs. Nous avons utilisé aussi deux méthodes différentes pour l'évaluation d'une solution de la méthode SEP. Dans la dernière partie de ce papier, nous discutons les résultats des deux méthodes.

Mots clés : Job shop, blocage, séparation et évaluation, graphe alternatif.

1. Introduction

En général, les problèmes d'ordonnancement modélisent des problèmes d'ateliers de fabrication; ces ateliers sont composés principalement de ressources (machines, ouvriers et matière première) utiles à la fabrication de plusieurs types de produits. La résolution de ces problèmes revient, dans les méthodes classiques, à déterminer les dates de début et de fin de chaque opération composant les gammes opératoire des produits. Les nouvelles méthodes de résolution, s'orientent vers la recherche de l'ordre des séquences de passage sur chaque machine. Cette technique est plus pratique et rapide pour trouver l'ordonnancement nécessaire. Ces nouvelles techniques, ne change pas la complexité du problème d'ordonnancement classé NP-difficile. Les méthodes de résolution des problèmes d'ordonnancement d'ateliers se divisent en deux principales classes :

Les méthodes exactes ont l'avantage de garantir l'optimum en parcourant toutes les combinaisons possibles d'une manière intelligente. Comme inconvénient majeur, ces méthodes sont caractérisées par un temps de réponse non raisonnable pour des instances de moyenne et grande tailles, la pratique montre que l'utilisation d'un ordinateur (machine de calcul) est insuffisant (espace mémoire, vitesse de calcul) devant une complexité de plus en plus grande.

Concernant les méthodes approchées elles sont très rapides; mais les solutions qu'elles retournent ne sont pas forcements optimales d'où le nom approchées. La validation de ces méthodes nécessite l'utilisation des méthodes exactes, donc on compare les résultats des méthodes approchées à ceux trouvés par les méthodes exactes, puis après validation on lance les méthodes approchées sur des exemples plus grands, mais rien n'assure qu'elles vont converger.

Le problème d'ordonnancement à deux machines F2\\Cmax a été résolu de façon optimale par Johnson. Cet algorithme, et ses variantes, est un moyen rapide d'optimiser l'ordonnancement de processus simples. En effet, il est facile de voir que la complexité de cet algorithme est en O(n log n). Les résultats obtenus par Johnson sont devenus des classiques dans la théorie de l'ordonnancement. Par contre, les problèmes de job shop sont en général NP-complets, même si l'atelier est simple. En effet, Graham R et al. [14] ont montré que les ateliers possédant plus de trois machines, ou un nombre de tâches supérieur ou égal à trois, sont NP-difficiles même si la préemption est permise. De même pour les problèmes à deux machines, dès qu'il y a recirculation, ils deviennent fortement NP-difficiles.

Dans ce papier nous présentons le problème d'ordonnancement avec blocage et sa résolution par une méthode exacte de type séparation et évaluation (SEP). Ce problème est un job shop qui prend en considération le cas où les machines n'ont pas d'aires de stockage, ainsi si la machine suivante d'une opération donnée n'est pas libre, cette opération va rester sur la machine courante même si elle a terminé son traitement, on dit alors que cette machine est bloqué.

2. Position du problème

L'ordonnancement d'atelier à cheminements multiples (Job Shop) est généralement formulé comme suit :
Soit un ensemble de travaux de cardinalité n noté $\mathbf{J=\{J_1,\ldots,Jn\}}$, à traiter par un ensemble de machines de cardinalité m noté $\mathbf{M=\{M_1,\ldots,M_m\}}$; chaque travail j a un sous ensemble ordonné d'opérations noté $\mathbf{Oj=\{oj_1,\ldots\ldots,oj_{pj}\}.}$
Pour cela des contraintes de précédences sont définis :

- ✓ S'il y a une contrainte de précédence entre deux opérations $\mathbf{o_i}$ et $\mathbf{o_j}$ $\mathbf{(o_i \rightarrow o_j)}$ alors $\mathbf{o_j}$ ne peut commencer avant que $\mathbf{o_i}$ se termine. Donc l'ordre de la gamme opératoire est obligatoire.
- ✓ Chaque machine ne peut traiter qu'une seule opération à la fois et un travail ne peut occuper plus qu'une machine en même temps.
- ✓ Les séquences de passage des opérations sur une même machine sont inconnues et à déterminer dans le but de minimiser la durée total de production $\mathbf{C_{max}}$ tout en respectant les contraintes du problème**.**

2.1. Les contraintes

L'atelier de fabrication est soumis à des contraintes technologiques. Ces contraintes touchent à la fois les possibilités d'utilisation des machines et les liens (contraintes) qui peuvent exister entre les opérations. Ces contraintes sont les suivantes :

a) Les machines sont indépendantes les unes des autres (pas d'utilisation d'outil commun par exemple).

b) Une machine ne peut exécuter qu'une seule opération à un instant donné.
c) Chaque machine est disponible pendant toute la durée de l'ordonnancement, en particulier, les pannes ne sont pas prises en compte dans ce modèle.
d) Une opération en cours d'exécution ne peut pas être interrompue (la préemption des opérations n'est pas permise).
e) Les travaux sont indépendants les uns des autres. En particulier, il n'existe aucun ordre de priorité attaché aux travaux.

D'autres types de contraintes peuvent êtres prises en considérations, ceci dépendra du problème posé et du cas à étudier, pour notre cas nous avons choisi les contraintes ci-dessus.
En plus de la relation de précédence on trouve encore d'autres contraintes; celles ci dépendent de l'environnement de production, dans certains ateliers de production, deux opérations peuvent se réalisées en séquentielle ou en parallèle, ceci dépend du fait de partager la même ressource; dans ce cas, une règle de priorité doit être définie entre ces opérations. Le passage séquentiel des opérations sur les machines risque d'être bloqué si celles-ci n'admettent pas d'espace de stockage pour les opérations traitées, ainsi l'opération suivante ne peut commencer que si la première quitte cette ressource partagée, cette contrainte est appelée « contrainte de blocage ».

2.2. La contrainte de blocage

Soient **i et j** deux opérations concurrentes sur la même machine M(i)=M(j) et $\sigma(i)$, $\sigma(j)$ leurs successeurs directs, respectivement. Dans le cas où la machine M n'admet pas un espace de stockage, alors le passage séquentiel de j sur M dépend de la fin de traitement de i sur la même machine, pendant ce temps, j restera bloquée jusqu'à ce que i quitte M; cette situation est représentée en reliant $\sigma(i)$ avec j, $\sigma(j)$ avec i par une paire d'arcs alternatifs *u1 et u2* tels que : *u1*=(σ(i),j) et *u2*=(σ(j),i). Chaque arc alternatif u_l représente le fait que l'opération terminale ne peut commencer avant le début de l'opération initiale, ainsi, $t_{\sigma(i)} <= t_j$ et $t_{\sigma(j)} <= t_i$.
La pondération des arcs alternatifs est souvent nulle, l'exception est faite avec la dernière opération *k* de chaque travail qui n'admet pas de successeur, donc $\sigma(k)$ n'existe pas, dans ce cas là on relie directement ces sommets (opérations) avec leurs concurrents sur la même machine en utilisant des arcs alternatifs pondérés, cette pondérations est strictement positif, dont la valeur est égale à la durée de traitement p_k du sommet initial de chaque arc alternatif.

3. Etat de l'art

Depuis la première apparition du problème de job shop dans la littérature un grand effort a été réalisé pour la conception des algorithmes B&B. Parmi les travaux remarquables dans se domaine d'optimisation combinatoire on trouve :

L'algorithme de Carlier et Pinson (1989) qui résout le problème de complexité 10x10 proposé par Fisher et Thompson, ensuite, Pinson et Carlier introduisent en 1989, toujours, la possibilité de fixer les arcs alternatifs sans séparation, ce qui rend leur algorithme très efficace. La majorité des travaux qui sont arrivés par la suite sont basés sur leurs résultats.

Concernant l'approche parallèle on trouve l'implémentation parallèle de la méthode B&B proposé par Pergaard et Clausen. Cet algorithme basé sur les résultats de Carlier et Pinson et Brucher et al.

La contrainte de blocage dans l'ordonnancement d'atelier a intéressé beaucoup de chercheurs ces dernières années à cause de ces applications dans l'industrie, l'agriculture les hôpitaux…etc.

Le problème d'ordonnancement flow shop avec 3 machines est NP-difficile au sens fort (Reddi et Ramamorthy 1972). Pour le cas de deux machines, le problème peut être réduit en un cas spécial du problème de voyageur de commerce (PVC) et sa résolution en temps polynomial avec l'algorithme de Gilmore et Gomory.

Pour m>2 le problème flow shop est considéré comme NP-difficile, pour cela plusieurs heuristiques sont développées pour résoudre ce problème : Xianpeng Wang et Lixin Tang [3] ont proposé un algorithme de recherche tabou pour résoudre le problème flow shop hybride avec capacité de stockage limitée où chaque travail doit passer par N stages et chaque stage contient m_j machines parallèles, pour construire une solutions réalisable ils ont utilisé une procédure GCP (Greedy Constructive Procedure) qui considère les espaces de stockage entres deux stages successives comme des stages contenant des machines parallèles et les temps de traitement de chaque travail sur ces machines est nul.

Un algorithme PSO hybride utilisant la procédure NEH pour construire une solution initiale est proposé par Bo Liu, Ling Wang et Yi-Hui Jin pour résoudre le problème d'ordonnancement flow shop avec capacité de stockage limitée. P. Débora Ronconi et luis R.S Henriques [9] ont étudié la minimisation du retard total pour le problème d'ordonnancement flow shop avec contrainte de blocage, un algorithme (FPD : Fitting Processing timess and Due dates) et deux méta heuristiques GRASP (Greedy Randomized Adaptative Search Procedures) sont proposés pour trouver une solution approchée.

Jozef Grabowski et Jaroslaw Pempera [8] ont utilisé la notion des blocks et anti-blocks pour construire un algorithme de recherche tabou pour la résolution du problème d'ordonnancement flow shop avec contrainte de blocage.

Ling Wang, Liang Zhang et Da-Zhong Zheng [7] ont proposé un algorithme génétique hybride pour le problème d'ordonnancement flow shop avec capacité de stockage limitée pour construire une population initiale ils ont utilisé la méthode NEH, ils ont appliqué la procédure de recherche locale sur la population avec une probabilité $(1-p_m)$ où p_m est la probabilité de mutation.

S. Martinez, S. Dauzère-Pérès, C. Guéret, Y. Mati et N. Sauer [4] ont étudié la complexité du problème flow shop avec la nouvelle contrainte de blocage où la machine k reste bloquée jusqu'à ce que le travail commence sont traitement sur la machine k+2, ce problème a beaucoup d'applications dans l'industrie et l'agriculture et peut être modélisé par un graphe disjonctif.

A. Soukhal, A. Oulamara et P. Martineau [2] ont étudié la complexité du problème d'ordonnancement flow shop à deux machines avec contrainte de transport, les travaux traités sur les deux machines sont transportés par des camions de capacité limitée. Dans leurs articles ils ont montré la difficulté de quelques cas avec la contrainte de blocage entre les machines.

A. Soukhal et P. Martineau [5] proposent un modèle mathématique et un algorithme génétique pour résoudre le problème d'ordonnancement flow shop avec un rebot relié avec toutes les machines, le rebot peut traiter un seul travail à la fois.

Débora P. Ronconi [6] a proposé une heuristique constructive appelée MinMax pour résoudre le problème flow shop avec contrainte de blocage, cette heurstique est comparée avec d'autres algorithmes comme l'algorithme PF et l'algorithme NEH…

W. Henry, Thornton et John L. Hunsucker [10] ont proposé un algorithme appelé Nis-heuristique pour la résolution du problème d'ordonnancement flow shop avec processeurs multiple et indisponibilité d'espace de stockage, les travaux doivent passer par m stages et chaque stage contient M_j machines parallèles. L'algorithme est comparé avec d'autres heuristiques qui résolvent le même problème.

Vince Craffa, Stefano Lanes, Tapan P. Bagchi et Chelliah Striskandarajah [11] ont utilisé un algorithme génétique pour résoudre le problème d'ordonnancement flow shop avec contrainte de blocage. Ils ont utilisé le lien entre la contrainte de blocage et la contrainte de sans attente pour construire une fonction d'évaluation de la fitness des éléments générés par l'algorithme génétique.

Pour le problème job shop avec contrainte de blocage : Alessandro Mascis and Dario Pacciarelli [1], ont proposé une modélisation par graphe alternatif avec quelque propriétés sur cette catégorie des graphes qui est une généralisation des graphes disjonctifs; des méthodes heuristique est une autre par séparation et évaluation sont présentées dans cet article pour le problème d'ordonnancement job shop avec contraintes de blocage et de sans attente.

Yazid Mati, Nidhal Rezg et Xialon Xie [12] ont proposé une recherche tabou pour le problème d'ordonnancement job shop avec contrainte de blocage.

Heinz Gröflin et Andreas Klinkert [13] ont développé un algorithme recherche tabou pour le problème d'ordonnancement job shop généralisé avec contrainte de blocage avec prise en compte du temps d'installation et du temps de nettoyage.

4. Modélisation du problème

Le problème job shop traditionnel suppose que l'espace de stockage est disponible et illimité.

Dans la pratique il existe plusieurs exemples dans l'industrie où la capacité de stockage est limitée ou indisponible ce qui cause la contrainte de blocage entre deux opérations successives.

La machine qui traite l'opération d'un travail donnée reste bloquée après sa fin de traitement jusqu'à ce que l'opération prochaine de ce travail commence son traitement sur la machine suivante, cette contrainte est appelée contrainte de blocage.

4.1 Modélisation par graphe disjonctif

Notons $\sigma(o_i)$ l'opération qui suit o_i dans le même travail. Les deux figures ci-dessous présentent la différence entre les arcs disjonctifs d'un problème d'ordonnancement job shop classique et les arcs alternatifs d'un problème d'ordonnancement avec contrainte de blocage :

4.1.1. Arcs disjonctifs

Nous distinguons trois types de contraintes disjonctifs entre chaque couple d'opérations qui s'exécutent sur la même machine :

a/ Arcs disjonctifs classiques

Ce type de contraintes apparaît lorsque la machine contient un espace de stockage ou dans le cas de deux opérations terminales.

Dans ce cas, les deux opérations o_i et o_j qui doivent s'exécuter sur la même machine m_k sont reliées par deux arcs disjonctifs de o_i vers o_j et de o_j vers o_i ce qui signifie que l'opération o_j ne peut commencer qu'après la fin d'exécution de l'opération o_i sur la machine ou le contraire.

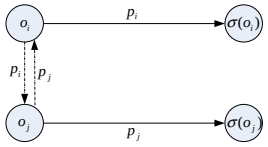


Figure 1. Arcs disjonctifs dans un job shop classique

b/ Arcs disjonctifs de blocage (arcs alternatifs)

Ce type de contraintes apparaît lorsque la machine m_k ne contient pas d'espace de stockage.

Dans ce cas, les deux opérations o_i et o_j sont reliées par deux arcs alternatifs de $\sigma(o_i)$ vers o_j et de $\sigma(o_j)$ vers o_i ce qui signifie que l'opération o_j ne peux commencer qu'après le début de l'opération $\sigma(o_i)$, ceci assure que la machine m_k est libre.

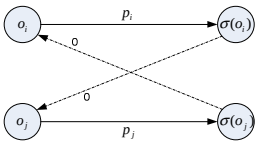


Figure 2. Arcs de blocage dans un job shop avec contrainte de blocage.

c/ Arcs croisés

Ce type de contraintes apparaît lorsque la machine m_k ne contient pas un espace de stockage et l'une des deux opérations est une opération terminale.

Soit o_i l'opération terminale : Dans ce cas, les deux opération o_i et o_j sont reliées par deux arcs alternatifs de o_i vers o_j et de $\sigma(o_j)$ vers o_i ce qui signifie que l'opération o_j ne peux commencer qu'après la fin d'exécution et de o_i, et o_j ne peux commencer qu'après le début d'exécution de l'opération $\sigma(o_i)$ ceci assure que la machine m_k est libre.

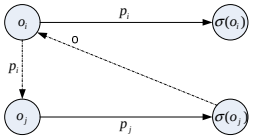
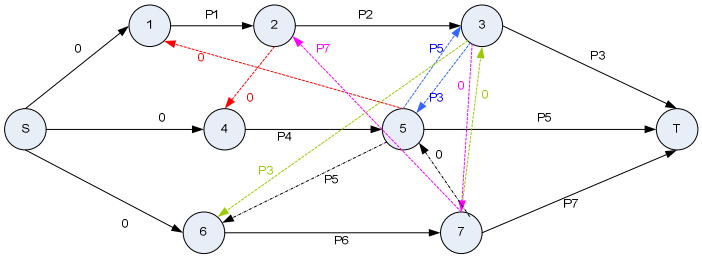


Figure 3. Arcs croisés dans un job shop avec contrainte de blocage

Exemple : Soit à ordonner 3 travaux j_1, j_2, j_3 sur 3 machines m_1, m_2, m_3

Travaux	La séquence des machines
1	1, 2,3
2	1,3
3	3, 2

Pour simplifier l'exemple nous considérons l'ensemble de toutes les opérations Numérotées de 1 à 7 :



S : est une tâche fictive qui représente la source, elle est liée à toutes les opérations qui sont prêtes à t=0.

T : est une tâche fictif qui représente le puits, elle est liée à toutes les opérations qui n'ont pas un successeur.

Remarque : nous rappelons que les poids des arcs altératifs sont nuls.

5. Résolution du problème posé

Dans cette partie nous décrivons la méthode exacte B&B que nous avons utilisé comme moyen de résolution; qui est un algorithme de création et parcourt d'un arbre n-aire « arbre de recherche », la création de l'arbre est assurée par le processus de séparation alors que le parcourt est assuré à chaque évaluation.

Dans la littérature plusieurs manières de parcourt sont citées : le parcourt en largeur, le parcourt en profondeur, le meilleur d'abord etc. On trouve même des heuristiques développées rien que pour la sélection d'un nœud parmi les autres.

5.1. La séparation

Le point de départ est un nœud qui représente la modélisation du problème. Nous avons évité de porter dans ce nœud toute information qui ne change pas pendant la séparation (les arcs conjonctifs, les données, …etc), cette manière de modéliser évite le gaspillage de l'espace mémoire; donc on a fait la différence entre deux types de données :

- Le problème initial : qui est fixe.
- Le sous-problème : représentant les contraintes sur les quelles le processus de séparation travaillera (les arcs alternatifs). Ce sont des données dynamiques qui changent avec le travail.

L'idée de séparation qu'on a utilisée se base sur l'énumération implicite de toutes les combinaisons possibles qu'on peut avoir en ordonnançant les travaux (opérations) concurrents sur chaque machine.

Le fait de séparé sur l'ordre entre les opérations donne un espace de recherche beaucoup plus petit (car lors du choix de l'opération qui doit passer en premier sur la machine concernée permet en réalité de fixer plusieurs arcs alternatifs en même temps au lieu d'un seul arc à fixer à chaque séparation, et nous évitons ainsi, tous les cas de non réalisabilité) que d'autres manières de séparation (exemple : la fixation pour chaque couple d'arcs alternatifs d'un seul arc dans les cas possibles entres les opérations concurrentes avec celle choisie). Les ordonnancements générés comportent plusieurs situations de violation des contraintes de précédences; pour cela une vérification de l'admissibilité doit se faire après chaque séparation.

Dans les problèmes de job shop idéales le circuit est un indicateur suffisant de non réalisabilité de la solution; mais dans notre cas on fait la différence entre un circuit nul est un circuit positif.

5.2. L'évaluation

Comme méthode d'évaluation nous avons utilisé l'algorithme de Bellman qui va donner à chaque opération sa date de commencement sur la machine qu'elle demande. La date de Fin d'exécution de la dernière opération représente la valeur de la fonction objectif notée C_{max}. Cette fonction ne sera appelée qu'après la mise à niveau du graphe avec une réduction (composante fortement connexe) s'il est nécessaire, assurant par ça un temps d'exécution très réduit.

Ainsi dans cette partie nous avons exploité l'algorithme de mise à niveau d'un graphe pour vérifier si le graphe n'admet pas de circuits. Dans le cas contraire on applique l'algorithme de forte connexité sur la partie restante du graphe où l'algorithme de mise à niveau c'est arrêté, la composante fortement connexe Cf détectée contiendra automatiquement un circuit, si celui-ci est de longueur positif alors on arrête la solution est non réalisable et nous stérilisons, par contre si la longueur du circuit est nulle, dans ce cas là, nous avons un cas de swapping, nous affectons, alors, à tous les sommets de la composante Cf la même pondération qui est le max des pondérations de x $\forall x \in$ Cf. Ensuite nous réduisons cette composante Cf en un seul sommet, et ainsi de suite, ceci nous permettra de garder le graphe sans circuit afin que nous puissions continuer à utiliser l'algorithme de Bellaman pour calculer la valeur du plus long chemin qui n'est rien d'autre que notre évaluation.

Une autre manière d'évaluer aussi, est l'algorithme de recherche de plus long chemin de Ford. Cet algorithme prend en considération les circuits d'une manière automatique contrairement à l'algorithme de Bellman.

5.3. La stérilisation

On stérilise dans trois situations :

Si le graphe contient un circuit positif : ce cas représente une solution non réalisable d'où il est inutile d'aller plus loin dans cette branche de l'arbre.

Le deuxième cas c'est la situation où l'évaluation du nœud représentant le sous problème donne une valeur de C_{max} supérieure ou égale à celle de la meilleure solution réalisable trouvée jusqu'à présent, qui est pour nous la borne sup.

Le troisième cas c'est la situation où la borne inférieure du sommet en cours dépasse la valeur de la borne sup.

5.4 Borne inférieure

Nous décrivons dans ce qui suit la borne inf. utilisée dans le cas d'un job shop classique, cette borne est basée sur la résolution des sous problèmes à une machine étudiée par Carlier.

a/ Le problème à une machine

Le problème à une machine est associé au problème job shop en choisissant une machine et relaxant les contraintes disjonctives concernant les autres machines.

Soit k un sous ensemble de l'ensemble de toutes les opérations O, k contient les opérations qui appartiennent à la même machine non relaxée.

Soient :

- r_i : La date de début au plutôt de l'opération i représentée par l(s,i) qui est la valeur du plus long chemin entre s et i.
- q_i : La valeur du plus long chemin de i vers T (le sommet terminal du graphe alternatif) noté aussi l(i,T).

$$H(k)= Min\{r_i / i \in k\} + \sum\{p_i / i \in k\} + Min\{q_i / i \in k\} \qquad 1$$

Proposition1 [1] : H(k) est une borne inférieure pour les makespan du problème de one-machine.

Proposition2 [1]: Soit V_k la valeur optimale pour le problème one-machine associé à la machine k.

LB= $Max\{V_k / k = 1, m\}$ est la borne inférieure pour le makespan du problème job shop.

Il est prouvé que LB= $Max\{H(k) / k \subseteq O\}$ peut être calculée en O(n log n).

Remarque : Nous avons constaté après application que dans le cas d'un job shop avec blocage cette borne inf. n'est vraiment pas très efficace. Ceci nous pousse à voir une autre borne plus efficace, en d'autre termes, la résolution des sous problèmes à une machine ne donne pas de bon résultats avec la contrainte de blocage car chaque opération ne dépend pas seulement de la machine actuelle mais dépend aussi de l'état de la machine suivante.

6. Discussion des résultats

Les résultats sont présentés dans un tableau de 6 colonnes :

La première colonne (nb_travaux x nb_machines) est la colonne de la taille du problème, ou nb_travaux est le nombre de travaux alors que nb_machines est le nombre des machines, nous rappelons que la complexité d'un problème de job shop de taille (nombre travaux x nombre machines) au pire des cas est donnée par la formule: (nombre travaux !)$^{\text{Nombre machines}}$ et elle représente la taille de l'espace de recherche (nombre de combinaisons possibles) ça en assurant que chaque ligne de données est complète i.e. que chaque travail doit demander toutes les machines.

Dans La deuxième colonne on a mis la valeur de C_{max} trouvée par le recuit simulé qu'on a développez et qui sert comme une borne supérieure d'entrée. La troisième colonne c'est le C_{max} trouvé par notre méthode de séparation et évaluation. Le temps de réponse se trouve dans la quatrième colonne, dans la 5eme colonne nous avons mis (Nb_noeuds) qui est le nombre des nœuds stérilisés puisque la fonction d'évaluation donne une valeur plus grande que la bonne solution trouvée jusqu'à présent. Le

nombre de nœuds non réalisables dans la colonne 6 c'est le nombre de nœuds stérilisés car ils portent un circuit positif, notant que la détection est immédiate à l'apparition du circuit grâce à l'algorithme de détection des composantes fortement connexes.

Nb_trav. x nb_mach.	Résultat recuit simulé (UB)	C_{max} Bellman	Cmax FORD	Temps (seconde) Bellman	Temps (seconde) FORD	Nb_noeuds LB>=UB Bellman	Nb_noeuds non réalisables Bellman
4x5	395	395	395	$188x10^{-3}$	0.031	81	105
4x10	633	633	633	4,875	0.516	332	321
4x15	957	955	955	25,578	0.531	259	429
5x10	778	681	681	77,250	7.28	1695	3031
5x15	994	-	923	-	2329,36	-	-
6x5	435	427	427	168,8	21.18	40951	119193
6x10	765	758	758	34920	1715.70	264224	343350
6x15	1247	1229	1229	26289.5	45020.96	8904	48667
7x5	679	638	638	2181	424.07	113484	416412
8x5	677	659	659	1778,4	12059.7	3347340	7516371
9x5	722	690	690	13932	2580	397858	1679816

Tableau 1. Résultats SEP avec Bellman et Ford.

Nous constatons d'après le tableau ci-dessus que l'évaluation de Ford prime sur l'évaluation de Belmann dans la majorité des exemples sauf à l'exemple 6x15 où Belmann prime sur Ford (en terme de temps d'exécution).

7. Conclusion

Dans ce papier nous avons présentez une méthode par séparation et évaluation permettant de résoudre le problème d'ordonnancement avec blocage, cette méthode utilise deux manières différentes d'évaluation, la première est basée sur l'algorithme de recherche de plus court chemin de Bellman utilisé avec les algorithmes de mise à niveau et des composantes fortement connexes, la deuxième utilise directement l'algorithme de recherche de plus long chemin dans un graphe de Ford. Une comparaison des deux méthodes est présentée, dans laquelle nous avons constaté que pour certains exemples Bellman prime pour d'autre c'est Ford qui prime. Une amélioration de notre travail peut être prise en compte en améliorant les bornes inf. et sup. du problème, ceci allègera la méthode pour des exemples un peu plus grands. Il serait aussi intéressant de voir l'effet de la parallélisation de cette SEP sur les tailles des problèmes à résoudre.

8. Références

1. A. Mascis and D. Pacciarelli. Job shop scheduling with blocking and no-wait constraints. European Journal of Operational Research 143, 498–517 (2001)

2. A. Soukhal, A. Oulamara and P. Martineau. Complexity of flow shop scheduling problems with transportation constraints. European Journal of Operational Research, 161, 32-41 (2005).

3. X. Wang and L. Tang. A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers. Computers & Operations Research, (In press).

4. S. Martinez, S. Dauzère-Pérès, C. Guéret, Y. Mati, and N. Sauer. Complexity of flowshop scheduling problems with a new blocking constraint. European Journal of Operational Research, 169, 855-864 (2006).

5. A. Soukhal and P. Martineau. Resolution of a scheduling problem in a flowshop robotic cell. European Journal of Operational Research, 161, 62-72 (2005).

6. D.P. Ronconi. A note on constructive heuristics for the flowshop problem with blocking. International Journal of Production Economics (2004).

7. L. Wang, L. Zhang and D. Zheng. An effective hybrid genetic algorithm for flow shop scheduling with limited buffers. Computers & Operations Research, 33, 2960-2971 (2006).

8. J. Grabowski and J. Pempera. The permutation flow shop problem with blocking: A tabu search approach. Omega, 35, 302-311 (2007).

9. D. P. Ronconi and L.R.S. Henriques. Some heuristic algorithms for total tardiness minimization in a flowshop with blocking. Omega, (In press).

10. H.W. Thornton and J.L. Hunsucker. A new heuristic for minimal makespan in flow shops with multiple processors and no intermediate storage. European Journal of Operational Research, 152, 96-114 (2004)

11. V. Caraffa, S. Ianes, T.P. Bagchi and C. Sriskandarajah. Minimizing makespan in a blocking flowshop using genetic algorithms. International Journal of Production Economics, 70, 101-115 (2001).

12. Y. Mati, N. Rezg and X. Xie. Scheduling Problem of Job-Shop with Blocking: A Taboo Search Approach MIC'2001 - 4th Metaheuristics International Conference, (2001).

13. H. Gröflin and A. Klinkert. A Tabu Search for the Generalized Blocking Job Shop MIC2005: The Sixth Metaheuristics International Conference, (2005).

14. R.Graham, Lawler E., Lenstra J.K., and Rinnooy Kan A., (1979).Optimization and approximation in deterministic sequenecing and scheduling theory: a survey. Annals of discrete mathematics, 5: pp.2 87-326.

Réseaux et applications réparties

Impact de la prise en compte des Contraintes Transactionnelles sur l'Orchestration des Services Web

Khebizi Ali[1], Hassina Seridi[2]
[1]Département d'Informatique, Université 8 Mai 45, Guelma 24000, Algérie
kheali@hotmail.com
[2]Université Badji Mokhtar, BP 12 Annaba 23000, Algérie
LabGed Laboratoire de Gestion du Document
seridi@labged.net

Résumé: Les progrès réalisés par la technologie des services Web demeurent insuffisants pour la prise en charge des vrais processus métiers trans-organisationnels. En effet, dans les scénarios de composition des services, l'élaboration des schémas d'orchestration et la gestion de la compensation demeurent des processus complexes et exigent une description plus riche des interactions engagées entre les divers services participants.
Dans cet article, basé sur l'analyse d'un scénario réel d'une application e-gouvernement dénommée «*Retraite*», nous aborderons l'analyse de compatibilité et d'équivalence des protocoles de services enrichis par les contraintes transactionnelles, et ce lors de l'orchestration des servies Web et nous proposerons une approche pour la modélisation des protocoles de compensation associés aux transactions inachevées.

Mots Clés: Protocoles de services, Contraintes Transactionnelles, Orchestration de Services, Compatibilité de services, Equivalence de services, Compensation.

1. Introduction

Les services Web offrent de fortes potentialités pour l'intégration des applications intra et interentreprises issues d'environnement hétérogènes et distribués sur le Web. Cependant, la première génération de services Web basée sur l'infrastructure: WSDL [1], SOAP [2] et UDDI [3], n'a permet que de surmonter le problème d'interopérabilité entre plateformes hétérogènes et reste, alors, insuffisante pour l'automatisation complète des vrais processus métiers trans-organisationnels. En effet, cette infrastructure de base est inadéquate pour la prise en charge de la composition des services afin de créer de nouvelles applications à forte valeur ajoutée sur la base de celles déjà existantes. La seconde génération de standards basée sur WS-Coordination [4], WS-Transaction [5] et BPEL [6] renforce l'infrastructure existante par les outils et mécanismes appropriés permettant la description des processus métiers, la composition des services et la gestion des interactions. Néanmoins, la nature intrinsèque des transactions métiers qui sont de longue durée et consommatrices de ressources exige des techniques de compensation dépassant les classiques propriétés ACID (Atomicité, Cohérence, Isolation, Durabilité). En effet, dans les environnements services Web, la compensation d'un protocole inachevé est assurée par le middleware d'une manière transparente en exécutant un autre protocole de compensation prédéfini par le développeur du service [7].

Or, la description actuelle du comportement externe des services, par leur interface et par leur protocole de conversation, n'exprime pas d'une manière consistante les caractéristiques inhérentes à la sémantique réelle des interactions engagées lors de la composition des services, bien que les contraintes d'ordre sur les opérations [8] et les contraintes temporelles [9] ont été prises en compte et injectées dans le modèle des protocoles de services modélisés par les automates d'états finis déterministes. En effet, les contraintes transactionnelles et les effets qu'elles induisent n'ont pas bénéficié de tout l'intérêt qu'elles revêtent.

Dans un tel contexte, le rehaussement de la description des protocoles de services par la prise en compte des contraintes transactionnelles impactera, promptement, l'analyse de compatibilité et d'équivalence des protocoles de services telles qu'elles ont été formalisées dans [8].

Nous proposons dans cet article un réexamen de l'analyse de compatibilité et d'équivalence des protocoles de services enrichis par les contraintes transactionnelles [10], et ce lors de l'orchestration des services Web. Nous exposerons, par la suite, une formalisation du processus de compensation des opérations et/ou services avortés et nous proposerons une modélisation des protocoles de compensation.

La suite de l'article sera structurée comme suit : dans la section 2, nous motivons notre travail, par l'exposé d'un scénario réel illustrant la substitution, la compatibilité et la compensation des services engagés lors de l'orchestration des services. L'analyse de compatibilité des protocoles de services, enrichis par les contraintes transactionnelles est présentée en section 3, et leur analyse d'équivalence est abordée au niveau de la section 4. La section 5, sera consacrée à la modélisation du processus de compensation. En fin, la section 6 contiendra nos conclusions et nos travaux futurs.

2. Intérêt et impact de la prise en compte des contraintes transactionnelles lors de l'orchestration des services Web

Avec la prolifération des services Web, leur composition est devenue un enjeu majeur pour les développeurs. Cette technique d'implémentation de nouveaux services, par articulation de ceux déjà existants, répond parfaitement aux objectifs de réutilisabilité des composants logiciels. Cependant, les problèmes de compatibilité des services découverts et susceptibles d'être intégrés dans la composition, les difficultés liées à la recherche de services équivalents afin de substituer ceux qui sont défaillants ou devenus incompatibles suite à leur évolution, ainsi que la complexité de la gestion des transactions et de leur compensation demeurent résiduels. Ces constats sont dus, principalement, à des déficits conceptuels au niveau de la description des protocoles des services eux-mêmes, et plus particulièrement, à l'absence de la prise en compte des contraintes transactionnelles dans la modélisation des comportements externes des services.

Afin de mettre en évidence les insuffisances précitées, nous exposons dans ce qui suit un scénario réel d'un service « *Retraite* » relatif à une application e-gouvernement du domaine de la sécurité sociale qui permet de prendre en charge les demandes de retraite des citoyens demandeurs.

Pour l'élaboration du service Web *Retraite*, permettant de liquider les droits en matière de retraite d'un citoyen, le développeur compose les services *Pension* (à son

tour composé des services : *Carrière* et *Etat- civil*), le service *Déclaration-Salaires* et le service *Assurance sociale*, comme illustré dans la **Fig.1.**
L'objectif du service *Retraite* est de permettre au client qui l'invoque, de prendre connaissance de ses droits de retraite, d'être notifié en cas de validation de sa demande et de bénéficier des prestations sociales.

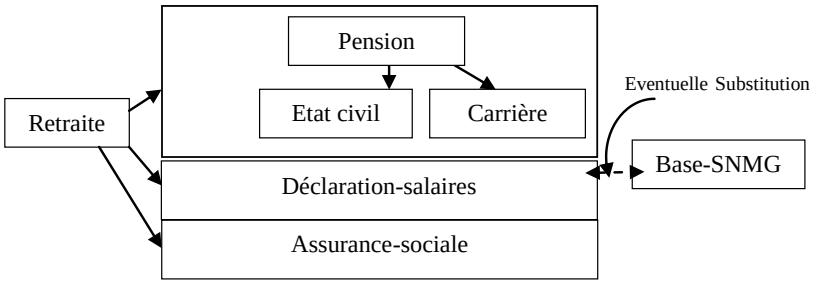


Fig. 1 : Composition du service *Retraite* par articulation d'autres services

Description des fonctionnalités des différents services à composer:

- Service ***Pension***: Permet de vérifier et de calculer les droits en matière de retraite sur la base des données état-civil et des données carrière.
- Service ***Carrière***: Fournit les données carrière (période d'activité, date entrée,…)
- Service ***Etat civil***: Fournit les données sur la situation matrimoniale et la liste des ayant-droits. Ces éléments entrent en compte dans le calcul des droits de retraite.
- Service ***Déclaration-salaires***: Fournit les salaires de référence servant au calcul de la retraite (moyenne des salaires des cinq meilleures années d'activité).
- Service ***Assurance-sociale***: le futur retraité fera l'objet d'une déclaration au niveau de l'organisme de sécurité sociale (service **Assurance-sociale)** pour pouvoir bénéficier des prestations sociales.
- Service **Base-SNMG** : En cas de défaillance du service ***Déclaration-salaires***, ce service fournira le salaire de référence (Salaire National Minimum Garanti).

Les différents services sont exposés par divers organismes (fournisseurs): la caisse des retraites pour les services *Pension, carrière* et *base-SNMG*, la caisse d'assurance sociale pour *Assurance-sociale*, la mairie pour le service *Etat-civil* et le service *Déclaration-salaires* est exposé par les divers employeurs.
D'un point de vue implémentation, le service *Retraite* est composé car il invoque d'autres services et l'orchestration traite, justement, de la façon dont ces différents services sont composés en un tout cohérent. Elle spécifie l'ordre dans lequel les services sont invoqués et les conditions sous lesquelles un service peut ou ne peut pas être invoqué. [7]. C'est un processus coopératif impliquant différents partenaires qui vise à intégrer de façon cohérente la collaboration des diverses activités des services participants. La notion de *protocole de service* [8] permet de représenter le processus métier composite «*Retraite*».

Nous détaillons, dans la **Fig.2**, le protocole du service composé *Retraite*. Le formalisme des automates d'états finis déterministes est utilisé comme outils de représentation. Les messages expriment les opérations permettant la transition entres les différents états de l'automate.

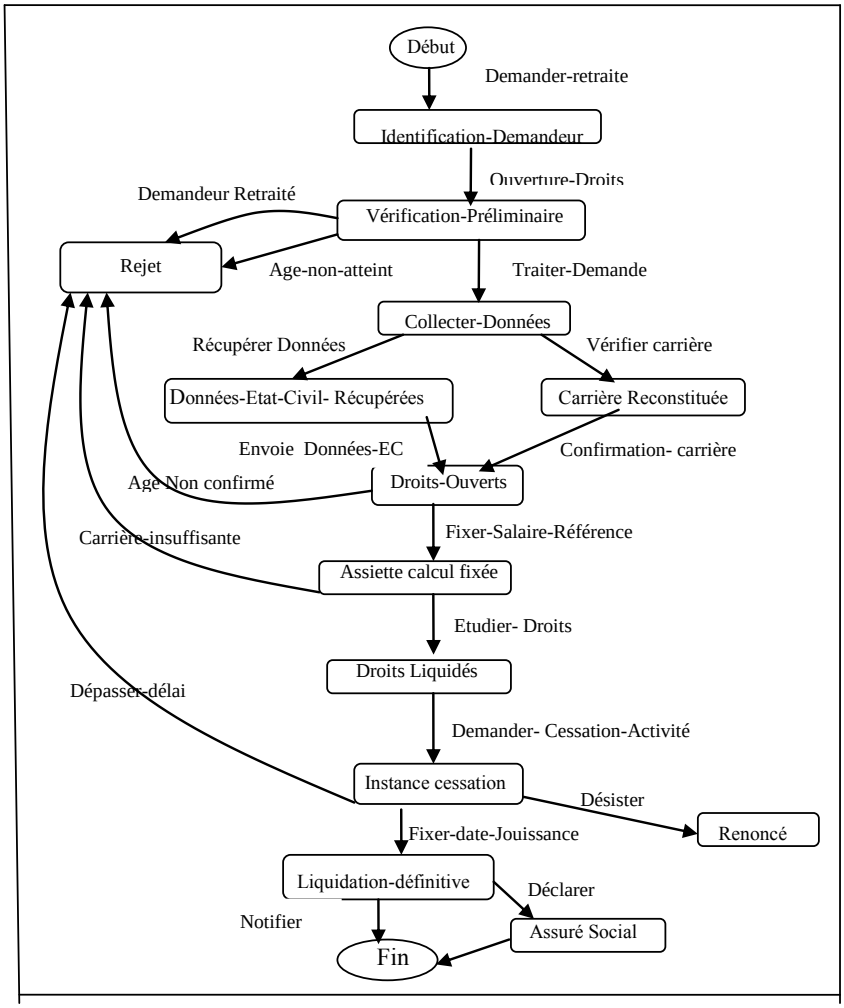


Fig. 2 : Protocole du service composé *Retraite*

Du point de vue interaction, la composition exige les conditions suivantes:

(i) Le protocole du service composite « *Retraite* » doit être compatible avec ceux des fournisseurs (*Déclarations-salaires*, *Assurance-sociale*) pour garantir que des conversations correctes puissent être engagées. Cette compatibilité doit être étendue aux effets engendrés par les différents messages.

(ii) En cas de défaillance d'un service (Déclaration-salaires par exemple), le service composite doit localiser un autre service fonctionnellement équivalent pour le substituer. La notion d'équivalence doit maintenir la cohérence de l'orchestration tout en tenant compte des contraintes transactionnelles. Le service Base-SNMG ne pourra substituer le service Déclaration-salaires sans que leurs effets ne soient équivalents.

(iii) Si le demandeur ne fournit pas une cessation d'activité dans un délai déterminé (06 mois), alors tout le processus est annulé, entrainant le déclenchement d'un nouveau protocole de compensation pour annuler sémantiquement les effets engendrés. L'élaboration du protocole de compensation doit, impérativement, tenir compte des effets transactionnels afin de pouvoir les défaire.

En dépit des avancés réalisées dans le cadre de la gestion des transactions au niveau middleware et ayant abouti aux consensus autour des spécifications largement adoptées (WS-Transaction [5], WS-Transaction Atomic [12] et WS-Business Activity [13]), le domaine de la gestion des transactions au niveau protocole reste faiblement exploré engendrant des difficultés d'analyse et de programmation lors de la composition des services Web. Par la prise en compte et la modélisation des effets transactionnels les notions de compatibilité, d'équivalence et de compensation prennent une dimension plus riche, car basées sur la sémantique réelle des interactions vue sous l'angle des effets engendrés par les transactions déclenchées.

Pour rehausser la description des protocoles de services à sa sémantique interactionnelle en prenant en considération les effets des transactions, nous avons proposé dans nos travaux précédents [10] un modèle formel de représentation des effets basé sur leur perception en tant que requêtes de mise à jour de la base de données de type: *Insertion, Modification et Suppression.* Les requêtes de consultation n'affecteront, en aucun cas, l'état du client. Conformément à notre modèle, la structure des messages échangés lors des interactions est de type: ***m (p, e, e')***, où :

m: le message et *p* sa polarité (+,-). Une polarité (+) indique que le message est en entrée (consommée) et une polarité (-) signifie qu'il est en sortie (produit) [8].

e: Ensemble des effets observés du coté du client, représenté par une requête Q de mise à jour de la base de données.

e': Ensemble des effets de compensation pour défaire les effets *e* représentés par une requête *Q'* de mise à jour pour compenser les effets *e*. On notera: *Q' compense Q*.

A noter que si les requêtes sont de type consultation qui n'affectent pas le client, le message sera noté : ***m (p, *,*)***

Par ailleurs, le nouveau modèle de protocole de service enrichi par les contraintes transactionnelles (***Protocole à effets transactionnels***) est représenté par l'automate d'état fini déterministe, décrit par le tuple $P = (S, s_0, F, M, R, S_B)$ [10].

Néanmoins, cet enrichissement des protocoles de services par la prise en compte des propriétés transactionnelles, impactera directement le processus d'orchestration des services et exige, par conséquent, un réexamen de leur gestion lors de l'orchestration. Nous aborderons, dans la suite de l'article, l'analyse de compatibilité et d'équivalence des protocoles à effets transactionnels et nous formaliserons le processus de compensation.

3. Analyse de Compatibilité des Protocoles à Effets Transactionnels lors de l'Orchestration des Services

L'analyse de compatibilité de deux protocoles permet de tester s'ils peuvent interagir correctement en engageant des conversations correctes entre eux [8]. Pour les protocoles de services à effets transactionnels, cette analyse consiste à vérifier si les deux protocoles en interaction engendrent les mêmes types d'effets.

Conformément au modèle d'effets proposé [10], les effets induits par les messages d'un service Web sont des actions affectant le client qui l'invoque et qui sont matérialisées par la modification de son état. La conséquence directe de cet apport est la révision de la notion de compatibilité des services.

Dans cette optique, les protocoles du service client et celui du fournisseur doivent induire des effets qui seront compatibles. Par *compatibilité des effets*, nous sous-entendons que les requêtes de mise à jour au niveau des bases de données sont de même type. Cette condition conduit à une redéfinition du concept de chemin d'interaction, tel qu'il a été spécifié dans [8] pour l'étendre au type de requête.

- **Chemin d'interaction étendu:** La définition du chemin d'interaction dans [8] est limitée aux messages et aux états. Elle est décrite conformément à l'expression générique: ***((Etat1.Etat2).Message)****

Pour prendre en compte les effets transactionnels, nous proposons une extension de cette spécification au type de requêtes de mise à jour, comme suit:

((Etat1.Etat2). Message. Type Requête)*

Cette extension permettra de garantir, lors de l'analyse, la vérification de la compatibilité des requêtes de mise à jour. Elle favorise une spécification plus riche de l'interaction entre les protocoles. Ainsi, deux protocoles de services ne pourront être compatibles sans que les requêtes associées aux messages ne le soient. Le besoin d'une nouvelle spécification de la compatibilité nous conduit à une redéfinition des concepts: *Effets compatibles* et *messages compatibles*.

- **Effets compatibles:** Deux effets induits par un ou plusieurs messages sont compatibles si les actions produites au niveau des bases de données sont identiques.

Le cas le plus évident de la compatibilité des effets est celui de l'analogie des types de requêtes des deux messages. Cependant, dans certaines situations, les effets sont traités par rapport à la séquence des requêtes. Un cas typique est le traitement d'une requête de modification. Elle est considérée de type Modifier dans un premier protocole, alors qu'au niveau du 2ième protocole, c'est une séquence d'une requête Insertion suivie d'une autre requête de Suppression. Dans ce cas, la compatibilité des effets est préservée, malgré l'incompatibilité des messages. Un deuxième cas concerne la décomposition d'une requête Modifier sur plusieurs attributs et/ou des sous ensembles d'attributs, où chaque modification est portée par un message à part. Pour illustrer ce deuxième type de compatibilité, nous reprenons le scénario de la Fig.2. La logique métier du service client peut traiter la cessation d'activité différemment de celle du fournisseur. Par exemple, au niveau du protocole du client de la Fig. 3, où les messages sont décrits par leurs effets et effets de compensation représentés par des requêtes (pour simplifier nous partons de l'état *Droits-Liquidés*). Les deux attributs : *Date-jouissance* et *Situation* de l'enregistrement concerné sont

modifiés en une seule passe en invoquant le message: *Notifier- Décision*, permettant la transition de l'état *Instance-cessation* vers l'état *Fin*. Dans ce cas, est-ce que des conversations correctes peuvent être engagées avec le service du fournisseur de la **Fig. 2** ? La réponse à cette question est positive, si les requêtes de mise jour sont de même type, même si la mise à jour est faite de manières différentes.

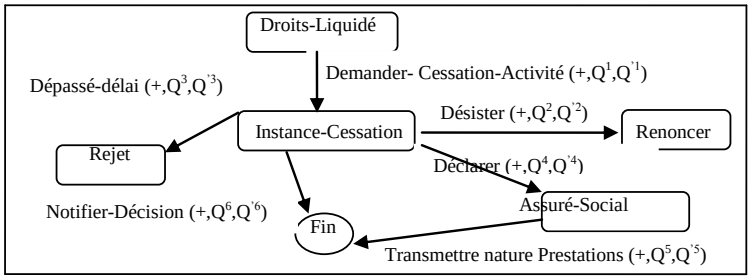


Fig. 3: Un protocole client qui fonctionne différemment. Pourra-t-il interagir correctement ?

- **Messages compatibles:** Deux messages appartenant à deux protocoles différents sont compatibles si les requêtes de mise à jour associées sont de même type.

A titre d'exemple : le message *Traiter-Demande* du protocole fournisseur du service *Retraite* **de la Fig. 2**, dont la requête de mise à jour est de type *Modification* ne pourra être invoqué correctement par le message : *Demander-retraite* d'un autre protocole client sauf si la requête associée à ce dernier est aussi de type *Modification.* Le cas contraire, il est évident que les effets observés seront non conformes et par conséquent, la conversation ne sera pas possible.

- **Définition formelle de la compatibilité des protocoles à effets transactionnels:**

Soient P_1 et P_2 deux protocoles de services à effets transactionnels modélisés avec des automates d'états finis, comme suit :

$P_1=(S^1,s^1_0,F^1,M^1,R^1,S^1_B)$ *et* $P_2=(S^2\ s^2_0, F^2, M^2, R^2, S^2_B)$ [10]

P_1 *et* P_2 sont compatibles s'il existe une conversation entre P_1 *et* P_2 (au moins un chemin d'exécution complet de P_1 est supporté par P_2) dans laquelle les effets de P_1 sont compatibles avec ceux de P_2.

Soit m_i le message à invoquer pour la transition courante, avec: $m_i(p^1,Q_1,Q'_1) \in P_1$ et $m'_i(p^2, Q_2, Q'_2) \in P_2$. La compatibilité des messages m_i et m'_i est vérifiée si :

- -Les polarités p^1 *et* p^2 se compensent (+, -) ;
- -En partant des mêmes états sources, les états cibles sont identiques, ou transitivement identiques; (transition par des états intermédiaires).
- -Les requêtes de mise à jour Q_1 et Q_2 sont de même type ou transitivement de même type (l'exécution d'une séquence de requêtes est de même type).
- -Les requêtes de compensation Q'_1 et Q'_2 sont de même type ou transitivement de même type (l'exécution d'une séquence de requêtes de compensation est de même type).

Plus concrètement et en se basant sur le concept de *la compatibilité des effets*, deux situations peuvent se présenter :

(i) Même type des requêtes de mise à jour: C'est le cas le plus simple. Il correspond à une compatibilité évidente des messages. Par exemple les messages *Etudier-droits* du service *Retraite* de la **Fig. 2**, permettant la transition de l'état *Assiette-calcul-fixée* vers l'état *Droits-liquidés* à pour effets de modifier l'enregistrement du demandeur en actualisant les attributs (tels que : salaire-mensuel, taux-retraite). Si le message du protocole du client est aussi de type *modification*, alors la compatibilité est garantie. Le cas contraire, l'interaction ne sera pas permise, même si elle est vérifiée du point du vue ordre et polarité des messages.

(ii) Cas de compatibilité des types de séquences de requêtes: Induisant les mêmes types d'actions au niveau des bases de données. C'est le cas le plus général et correspondant à des situations de transitivité. Dans ce cas les logiques métiers du service client et celui du fournisseur peuvent être différentes, en termes d'ordre des opérations ou des états parcourus, mais elles demeurent canalisées par des garde-fous qui sont les types de requêtes, garantissant une conversation correcte.

A titre d'illustration, considérant un client dont la logique métier pour le traitement des demandes de retraite est schématisée par le protocole de service à effets transactionnels de la **Fig. 3.** Si les requêtes des messages: *Demander-Cessation-Activité*, *Désister* et *Délai-dépassé* sont du même type que ceux correspondantes dans le protocole *Retraite* de la **Fig. 2**, la compatibilité partielle des chemins aboutissant aux états finaux *Renoncer* et *Rejet* est vérifiée. Par contre, pour les chemins d'exécution complets: *Droits-Liquidés.Instance-Cessation.Assuré-Social.Fin* et *Droits-Liquidés.Instance-Cessation.Fin, elle* n'est pas évidente. En prenant en compte les effets des opérations, l'interaction ne sera correcte que si les effets soient compatibles. Comme, la transition de l'état : *Instance-cessation* vers l'état *Fin* est déclenchée par le message unique *Notifier-Décision*$(+,Q_6,Q'_6)$ dans le protocole du service client, alors qu'elle est réalisée par deux messages distincts : *Fixer-date-jouissance*$(-,Q_7,Q'_7)$ et *Notifier-Retraite* $(-,Q_8,Q'_8)$ dans le protocole du service fournisseur de la **Fig.2**, la compatibilité est préservée seulement si: Q_6 est de même type que la séquence $Q_7 o\, Q_8$, où *o* désigne d'ordonnancement des requêtes.

4. Analyse d'Equivalence des Protocoles à Effets Transactionnels lors de l'Orchestration des Services

L'analyse d'équivalence a pour objectifs de s'assurer que deux protocoles de services peuvent être utilisés interchangeablement dans n'importe quel contexte d'une manière transparente à l'utilisateur [8]. Elle est utile dans le contexte de la substitution d'un protocole défaillant (panne de service, évolution de version, …) par un autre qui offrira, au moins, les mêmes fonctions. Dès lors, son intérêt est grandissant lors de l'orchestration des services. Cependant, cette analyse exige une révision et un raffinement spécifique pour les protocoles à effets transactionnels, du fait qu'elle sera, plutôt, basée sur la sémantique des transactions vue sous l'angle des effets produits.

Pour illustrer cet impact, les deux protocoles des services *Déclaration-Salaires* et *Base-SNMG*, intégrés dans la composition du service *Retraite* et schématisés par la **Fig. 4**, ne seront équivalents que si les effets produits, le sont. Autrement, si les requêtes de mise à jour associées aux messages sont équivalentes.

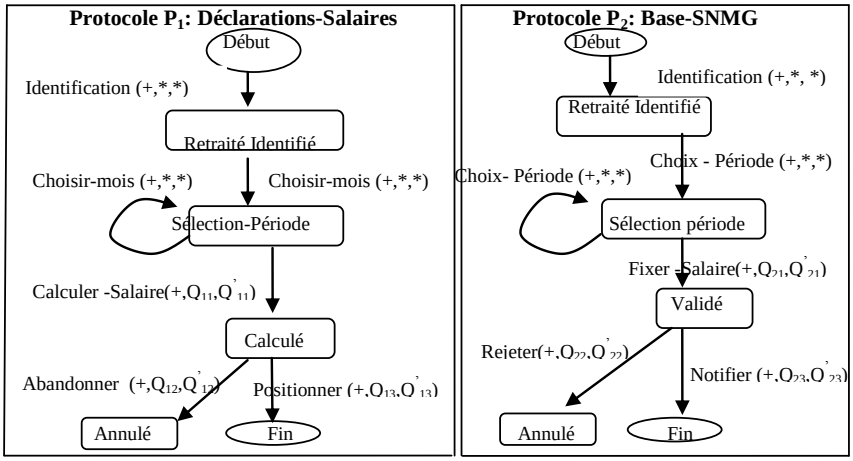


Fig. 4: Deux protocoles: Equivalence structurelle vs Equivalence à base d'effets

Les messages de la forme ***m(p,*,*)*** correspondent à de simple consultations de la base et n'induisent aucun effet. Pour que les protocoles P_1 et P_2 soient équivalents, il faut que les équations (1) et (2) soient vérifiées: (le symbole ≡ exprime l'équivalence des requêtes) de mise à jour au niveau de la base de données.

$$\begin{cases} Q_{11} \equiv Q_{21} \text{ et } Q_{12} \equiv Q_{22} \text{ et } Q_{13} \equiv Q_{23} & (1) \\ Q'_{11} \equiv Q'_{21} \text{ et } Q'_{12} \equiv Q'_{22} \text{ et } Q'_{13} \equiv Q'_{23} & (2) \end{cases}$$

La réalisation des équations (1) et (2), simultanément, explique une façon de faire, de la part des fournisseurs de services, parfaitement identique; chose qui n'est pas évidente dans la réalité, du fait que chaque fournisseur a sa propre logique métier qui lui est spécifique. Effectivement, certaines opérations peuvent être regroupées, d'autres sont éclatées ou ordonnées différemment. Par ailleurs, les logiques de compensation diffèrent, éventuellement, d'un fournisseur à l'autre, engendrant la non vérification de l'équation (2), même si (1) est vérifiée. Ces constats, nous conduisent à raisonner en termes d'effet globale induit par la séquence de requêtes pour chaque chemin d'exécution complet (qui commence d'un état initial et se termine par un état final [8]).

L'équivalence des effets globaux est formalisée par les équations (3) et (4).

$$\begin{cases} Q_{11} \circ Q_{12} \equiv Q_{21} \circ Q_{22} \text{ et } Q_{11} \circ Q_{13} \equiv Q_{21} \circ Q_{23} & (3) \\ Q'_{12} \circ Q'_{11} \equiv Q'_{22} \circ Q'_{21} \text{ et } Q'_{13} \circ Q'_{11} \equiv Q'_{23} \circ Q'_{21} & (4) \end{cases}$$

L'examen approfondi de la vérification des équations (1), (2), (3) et (4) conduit à plusieurs situations possibles. Chacune de ces situations détermine une ***classe d'équivalence*** des protocoles à effets transactionnels.

1. L'équation (1) seule est vérifiée : Equivalence Stricte des effets directs.
2. L'équation (2) seule est vérifiée: Equivalence Strict des effets de compensation.
3. Les équations (1) et (2) sont vérifiées: Equivalence Stricte Parfaite.
4. L'équation (3) seule est vérifiée: Equivalence convergente à effets directs des états des bases de données.
5. L'équation (4) seule est vérifiée: Equivalence convergente à effets de compensation des états des bases de données.

6. Les équations (3) et (4) sont vérifiées : Equivalence Parfaite à convergence des états des bases de données.

La dernière classe d'équivalence revêt un intérêt particulier. Elle reflète une manière de faire différente, de la part des fournisseurs de services, qui est matérialisée au niveau de la base de données par des requêtes différentes, soit dans leur type, soit dans leur ordre, tout en convergeant vers des bases de données équivalentes. Il s'ensuit que l'équivalence des protocoles à effets transactionnels peut être vérifiée sans que l'équivalence structurelle le soit. Par ailleurs, nous déduisons que l'équivalence stricte n'est qu'un cas particulier de l'équivalence à convergence des états des bases de données.

5. Processus de compensation des services lors de l'Orchestration

Le mécanisme de compensation offre un cadre conceptuel adéquat pour défaire sémantiquement les effets engendrés par les transactions inachevées durant l'orchestration d'un service Web. Mais, ce processus en lui-même, demeure fastidieux car basé sur la simple manipulation des données et aucune garantie n'est offerte pour vérifier sa **consistance** avec le protocole déclencheur. En se basant le modèle de représentation des effets intégrant, conjointement, pour chaque message les effets induits et leurs effets de compensation modélisés par des requêtes sous la forme ***m (p, Q, Q')***, il apparait que la problématique de la compensation des services lors de l'orchestration est réduite à celle de la manipulation des requêtes de mise à jour de la base de données. En effet, pour défaire les effets induits par une requête Q d'un message m, il suffira d'exécuter la requête inverse Q'. Cette simplicité, offerte par la richesse du modèle de représentation, permet d'identifier clairement les requêtes nécessaires à la modélisation du protocole de compensation et de les structurer dans l'ordre exprimant la logique de compensation afin de construire le protocole de compensation.

Définition formelle de la compensation: Une requête Q_j décrivant les effets observés du côté du client et induits par une opération d'un service Web S, compense les effets d'une autre requête Q_i, lors de l'exécution du service Web S, et on notera : $[Q_j\ comp(Q_i)]_{S,}$ si l'exécution de la séquence de requêtes : $Q_i\ o\ Q_{j,}$ n'engendre aucun effet du côté du client, sauf ceux, volontairement, désirés par le fournisseur de S et exprimant les charges des transactions imposées aux clients, suite à l'annulation de l'opération. On parlera de *couple effet-compensation*.

Plus formellement, soit un message ***m (p, e, e')*** tel que:

e : Les effets du message m induits par la requête Q_i.

e': Les effets du message du protocole compensatoire représenté par la requête Q_j.

Les effets de compensation peuvent être décomposés en deux types de sous-effets:

e'_a: Les effets d'annulation des effets du message e, représenté par la requête Q_a.

e'_f : Les effets volontairement désirés par le fournisseur du service et qui sont associés aux charges affectés au client suite à l'annulation de la transaction. Ils sont exprimés par la requête Q_f.

La décomposition des effets associés aux requêtes donnera: $e' \equiv e'_a + e'_f$;

$Q_j = Q_a\ o\ Q_f$; d'où : $[(Q_a\ o\ Q_f)\ comp(Q_i)]_{S;}$ exprimant le fait que la séquence de requêtes $Q_a\ o\ Q_f$ compense la requête Q_i dans le protocole de service S.

Il est claire que si : $Q_f \equiv \emptyset$ (pas de charges : *Effectless transactions* dans [11]), alors, l'annulation de la transaction n'affecte pas l'état de la base et on aura: $[Q_a\ comp(Q_i)]_S$.
Cette approche pour aborder le processus de compensation, permet de doter les développeurs de services des outils adéquats pour la modélisation et la vérification des protocoles de compensation, tout en garantissant que la compensation, assure effectivement, l'annulation sémantique du protocole déclencheur. Cette façon de faire est très rentable en termes de productivité du fait qu'elle évite de procéder par des suites et scénarios de tests qui sont non exhaustifs.
Pour illustrer cet apport, nous exposons dans la **Fig. 5**, le protocole de compensation du protocole P_1 : *Déclaration-Salaires* de la **Fig. 4**.
Ce protocole est invoqué implicitement par le fournisseur de service dans le cas ou le client invoque l'opération *Abandonner* permettant la transition de l'état *Calculé* vers l'état *Annulé* du protocole P_1 de la **Fig. 4**.
La logique métier du fournisseur consiste à annuler les modifications réalisées par les requêtes : Q_{11} et Q_{12}, en exécutant séquentiellement les requêtes Q'_{11} et Q'_{12}. A noter que ces deux dernières requêtes ont des effets permanents et ne peuvent être compensées (*Definite transitions* dans [11]).

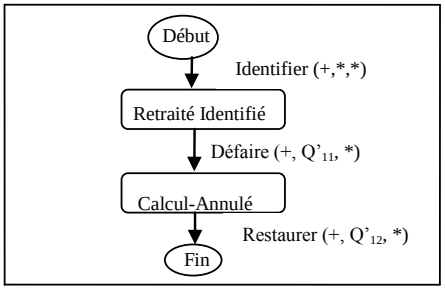


Fig. 5: Protocole de compensation du protocole P_1: *Déclaration-Salaires*

Dans la pratique, il est opportun de décomposer les requêtes complexes en un ensemble de requêtes basiques, dont la séquence d'exécution produira les mêmes effets au niveau de la base, permettant, ainsi d'identifier les effets de bases et de faciliter par conséquent la spécification des requêtes de compensation associées. En plus, la recherche d'une requête globale, dont l'effet résultant reflète l'annulation d'une séquence de requêtes, permettra de compenser une séquence d'opérations en une seule passe. (situation où tout le service est abandonné en une seule opération: fermeture de la fenêtre active, par exemple).

6. Conclusions

La composition des services Web est une technique permettant la réutilisation des services déjà existants pour créer de nouveaux services à forte valeur ajoutée. Pour sa mise en œuvre opérationnelle, la connaissance de la structure et du comportement des services constituants est inévitable, afin de s'assurer de la compatibilité et de l'équivalence des services à composer. Dans cet article, nous avons mis en évidence l'intérêt de la prise en compte des contraintes transactionnelles inhérentes aux interactions engagées lors des scénarios de composition de type orchestration des

services. Nous avons abordé l'analyse de compatibilité et d'équivalence des protocoles de services à base d'effets transactionnels, tout en identifiant les classes d'équivalence. Une approche, basée sur le modèle de requêtes de mise à jour de la base de données, et permettant de modéliser les protocoles de compensation des opérations et/ou services avortés a été proposée.
Comme travaux futurs, nous envisageons la spécification des algorithmes relatifs aux différentes classes d'équivalence déjà identifiées ainsi que la proposition des opérateurs algébriques de manipulation des effets transactionnels, afin de consolider l'assise théorique existante.

Références

1. R. Chinnici et al. Web Services description Language (WSDL) version 2.0 June 2007. http://www.w3.org/TR/wsdl20/
2. M. Gudgin et al. SOAP version 1.2, July 2001. http://www.w3.org/TR/2001/WD-soap12-20010709/
3. T. Bellwood et al. UDDI Version 3.0.2 UDDI Spec Technical Committee Draft, 2004. http://uddi.org/pubs/uddi_v3.htm/
4. F. Cabrera et al. Web Service Coordination (WS-Coordination), August 2005. http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-tx/WS-Coordination.pdf
5. F. Cabrera et al. Web Service transaction (WS-Transaction), January 2004. http://dev2dev.bea.com/pub/a/2004/01/WS- Ttransaction.html/
6. Web Services Business Process Execution Language Version 2.0 OASIS Standard, 11 April 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html/
7. Gustavo Alonso et al: Web services concepts Architectures and applications, Edition Springer Verlag Berlin 2004.
8. B. Benatallah et al : Representing, Analysing and Managing web Service Protocols. Data Knowledge Ingineering. 58 (3): 327-357, 2006.
9. J. Ponge et al: Fine-Grained Compatibility and Replaceability Analysis of Timed Web Service Protocols. ER 2007: 599-614
10. Ali Khebizi: External Behavior Modeling Enrichment of Web Services by Transactional Constraints, ICSOC PhD Symposium, December 2008. http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-421/paper12.pdf/
11. B. Benatallah et al : Web Service Conversation Modeling A cornerstone for E-Business automation, IEEE Internet computing 8 (1) (2004) 46-545 WSC
12. F. Cabrera et al. Web Services AtomicTransaction (WS-AtomicTransaction) August 2005 http://specs.xmlsoap.org/ws/2004/10/wsat/wsat.pdf/

[13] F. Cabrera et al. Web Services Web Services Business Activity Framework (WS- BusinessActivity) August 2005, http://specs.xmlsoap.org/ws/2004/10/wsba/wsba.pdf

Tolérance aux pannes dans les grilles de calcul

Bakhta Meroufel[1], Ghalem Belalem[2] , Nadia Hadi[3]

[1]bmeroufel@yahoo.fr , [2]ghalem1dz@univ-oran.dz, [3]yhana9@yahoo.fr
Département d'Informatique
Faculté des Sciences
Université d'Oran (Es Sénia)

Résumé :

Les grilles de calculs sont des systèmes distribués dans lesquels les pannes existent et ne sont pas des événements rares mais naturels. Comme il s'agit de connecter beaucoup de sites, eux mêmes potentiellement équipés d'une importante quantité de ressources, la notion de pannes est indissociable des environnements de grilles.

Ce présent travail consiste à construire une topologie logique basée sur la clusterisation hiérarchique non couvrante. Sur cette topologie, nous proposons un mécanisme de tolérance aux pannes qui est divisé en deux phases : (i) La première est « Prédiction des pannes » : chaque père surveille ses fils, si un de ses fils susceptible d'être défaillant, le père doit réagir pour protéger les répliques de ce nœud, donc il doit prendre des décisions adéquates en respectant l'espace mémoire et la demande sur cette réplique, (ii) La deuxième phase est la « détection de panne »: chaque nœud envoie périodiquement un message de vie a son père, si ce message n'arrive pas après un certain temps donné, le père considère son fils comme un nœud défaillant et il déclenche l'étape d'auto stabilisation.

Mots clé : Grille de calcul et de données, réplication, arbre recouvrant, tolérance aux pannes, clusterisation (regroupement),

1 Introduction

La perte d'un nœud dans la grille peut avoir des incidences sur le fonctionnement du système de grille et causer des pertes de données (plus souvent des répliques d'un objet donné) qui existent sur ces nœuds.

Dans la littérature [2][10][13][14], il existe plusieurs manières d'envisager une panne dans un système réparti à grande échelle telles que les grilles de calcul (Figure 1), nous pouvons citer :

a) **La réplication (Masking)** : La tolérance aux pannes par duplication consiste en la création de copies multiples des composants sur des processeurs différents. Cette approche par duplication rend possible le traitement des pannes en les masquant.

b) **Forward recovery :** Un point de reprise est un ensemble d'éléments (l'état de la mémoire par exemple) permettant de relancer ce processus en cas de défaillance d'un nœud.
c) **Backward recovery (auto-stabilisation) :** Dijkstra [1] définit un système réparti comme étant *auto-stabilisant* si, indépendamment de son état initial, ce système retourne à un *état légitime* en un nombre fini d'étapes, c'est-à-dire un état à partir duquel le système fonctionne correctement jusqu'à ce qu'une nouvelle défaillance survienne.

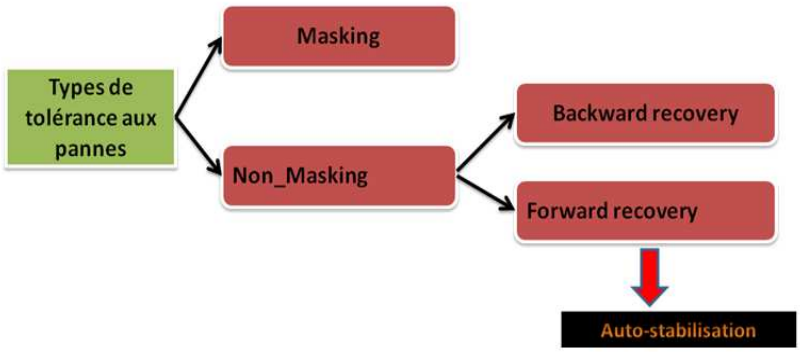


Fig.1. Les mécanismes de tolérance aux pannes

Notre proposition consiste à établir un mécanisme de prédiction des défaillances et de combiner deux techniques de tolérance aux pannes :

- La réplication pour tolérer la panne d'une donnée.
- L'auto stabilisation pour tolérer la topologie après pannes.

Le reste de l'article est organisé comme suit : La section 2 présente l'état de l'art sur les mécanismes de tolérance aux pannes dans les différents intergiciels et les plates formes des grilles de calcul. Dans la section 3, nous donnons une description du système sur lequel notre travail est basé. Nous réservons la section 4 à notre proposition de la tolérance aux pannes. Cette proposition est basée sur trois phases à savoir : la prédiction, la réplication et l'auto stabilisation du système. Nous terminons par une conclusion et des perspectives pour nos travaux futurs.

2 Etat se l'art

Condor [3] utilise les points de reprises pour tolérer les fautes mais il ne supporte pas la perte du coordinateur qui joue le rôle de service d'information et d'ordonnanceur. Il en est globalement de même pour toutes les approches centralisées.

Globus [4] peut survivre à des défaillances de nœuds qui toucheraient son système d'information (MDS) en utilisant le principe de réplication. LDAP est tout à fait adapté à cela.
De plus, si une partie de MDS, non répliquée, est perdue lors d'une défaillance, seulement le sous-arbre associé est perdu, ce qui n'est pas forcément catastrophique si la défaillance n'a pas lieu tout près de la racine de l'arbre. PUNCH [5] utilise aussi le principe de réplication pour assurer la continuité du service malgré les défaillances.
MPICH-V [6] est conçu pour tolérer la volatilité des nœuds mais il repose sur un ensemble de ressources stables pour le stockage des messages notamment.
Des approches complètement distribuées tolèrent beaucoup mieux la perte de nœuds puisque les services ne sont pas centralisés sur un seul nœud. C'est par exemple le cas de Vishwa [7] ou Zorilla [8] qui sont fondés sur des approches pair-à-pair. Toutefois, le système NodeWiz [9] fondé sur une approche pair-à-pair ne supporte pas les défaillances de nœuds.
Dans [15] nous trouvons une table de comparaison entre les différents intergiciels, et parmi les critères de comparaison, nous pouvons noter le mécanisme de suivi du système après pannes, l'existence des mécanismes de détection des pannes et les supports pour la tolérance.
Multi-agent DARX [10] utilise la réplication afin de fiabiliser des agents logiciels. Ce système propose d'adapter le mécanisme de réplication pour tolérer les fautes ainsi que le nombre de copies en fonction : 1) des agents répliqués, et 2) de l'estimation du niveau de risque de fautes.
Un compromis entre les systèmes de MVP et les systèmes pair-à-pair à grande échelle est proposé par la plate-forme JuxMem [11], un service de partage de données modifiables pour la grille. Il permet l'expérimentation des stratégies de tolérance aux fautes par l'utilisation des points de reprise et la réplication des fournisseurs.

3 Description de notre système

Le système proposé est bien adapter aux environnements à grande échelle où il n'existe pas une mémoire partagée. La communication est faite par le passage des messages. Chaque élément du système est identifié par un identificateur unique et des valeurs locales qui indiquent sont degré de vivacité.
Le nœud peut lire et modifier son propre état mais il n'accède à l'état de ces voisins que par lecture. Il peut stocker une ou plusieurs répliques de différents fichiers. La réplique est identifiée par un identificateur (ID) et un numéro de version.

3.1 Topologie logique

Il est difficile de gérer une topologie aléatoire avec des milliers des nœuds et arêtes. Pour cette raison nous trouvons la notion de la topologie logique qui permet de réorganiser les réseaux sous une forme favorable sans toucher cette topologie physiquement.

Dans notre travail, nous utilisons une clusterisation hiérarchique non couvrante (Figure 2). Cette topologie a plusieurs avantages parmi lesquels nous citons :

- Minimisation du temps de réception de chaque message (la gestion et la communication et faite au niveau de chaque cluster).
- Réduction du nombre de messages échangés grâce à l'architecture arbre.

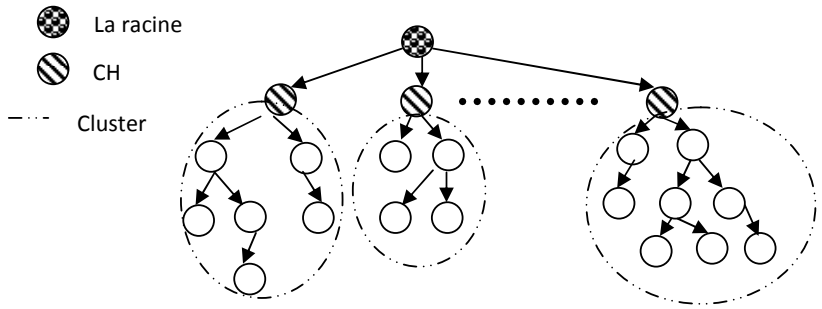


Fig.2. La topologie logique

La racine de la topologie contient la liste des répliques de chaque cluster ce qui permet le bon routage de la requête qui parvient d'un Cluster-Head (CH).
Le Cluster-Head : présente aussi la racine de l'arbre à l'intérieur du cluster et il permet de gérer les nœuds localement et de communiquer avec les autres clusters.
Le nœud représente un élément de stockage qui contient les données et il ne communique qu'avec les nœuds de même cluster.

4 Proposition d'un mécanisme de tolérance aux pannes

Nous décrivons les différentes techniques utilisées pour la tolérance aux pannes. Nous pouvons noter que chaque cluster peut déclencher et exécuter ces techniques de tolérance.

4.1 Prédiction de la panne

Chaque nœud possède des valeurs locales qui indiquent son degré de vivacité (énergie, durée de vie,..). Si cette dernière est inférieure à un certain seuil donné, le nœud informe son père qui doit réagir pour protéger les répliques de son fils:

- Si le père possède les répliques de son fils, il ne doit pas réagir.

- Sinon il envoie une demande à ses voisins (père et fils) pour stocker une ou plusieurs de ses répliques.
- Le nœud qui accepte la demande envoie un OK et la fréquence de demande sur cette réplique sinon il passe la demande à ses voisins.
- Quand le père reçoit ces OK il classe les fréquences de demande de la réplique par ordre croissant puis il envoie la réplique au nœud qui a la plus grande fréquence d'accès.
- Si la réplique est rare (un faible degré de réplication), le père peut envoyer cette réplique à plusieurs demandeurs.
- Les nœuds qui n'acceptent pas de stocker la donnée sont :
    - ❖ Les nœuds qui ont déjà cette réplique ou ils ont des voisins qui possèdent cette réplique.
    - ❖ Les nœuds qui n'ont pas d'espace de stockage suffisant.
    - ❖ Les nœuds qui ne reçoivent aucune demande sur cette réplique (la fréquence de demande sur cette réplique est nulle).
    - ❖ Le cluster-head.

Nous supposons que le temps entre la prédiction de la panne et la panne réelle doit être inférieur au temps de la réaction du père.

4.2 Détection des pannes

Fischer, Lynch et Paterson [12] ont prouvé que dans un système réparti asynchrone, il n'existe pas d'algorithme déterministe qui résolve le problème du consensus lorsqu'un seul processus tombe en panne.
Les travaux de Chandra and Toueg [13] sont les premiers qui ont proposé « *unreliable failure detectors* » où ils ont montré que par un ajout de ce type de détecteur au système asynchrone, il est possible de résoudre le problème de consensus.
Dans notre travail la détection des pannes est faite par l'envoi des messages de vie [2] (ce modèle permet de minimiser le nombre des messages nécessaire pour la détection).
Chaque nœud envoie périodiquement un message de vie à son père, si après certain temps ce message n'arrive pas, le père déclare que son fils est nœud en panne et il déclenche la phase d'auto-stabilisation pour garder la connectivité de la topologie. La panne des Cluster-Head est détectée par la racine. Pour détecter la panne de la racine, elle envoie le message de vie à ses fils.

Nous avons proposé deux types de messages de vie selon l'état de l'émetteur. Un nœud peut avoir un des deux états, soit suspect (le degré de vivacité < un seuil) sinon le nœud est considéré comme correct.
Si le nœud émetteur est correct le message de vie sera de la forme : <MV1, ID émetteur, IN, OUT > où
MV1 : indique un message de vie d'un émetteur correct.
IN : contient la liste des répliques qui ont été ajoutées dans l'émetteur.

OUT : contient la liste des répliques qui ont été supprimées dans l'émetteur (soit par l'utilisateur ou par le système). Généralement IN et OUT contiennent un seul identificateur d'une réplique.

Cette technique permet de contrôler la dynamique des répliques.
Si l'émetteur est suspect, le message de vie sera de la forme :
< MV2, ID émetteur, numéro de séquence >.
MV2 : indique un message de vie d'un émetteur suspect.
Le numéro de séquence indique le nombre des messages envoyés après la prédiction de panne.
Cette valeur permet de détecter la prédiction incorrecte de panne. Ce type de message permet à un père suspect de contrôler un fils suspect.

On peut avoir trois états :

1. Si le père continuera à recevoir les messages de vie d'un fils suspect, alors après certain nombre de ces messages, le père déclare que la prédiction choisie sur l'état de son fils est incorrecte.
2. Si les messages de vie d'un fils suspect n'arrivent pas alors le père déclare que ce nœud est dans un état de panne et que la prédiction est correcte.
3. Si les messages de vie d'un fils correct n'arrivent pas alors il est déclaré en panne malgré qu'il n'était pas suspect et les répliques de ces nœuds seront perdus.

4.3 L'auto-Stabilisation

En cas de panne d'un nœud, la topologie logique sera déconnectée, c'est-à-dire, qu'il y aura des nœuds qui n'ont aucune arête connectée avec la topologie (sauf dans le cas de panne d'une feuille). L'auto stabilisation permet de passer de cet état illégitime à un état légitime où tous les nœuds sont liés (Figure 3).

- Cette phase ne sera déclenchée qu'après une panne.

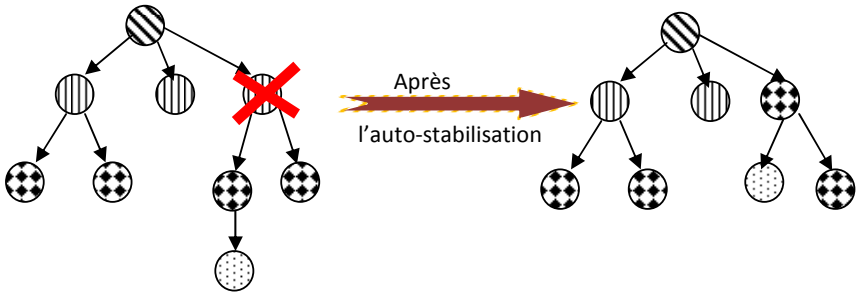


Fig.3. L'auto-Stabilisation après panne

L'auto-stabilisation n'exige pas un état initial et elle est adaptée à la nature dynamique du réseau [14]. Le seul inconvénient de cette technique est qu'elle donne des résultats incorrects pendant la stabilisation du système (phase transitoire du système).

4.4 Gestion de la dynamicité des nœuds

La dynamicité des répliques est gérée par l'envoi des messages de vie qui ont comme émetteur un nœud correct.
A chaque fois que le père reçoit ce message il vérifie les listes IN et OUT pour la mise à jour de sa vue des répliques des fils.
La dynamicité des nœuds correspond au scénario suivant :
Si un nouveau nœud arrive, il envoie une demande d'inscription à tous ces voisins physiques. Les voisins qui acceptent cette inscription, répondent par un message qui contient : son ID, le niveau, charge et la connectivité.
Le nouveau nœud choisit comme père le nœud qui a moins de charges et de connectivité et il commence à lui envoyer les messages de vie.
Si un nœud sort du système, il signale à son père par l'envoie de messages de désistement et le père doit réagir de la même façon que le cas d'une panne.

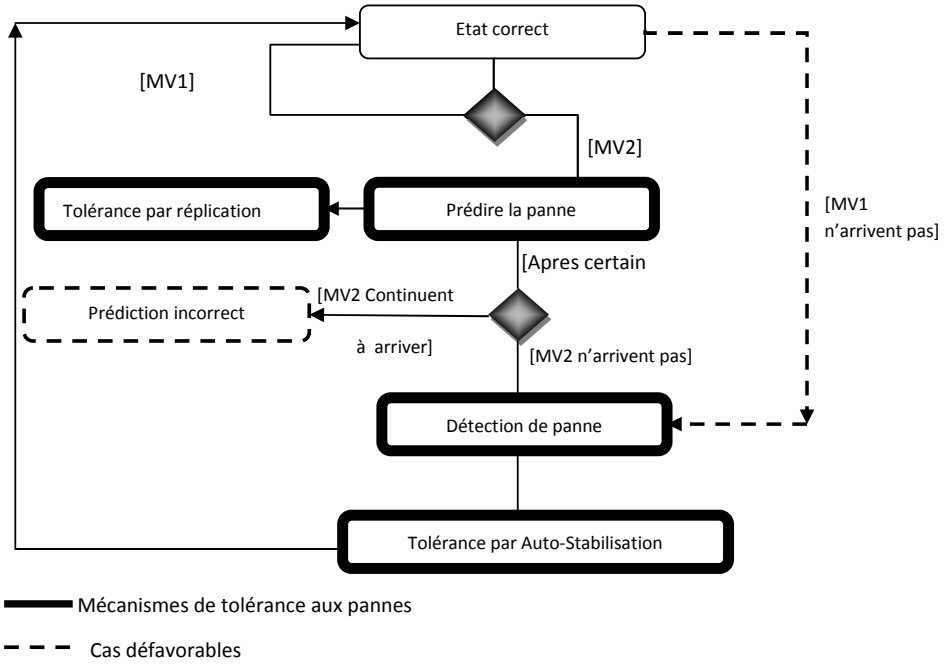


Fig.4. Notre proposition de tolérance aux pannes

5 Conclusion

Dans ce présent rapport, nous avons présenté notre proposition de tolérance aux pannes qui permet de réagir avant que la panne se présente pour protéger les répliques et augmenter la disponibilité des données, et après l'arrivée de la panne pour protéger la connectivité de la topologie.

Notre approche est en cours de réalisation. En perspective, à moyen court, nous pensons à étudier le comportement de notre approche d'un point de vue passage à l'échelle, et de la comparer avec les autres approches classiques. Comme perspective à long terme, nous proposons d'étendre l'approche proposée par un aspect intelligent dans les prises de décision et cela par l'intégration des agents au sein de chaque cluster dans un but d'implémenter une intelligence distribuée coopérative entre les agents.

References

1. E.W. DIJSTRA. Self stabilizing systems in spite of distributed control. Communications of the ACM, 17(11):643–644, 1974.
2. B. HAMID. Distributed Fault-Tolerance Techniques for Local Computations, these de doctorat, Juin 2007
3. M. LITZKOW, M. LIVNY, M. MUTKA. Condor - A Hunter of Idle Workstations . In 8th International Conference of Distributed Computing Systems, June 1988.
4. I. FOSTER ,C. KESSELMAN. Globus: A Metacomputing Infrastructure Toolkit. International Journal of Supercomputer Applications, 11(2):115–128, 1997.
5. N.H. KAPADIA , José A. B. FORTES. PUNCH: An architecture forWeb-enabled wide-area network-computing. Cluster Computing, 2(2):153–164, 1999.
6. A. BOUTEILLER, T. HERAULT, G. KRAWEZIK, P.LEMARINIER, F. CAPPELLO. MPICH-V Project: A Multiprotocol Automatic Fault Tolerant MPI. International Journal of High Performance Computing Applications, 20(3):319– 333, 2006.
7. G. Tarun VENKATESWARA REDDY M., Vijay Srinivas A. Janakiram D. Vishwa: A Reconfigurable Peer-to-Peer Middleware for Grid Computations. In Proceedings of the 35th International Conference on Parallel Processing, pages 381–390, Ohio, USA, August 2006. IEEE Computer Society.
8. N. DROST, R. V. van NIEUWPOORT, H. E. BAL. Simple locality-aware co-allocation in peer-to-peer supercomputing. In Proceedings of the Sixth International Workshop on Global and Peer-2-Peer Computing (GP2P), volume 2, page 14, Singapore, May 2006.

9. S. BASU, S. BANERJEE, P. SHARMA, S.Ju LEE. NodeWiz: peerto- peer resource discovery for grids. In Proceedings of the IEEE International Symposium on Cluster Computing and the Grid 2005 (CCGrid 2005), pages 213–220, Cardiff, UK, May 2005.

10. O. MARIN. The DARX Framework: Adapting Fault Tolerance For Agent Systems. These de doctorat, 'Université de HAVE, December 2003.

11. J.F. Deverge. Cohérence et volatilité dans un service pair-à-pair de partage de données. IRISA, Projet Paris, Juin 2004.

12. M. J. Fischer, N. A. Lynch, M. S. Paterson. Impossibility of distributed consensus with one faulty process. Journal of the ACM, 32(2):374-382,April 1985.

13. T. D. CHANDRA and S. TOUENG. Unreliable failure detectors for reliable distributed systems. Journal of the ACM, 1996.

14. F. REICHENBACH. Service SNMP de détection de faute pour des systèmes répartis. Ecole polytechnique de LAUSANE, Fevrier 2002

15. E. JEANVOINE. Intergiciel pour l'exécution efficace et fiable d'applications distribuées dans des grilles dynamiques de très grande taille. thèse de doctorat, l'Université de Rennes, novembre 2007.

Un protocole de routage ER-AODV à basse consommation d'énergie pour les réseaux mobiles Ad hoc

Said Khelifa, Zoulikha Mekkakia Maaza

Université des Sciences et de la Technologie d'Oan Mohamed Boudiaf
USTO-MB Oran,Algérie

said.khelifa@yahoo.fr, mekkakia@univ-usto.dz

Abstract. Au regard de l'importance de la conservation d'énergie dans les réseaux mobiles ad hoc, nous nous sommes intéressés à décrire ER-AODV (Energy Reverse Ad-hoc On-demand Distance Vector routing) un protocole de routage réactif qui repose sur une politique qui combine deux mécanismes appliqués au protocole AODV.
AODV et la plupart des protocoles de routage à la demande utilisent un chemin de réponse unique le long du chemin inverse. À cause de l'évolution rapide de la topologie, le paquet réponse de route peut ne pas arriver au nœud source, (c'est à dire après plusieurs envois de messages de requête de route, le nœud obtient un message de réponse). Cela augmente la consommation d'énergie. Pour éviter ce problème nous proposons un mécanisme qui, essaie de répondre par plusieurs routes réponses.
Le deuxième mécanisme, vise à incorporer l'énergie comme métrique de routage dans le processus de sélection de la route. En effet les énergies résiduelles des nœuds mobiles ont été considérées lors de la prise des décisions de routage.
Les résultats de simulation montrent que le protocole ER-AODV répond à une meilleure conservation d'énergie.

Keywords: conservation d'énergie, Réseaux Ad hoc, ER-AODV, Reverse AODV, Energy AODV.

1 Introduction

De nos jours, le domaine de télécommunication à pris un nouvel envol grâce à l'évolution technologique. De plus en plus, les environnements de communication utilisent les réseaux sans fil (réseaux ad hoc) plutôt qu'une infrastructure câblée pour communiquer. Un réseau ad-hoc est une collection de nœuds mobiles formant un réseau à topologie variante et fonctionnant sans station de base et sans administration centralisée,
Si l'idée générale d'un réseau ad hoc est triviale, il n'en est pas de même pour la réalisation et le déploiement d'un tel système. Les réseaux ad hoc sont des réseaux caractérisés par des ressources limitées en énergie puisque les nœuds mobiles opèrent habituellement avec des batteries. C'est un intérêt particulier pour ces réseaux où on

s'attend à ce que les dispositifs soient déployés pendant de longues périodes. La conservation d'énergie s'avère donc être un facteur primordial pour la durée de vie du réseau. Beaucoup de travaux de recherches ont été consacré à cette issue, Il existe plusieurs solutions qui s'intéressent à la consommation d'énergie, et qui peuvent être divisées en trois catégories: (i) Mise en marche / arrêt de l'émetteurs radio pour économiser l'énergie [1] [2], (ii) des protocoles basés sur le contrôle de la topologie , qui visent à réduire la portée des nœuds, tout en maintenant une connectivité complète du réseau [3] [4], et (iii) des protocole de routage avec une consommation minimale d'énergie [5].

Dans cet article, nous considérons le coût des paquets de données expédiés dans le réseau et le coût des paquets de contrôle employés pour maintenir le réseau. Pour cela, nous nous sommes intéressé à décrire ER-AODV (Energy Reverse Ad-hoc on-demand Distance vector routing) un protocole de routage réactif qui repose sur une politique qui combine deux mécanismes appliqués au protocole AODV [6]. Nous choisissons AODV parmi tous les autres protocoles de routage à la demande parce que, AODV consomme moins d'énergie que les autres protocoles de routage similaires, telles que DSDV et Tora, comme indiqué dans [7].

AODV et la plupart des protocoles de routage à la demande utilisent un chemin de réponse unique le long du chemin inverse. À cause de l'évolution rapide de la topologie le paquet réponse de route (RREP) peut ne pas arriver au nœud source, (c'est à dire après plusieurs envoie de messages de requête de route (RREQ), le nœud source obtient un RREP). Cela augmente la consommation d'énergie. Pour éviter ce problème nous proposons un mécanisme qui, essaie de répondre par plusieurs routes réponses. De cette manière on obtient un chemin de routage avec le moins de messages RREQ.

AODV est fondé sur le principe de vecteurs de distance c'est à dire du nombre des sauts entre l'émetteur et le récepteur. Afin d'améliorer davantage le routage en terme de conservation d'énergie nous proposons un deuxième mécanisme, qui vise à incorporer la métrique de l'énergie au lieu du nombre des sauts dans le processus de sélection de la route. En effet les énergies résiduelles des nœuds mobiles ont été considérées lors de la prise des décisions de routage.

Le reste de cet article est organisé comme suit. Dans la section 2, nous décrivons les caractéristiques les plus importantes du protocole de routage AODV et de ses principales limitations. Dans la section 3, nous décrivons nos deux mécanismes en détail. L'évaluation des performances du protocole ER-AODV sera abordée par des simulations sous NS2 dans la section 4. À ce stade de notre étude, nous discutons seulement les résultats de notre premier mécanisme. La section 5 conclut cet article en résumant les résultats.

2 Le protocole de base AODV

Le protocole de Routage AODV (Ad hoc On Demand Distance Vector) [6] est un protocole de routage réactif unicast, il est considéré comme une amélioration du protocole DSDV. Le protocole AODV, réduit le nombre de diffusions de messages, et cela en créant les routes lors du besoin.

L'AODV est basé sur l'utilisation de deux mécanismes « Découverte de route » et «Maintenance de route». Ce protocole utilise les principes des numéros de séquence afin de maintenir la consistance des informations de routage. A cause de la mobilité des nœuds, les routes maintenues par certains nœuds deviennent invalides. Les numéros de séquence permettent d'utiliser les routes les plus nouvelles ou autrement dit les plus fraîches.

L'AODV utilise une *requête de route* (RREQ) dans le but de créer un chemin vers une certaine destination. Cependant, l'AODV maintient les chemins d'une façon distribuée en gardant une table de routage, au niveau de chaque nœud intermédiaire appartenant au chemin cherché.

A chaque utilisation d'une entrée de la table de routage, son temps d'expiration est remis à jour. Si une nouvelle route est nécessaire, ou qu'une route disparaît, la mise à jour de ces tables s'effectue par l'échange de trois types de messages entre les nœuds :
• RREQ Route Request, un message de demande de route.
• RREP Route Reply, un message de réponse à un RREQ.
• RERR Route Error, un message qui signale la perte d'une route.

2.1 Fonctionnalité

Un nœud diffuse (broadcast) une RREQ, dans le cas où il aurait besoin de connaître une route vers une certaine destination et qu'une telle route n'est pas disponible. Cela peut arriver si la destination n'est pas connue au préalable, ou si le chemin existant vers la destination a expiré sa durée de vie ou il est devenu défaillant. Après la diffusion du RREQ, la source attend le paquet réponse de route (RREP). Si ce dernier n'est pas reçu durant une certaine période (appelée RREP_WAIT_TIME), la source peut rediffuser une nouvelle requête RREQ.

Quand un nœud intermédiaire envoie le paquet de la requête à un voisin, il sauvegarde aussi l'identificateur du nœud à partir duquel la première copie de la requête est reçue. Cette information est utilisée pour construire le chemin inverse, qui sera traversé par le paquet RREP de manière unicast. Une fois le paquet RREP arrivé à la destination, l'envoie des paquets de données peut démarrer.

2.2 Limitations d'AODV

Dans le protocole AODV, les routes sont établies en fonction du « nombre minimal des sauts »(le plus court chemin). Cependant, si le nombre des communications augmente le principe du plus court chemin n'est plus le critère optimal du choix des routes, il est préférable alors d'utiliser d'autres métriques qui ont un effet significatif sur la connectivité et la durée de vie du réseau. En outre, l'énergie est une contrainte très importante dans les réseaux ad hoc. Si un nœud, qui participe dans le processus d'établissement de routes, a une énergie très faible, la route peut être déconnectée très vite, cela peut avoir un effet néfaste sur la durée de vie du réseau. Pour faire face à ce problème, la métrique d'énergie devrait être prise en compte dans le processus d'établissement de routes. À cet effet, nous proposons un mécanisme qui considère les énergies résiduelles des nœuds mobiles lors de la prise des décisions de routage.

3 Le protocole ER-AODV

Prenant en compte les différents problèmes et contraintes décrites ci-dessus, nous proposons un protocole de routage réactif ER-AODV qui vise à maximiser la durée de vie du réseau et d'améliorer les performances obtenues par le protocole de routage de base AODV. Ainsi, le but est de diminuer le coût des paquets de contrôle employés pour maintenir le réseau en intégrant le mécanisme nommé "Reverse AODV", et de router autour des nœuds possédant une plus grande énergie résiduelle en intégrant le mécanisme "Energy AODV" à notre protocole.

3.1 Reverse AODV

3.1.1 Aperçu du mécanisme

Le principe de ce mécanisme est d'établir un chemin de routage avec le moins de messages RREQ, en essayant de répondre par plusieurs routes réponses afin d'y gagner en énergie consommé lors de l'expédition des paquets de contrôle.

Nous proposons ce mécanisme pour éviter la perte des paquets RREP, et améliorer les performances de routage dans les réseaux ad hoc. Comme illustré dans la figure 1, Reverse AODV utilise exactement la même procédure qu'AODV à diffuser le message RREQ lors de l'envoie de la réponse de route au nœud source. Lorsque ce dernier reçoit le paquet RREP, l'acheminement des paquets de données peut commencer immédiatement.

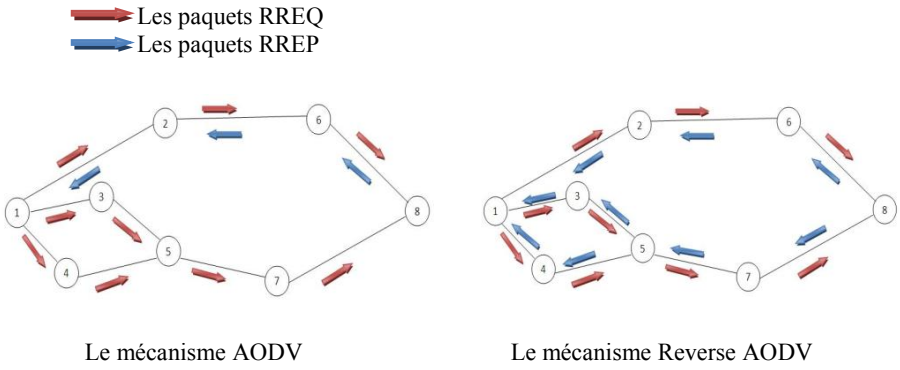


Figure 1. Exemple des deux mécanismes AODV & Reverse AODV.

3.1.2 Découverte des routes

Les nœuds mobiles qui utilisent le mécanisme Reverse AODV, échangent trois types de paquets de contrôle: les RREQs, les RREPs et les RRERs. Les RREQs et les RRERs ont le même format que RREQs et RRERs défini dans [6] pour AODV, alors que le format RREP a été légèrement modifié.

Le paquet RREP (Figure 2) contient les informations suivantes : reply source id, reply destination id, reply broadcast id, hop count, destination sequence number, reply time(timestamp).

Type	Reserved	Hop Count
Broadcast ID		
Destination IP address		
Destination Sequence Number		
Source IP address		
Reply Time		

Figure 2. Format du paquet RREP dans Reverse AODV.

Lorsque le paquet diffusé RREP arrive à un nœud intermédiaire, il vérifie la redondance. Si ce dernier a déjà reçu le même paquet, alors le paquet est ignoré, sinon il le transmet au prochain nœud. Par conséquent, les nœuds sauvegardent ou mettent à jour les informations suivantes de la table de routage :

- Adresse de la destination
- Adresse de la source
- Nombre de sauts jusqu'à destination
- Numéro de Séquence de la destination
- Délai d'expiration de la route et le prochain saut jusqu'à la destination.

Dés que le premier paquet RREP arrive à la source, la transmission des paquets de données commence, les autres paquets RREPs qui arrivent par la suite sont sauvegardés pour une utilisation ultérieure. Les chemins alternatifs peuvent être utilisés lorsque le chemin principal tombe en panne.

3.2 Energy AODV

Ce mécanisme propose une nouvelle approche adaptative qui vise à incorporer la métrique" l'énergie résiduelle des noeuds" au lieu du nombre des sauts dans le processus de sélection de la route. En effet, on définit le taux de consommation de l'énergie pour chaque nœud qui permet d'estimer sa durée de vie. Ensuite, on définit un coût qui correspond à cette durée ainsi qu'au niveau d'énergie. Cette information est alors utilisée pour le calcul des routes.

3.2.1 Paquets de contrôles & structures de données utilisées

Le paquet RREQ

Un champ appelé " min_bat " a été ajouté aux paquets RREQ. Il prend comme valeur l'énergie résiduelle minimale des nœuds traversés par le paquet RREQ.

Table de routage

Cette structure est utilisée pour stocker toutes les routes disponibles vers la destination, indexées par l'identificateur de la source. Chaque entrée dans la table de routage contient les champs suivants:

- **Src:** maintient l'identificateur du nœud source qui a initié le processus de découverte de routes.
- **Seq:** maintient le numéro de séquence du paquet RREQ.
- **Route:** contient la séquence des nœuds traversés par les paquets RREQ.
- **Min_bat**: stocke la valeur "énergie résiduelle minimale" des nœuds traversés par les paquets RREQ.
- **Arrival_time:** garde le temps d'arrivée des paquets RREQ au niveau du nœud destination.

Le contenu des quatre premiers champs est directement extrait des paquets RREQ qui arrivent.

3.2.2 Calcul de la durée de vie d'un nœud

Dans notre mécanisme, nous ne considérons pas uniquement le niveau d'énergie dans chaque nœud. Cependant, nous tenons en compte aussi le taux de consommation d'énergie à chaque période constante de temps (T_{update}) [8]. Pour chaque nœud, on suit la formule suivante pour calculer le taux de consommation d'énergie :

$$\text{Energie}_{\text{consom}}\ (j) = \frac{\text{Energie}_{\text{rest}}\ (J-1) - \text{Energie}_{\text{rest}}\ (j)}{T_{\text{Update}}}$$

Où $\text{Energie}_{\text{rest}}\ (J)$ est le niveau d'énergie restante calculé dans la période j comme suit :

$$\text{Energie}_{\text{rest}}\ (J) = \max\ \{\ \text{Energie}_{\text{cour}}\ (j) - \sum_{i=1}^{I=N\ pkts} E_{Tx}\ (i), 0\}$$

Où $\text{Energie}_{\text{cour}}\ (j)$ est le niveau courant de l'énergie et E_{Tx} est la quantité d'énergie nécessaire pour transmettre les N_{pkts} paquets qui sont déjà dans la file d'attente.

le taux de consommation d'énergie $\text{Energie}_{\text{consom}}\ (j)$ ainsi que le niveau d'énergie restante $\text{Energie}_{\text{rest}}\ (J)$ permet d'estimer, dans chaque intervalle de temps j , la durée de vie $T_{\text{lifetime}}\ (j)$ d'un nœud dans le réseau comme suit :

$$T_{\text{lifetime}}\ (j) = \frac{\text{Energie}_{\text{rest}}\ (J)}{\text{Energie}_{\text{consom}}\ (j)}$$

3.2.3 Calcul du coût d'une route

En utilisant la valeur $T_{lifetime}$ (j) , chaque nœud j peut calculer le coût de la route à chaque réception d'un paquet RREQ. Ce coût est calculé comme suit :

$$\mathrm{Cout}_{res-life}\ (j) \quad = T_{lifetime} \quad (j) * W_k$$

On note que W_k est un facteur multiplicatif dans l'intervalle [0,1] définit pour chaque niveau d'énergie. Par conséquent nous définissons quatre valeurs de W_k se référant à quatre intervalles d'énergie. Le premier est de 50% à 100% de la valeur initiale d'énergie (W_k =1). Le deuxième est de 30% à 50% (W_k =0.75), le troisième de 10% à 30% (W_k =0.5) et le dernier est de 0% à 10% (W_k =0.25).
Ainsi le coût d'une route de la source à la destination se calcule comme suit :

$$\frac{\sum_{i=1}^{i=nbr\ node} \mathrm{Cout}_{res-life}\ (j)}{\text{nbr node}}$$

Où nbr node est le nombre de nœuds intermédiaires.

4 Evaluation des performances

Dans le cadre de notre travail, l'évaluation des performances du protocole de routage ER-AODV sera abordée par des simulations sous NS2 [9]. À ce stade de notre étude, nous discutons seulement les résultats de notre premier mécanisme Reverse AODV.
Pour notre simulation, nous considérons des sources de type UDP, CBR et un modèle de mobilité "random waypoint ". Les scenarios utilisent une surface de simulation égale à : 1000m x 1000m. Le nombre de sources varient de 10 à 50, tous les liens sont considérés comme bi directionnels. Les critères de performance évalués sont :

- Le gain en termes de conservation d'énergie : la moyenne de l'énergie qui reste dans chaque nœud.
- Le ratio de paquets délivrés par rapports aux pertes.
- Le temps d'arrivée moyen des paquets de bout en bout.

Chaque simulation est exécutée pendant 100 secondes et répétée 10 fois. Nous comparons notre protocole ER-AODV avec AODV, on évalue ces performances en augmentant le nombre de nœuds. On considère au départ une topologie de 10 nœuds mobiles.

La figure 3 montre le ratio de paquets délivrés par AODV et ER-AODV. En augmentant le nombre de nœuds l'écart entre ces deux protocoles se creuse de plus en plus, les résultats sont mieux montrés dans la figure 4.

Le gain relatif au ratio de paquets délivrés par ER-AODV par rapport à AODV montré dans la figure 4 est calculé comme suit :

$$\frac{\text{ratio delivré par ER. AODV} \quad - \text{ratio delivré par AODV}}{\text{ratio delivré par AODV}} * 100\%$$

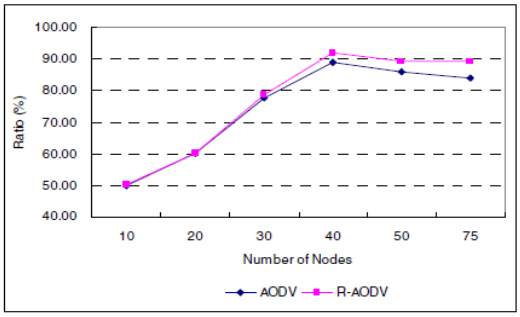


Figure 3. Le ratio de paquets délivrés en augmentant le nombre de nœuds

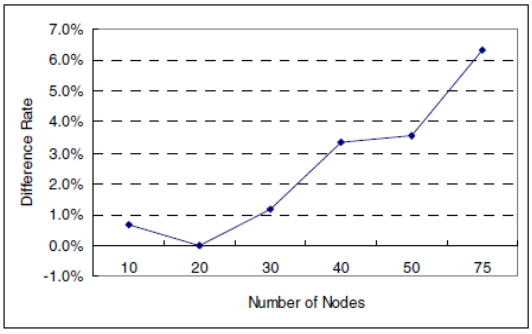


Figure 4. La différence dans le ratio de paquets délivrés entre les deux protocoles

La figure 5 montre la moyenne d'énergie qui reste dans chaque nœud à la fin de la simulation de nos deux protocoles, on peut voir que ER-AODV consomme moins d'énergie que AODV, malgré qu'il ya plus de paquets de données envoyés par ER_AODV que AODV comme nous l'avons vu dans la figure 3 et 4.

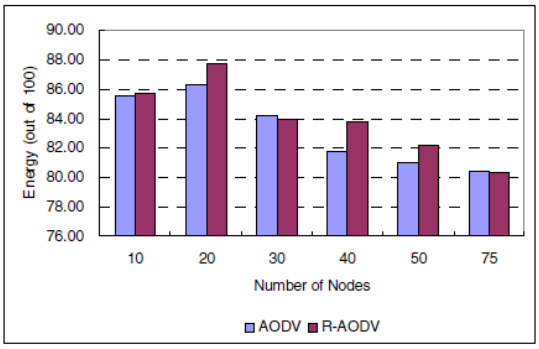


Figure 5. La moyenne d'énergie qui reste dans chaque nœud en augmentant le nombre de nœuds

La figure 6 montre Le temps d'arrivée moyen des paquets de bout en bout pour chaque protocole, on peut voir que ER-AODV a un temps d'arrivé inférieur à celui de AODV. La raison est que AODV choisit une route plus tôt, et que R-AODV choisit une route récente en fonction de la requête inverse.

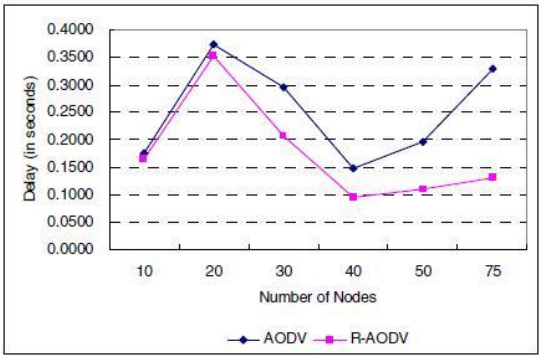


Figure 6. Le temps moyen d'arrivée des paquets de bout en bout en augmentant le nombre de nœuds

5 Conclusion

Dans les protocoles de routage à la demande, la réception du paquet RREP est très importante, car la perte des RREPS est très coûteuse. Si le paquet RREP est perdu alors le coût de la procédure de découverte des routes sera très élevé en termes de nombre de paquets de contrôle, et en termes d'énergie.

Nous proposons l'idée du mécanisme Reverse AODV, qui essaie de répondre par plusieurs routes réponse. La découverte de routes dans Reverse AODV est réalisée en moins d'itération que AODV. Les simulations ont montré que Reverse AODV répond à de meilleures performances que AODV. Nos travaux futurs consistent à implémenter le mécanisme Energy AODV afin de conserver davantage l'énergie consommée.

Références

1. L. M. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In IEEE INFOCOM, 2001.
2. Y. Xu, J. Heidemann and D. Estrin, "Adaptive energy-conserving routing
for multihop ad hoc networks," Technical Report TR-2000-527,2000.
3. Ramanathan and Rosales-Hain, "Topology control of multihop wireless networks using transmit power adjustment," IEEE Infocom 2000, 2000.
4. B. Chen, K. Jamieson, H. Balakrishnan, et R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001), Rome, Italy, juillet 16-21, 2001.
5. L. M. Feeney. "An energy consumption model for performance analysis of routing protocols for mobile ad hoc networks". Mobile Networks and Applications, 6(3):239–249, Juin 2001.
6. C. Perkins, Ad hoc on demand distance vector (AODV) routing, Internet-Draft, draft-ietf-manet-aodv-04.txt, pages 3-12, October 1999, Work in progress.
7. Ya Xu, John Heidemann, and Deborah Estrin, Adaptive Energy- Conserving Routing for Multihop Ad Hoc Networks, Research Report527, USC/Information Sciences Institute, October, 2000.
8. Lamia Romdhani and Christian Bonnet "Energy Consumption Speed-Based Routing for Mobile Ad Hoc Networks". Distributed Computing Systems Workshops, 2004. Proceedings. 24th International Conference on
9. NS, The UCB/LBNL/VINT Network Simulator (NS), http://www.isi.edu/nsnam/ns/, 2004.

Vers la spécification des exigences de sécurité des systèmes d'information

Salim CHEHIDA [1], Mustapha kamel RAHMOUNI [2]
[1] Département d'Informatique, Université de Mostaganem, Algérie
salimchehida@yahoo.fr
[2] Département d'Informatique, Université d'Oran Es-Sénia, Algérie
kamel_rahmouni@yahoo.fr

Abstract. De plus en plus notre société est dépendante d'Internet, Où le besoin de sécurité s'applique pour le commerce électronique, l'accès distant à une machine, le transfert de fichier, l'accès à certaines parties d'un site contenant des données confidentielles et d'autres applications. La spécification fonctionnelle des systèmes d'information n'est pas suffisante, la conception et la réalisation de ces systèmes doivent tenir compte, en plus des besoins fonctionnels, des différentes exigences de sécurité. La prise en compte des contraintes de sécurité (authentification, intégrité, confidentialité, non répudiation, disponibilité, etc.) au niveau de la modélisation constitue l'un des principaux challenges pour les concepteurs des SI (particulièrement lorsqu'ils sont connectés au web). UML s'est imposé comme le langage standard pour la modélisation des vues multiples d'un système à l'aide d'un ensemble de mécanismes d'extension. Dans ce travail, nous proposons une démarche et un ensemble d'extensions d'UML pour la spécification des exigences de sécurité des systèmes d'information.

Mots clés : Sécurité des systèmes d'information, Modélisation, Spécification, UML, UMLsec.

1. Introduction

La généralisation de la connexion à l'Internet offre des possibilités nouvelles et prometteuses, elles introduisent également un certain nombre de risques dont il faut prendre conscience, en mesurer les conséquences éventuelles, et en connaissance de cause prendre les mesures adéquates. Les entreprises se trouvent désormais confrontées au contrôle efficace de la confidentialité, de l'intégrité et de la disponibilité de ces informations. La sécurité à posteriori des SI (Firewall, Antivirus, etc.) peut donner des résultats mais elle ne peut remplacer à elle seule une véritable politique de sécurité. Nous pensons que l'élaboration d'une politique de sécurité pour un système d'information doit se faire en même temps que la modélisation, et que le modèle final doit intégrer les spécifications de sécurité. La sécurité des systèmes d'information doit donc commencer par l'élaboration d'un « modèle » en identifiant : Quelles sont les menaces ? Que doit-on protéger ? Pourquoi ? C'est donc le modèle - répondrant à ces

trois questions- qui donne un sens au mot « sécurité ». [22] UML est un langage standard pour visualiser, spécifier, construire et documenter un système logiciel. Malgré sa richesse, ce langage n'est pas adapté à tous les domaines ; il utilise un ensemble des mécanismes d'extension (les stéréotypes, les étiquettes et les contraintes) pour modéliser des différents aspects du système. *UMLsec* est une extension d'UML proposée par Jürjens (Munich University of Technology) comprennent un ensemble des profils pour la sécurité au niveau des modèles conceptuels. Pour la spécification des exigences de sécurité, *UMLsec* introduit deux mécanismes dans les diagrammes d'activité afin de sécuriser des transactions électronique ; le stéréotype **<<**fair exchange**>>** est utilisé pour assurer un échange équitable lors d'une transaction électronique et le stéréotype <<provable>> pour assurer la non répudiation dans les transactions de e-commerce. Le présent article propose une nouvelle démarche et un ensemble d'extensions (en plus de profils UMLsec) permettant la spécification des exigences de sécurité des systèmes d'information avec UML. Dans ce travail, nous avons aussi présenté des exemples de spécification de quelques besoins de sécurité du système ANEM (Agence Nationale d'Emploi) pour la mise en œuvre des extensions et de la démarche proposée.

2. Mécanismes d'extension d'UML

Ces mécanismes comprennent les stéréotypes, les étiquettes et les contraintes.

2.1 Stéréotype

Les stéréotypes permettent d'étendre la sémantique des éléments de modélisation et de définir de nouvelles classes d'éléments, en plus du noyau prédéfini par UML. En d'autre terme, un stéréotype est utilisé pour définir une utilisation particulière d'éléments de modélisations ou pour modifier la signification de ces éléments ; l'élément stéréotypé et son parent ont une structure identique mais une sémantique différente. Le nom du stéréotype est placé entre guillemets. Une icône peut être associée à un stéréotype (aucune icône n'est prédéfinie par UML) [17].

2.2 Etiquette

Une étiquette ou valeur marquée est une paire (nom, valeur) qui ajoute une nouvelle propriété à un élément de modélisation. Les propriétés permettent l'extension des attributs des éléments [18]. La spécification d'une valeur marquée prend la forme : nom = valeur. Une valeur marquée est indiquée entre accolades.

2.3 Contrainte

Une contrainte est une relation sémantique entre les éléments de modélisation. UML ne spécifie pas une syntaxe particulière pour les contraintes, qui peuvent ainsi

être exprimées en langage naturel, en pseudo-code, par des expressions de navigation ou par des expressions mathématiques. Chaque contrainte est indiquée entre accolades et placée près de l'élément (stéréotypé ou non) auquel elle est associée.

3. La démarche

La démarche est définie par une séquence d'étapes successives et ordonnées permettant la spécification des exigences de sécurité d'un système d'information. La figure suivante présente les différentes étapes de notre démarche.

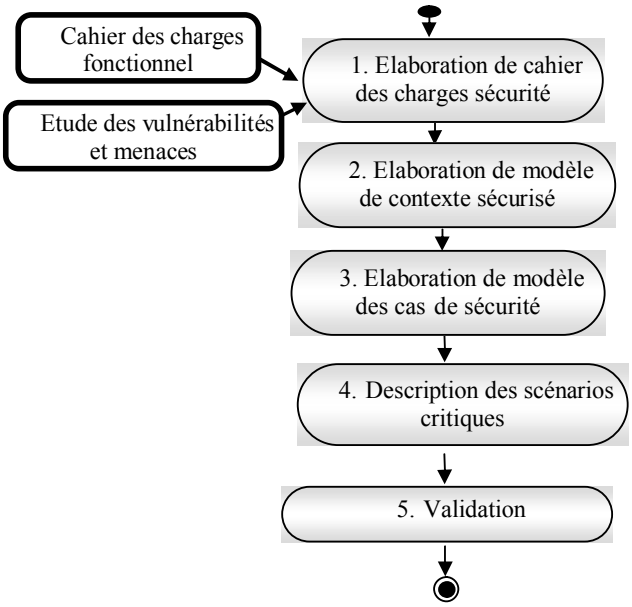


Fig. 1. La démarche de spécification des exigences de sécurité

- L'élaboration de cahier des charges est la première étape de la démarche. Cette étape jeu un rôle très important, c'est le point de départ de la spécification. Elle consiste à effectuer un premier repérage des différentes exigences de sécurité en utilisant le texte.
- La modélisation de contexte sécurisé consiste à définir les différents services de sécurité attendus du système considéré comme une boîte noire. L'objectif principal de cette étape est donc de préparer le terrain aux étapes de spécification des cas de sécurité et de description des scénarios critiques.

- Le modèle des cas de sécurité permet de structurer les services de sécurité fournis par le système (toujours envisagé comme une boite noire) pour les différents acteurs en un ensemble de cas de sécurité. Les cas de sécurité permettent de définir ce qu'on attend du système en terme de sécurité. Par exemple : Assurer l'authentification d'un utilisateur, Assurer l'intégrité et la confidentialité des informations échangées, Assurer la non répudiation d'une transaction,...
- Les scénarios critiques consistent à décrire les interactions ou les actions qui incluent un risque en mettant en jeu les différentes services ou propriétés de sécurité spécifiées par les cas de sécurité. Par exemple : les scénarios qui permettent d'assurer la non répudiation dans les transactions; il garantir que si une action est exécutée, elle ne peut pas être niée (elle est prouvée), les scénarios qui permettent d'assurer un échange équitable lors d'une transaction, les scénarios qui spécifient les interactions permettant l'échange des informations critiques (nécessite une confidentialité et une intégrité).
- Après l'identification des cas de sécurité et les scénarios critiques, le chef de projet valide ces cas avec le client ou les acteurs concernés. Si l'ensemble des exigences assurées par les cas de sécurité ne répond pas aux besoins de cahier des charges, l'équipe de spécification doit reprendre la spécification et corriger les erreurs.

4. Cahier des charges sécurité

Sur Internet, il est beaucoup plus difficile d'évaluer la sûreté des affaires. En outre des menaces sérieuses ont émergés du fait que le commerce électronique utilise un réseau public pour effectuer des transactions ayant un caractère critique. Un cahier des charges sécurité consiste à identifier les menaces, les vulnérabilités du système ainsi que les risques et les attaques possibles sur le système.

4.1 Menace

Une menace est un danger qui existe dans l'environnement d'un système indépendamment de celui-ci. Il représente l'ensemble des actions de l'environnement d'un système pouvant entraîner des pertes financières. Un système informatique sera d'autant plus menacé que les informations qu'il contient auront une valeur à la fois pour leur propriétaire et pour d'autres entités. [2]

4.2 Vulnérabilité

Une vulnérabilité est une erreur ou faille dans un système informatique permettant à un attaquant de porter atteinte à la sécurité de ce système, c'est-à-dire à son fonctionnement normal, à la confidentialité et l'intégrité des données qu'il contient et même à la disponibilité de ce système. Ces vulnérabilités sont la conséquence de faiblesses dans

la conception, la mise en œuvre ou l'utilisation d'un composant matériel ou logiciel du système, mais il s'agit généralement de l'exploitation de bugs logiciels.

4.3 Risque et attaque

Le risque est la combinaison de la menace et de la vulnérabilité. En l'absence de vulnérabilité, les menaces n'exposent à aucun risque. De même, en l'absence de menaces, la vulnérabilité n'expose à aucun risque. Mesurer un risque consiste à tenter d'identifier la probabilité qu'un événement dommageable survienne. [19] Parmi les risques, notons:

- vol d'informations confidentielles
  Les informations doivent être protégé car l'espionnage industriel est une réalité. Les données financières doivent aussi être protégées, ainsi que l'accès aux fichiers confidentiels.
- Modifications de données de haute importance
  Il s'agit des modifications stratégiques mais non détectables dans l'immédiat. Il y a une certaine gravité si ces modifications ne sont découvertes que trop tardivement.
- Rebonds
  Dans le cadre d'une attaque en règle, utilisation d'un système local à un établissement pour rebondir en cascade vers un autre site. L'établissement est donc responsable car étant le dernier dans la chaîne.
- Déni de service
  La qualité des systèmes informatique et du réseau doivent être garantie et maintenue. Ceci peut entraîner une perte de confiance dans l'outil informatique et réseau mais aussi une perte d'argent par les retards accumulés si les performances de l'outil informatique se dégradent.
- Destruction ou corruption d'informations
  Ceci peut occasionner des pertes de temps considérables pour restaurer les systèmes et les bases de données. De plus si des sauvegardes ne sont pas régulièrement assurées, cela peut devenir catastrophique.
- mascarade d'identité
  Cela peut porter un lourd préjudice à l'image d'un établissement, quant aux relations ultérieures avec des partenaires.
- Utilisation frauduleuse de ressources
  Intrusion extérieure ou intérieure sur une machine sans autorisation et utilisation de ressources réservées.

5. Modèle de contexte sécurisé

De nombreux auteurs, comme G.Bouch dans [Object solutions [21]] ou plus récemment P.Roques et F.vallée dans [UML en action [1]], ont préconisé l'utilisation de

diagramme de collaboration pour représenter de façon synthétique les différents besoins fonctionnels d'un système. Après l'élaboration du cahier des charges sécurité, on peut donc présenter les différentes exigences de sécurité sur un diagramme, que l'on peut qualifier de modèle de contexte sécurisé (secure context model). Le diagramme de collaboration est utilisé de la façon suivante :

- Le système est représenté par un objet central, cet objet est entouré par d'autres objets symbolisant les différents acteurs.
- Les objets sont reliés par des liens, sur chaque lien sont montrés des messages en sortie de système pour représenter les différents services de sécurité assurés par le système.

La figure suivante présente un exemple de modèle de contexte sécurisé.

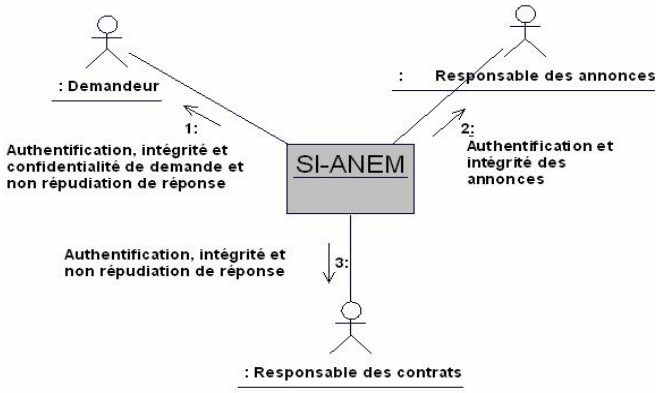


Fig. 2. Modèle de contexte sécurisé

6. Modèle des cas de sécurité

Cette étape consiste à identifier les cas de sécurité à partir de modèle de contexte sécurisé. Un cas de sécurité représente un service de sécurité rendu par le système pour un ou plusieurs acteurs. Pour cela, on utilise les cas d'utilisation de manière différente en introduisant les notions de cas de sécurité et de diagramme des cas de sécurité. Un cas de sécurité spécifie un comportement attendu du système pour répondre à des besoins de sécurité sans imposer le mode de réalisation de ce comportement. Il permet de décrire ce que le futur système devra faire en terme de sécurité informatique sans définir comment il le fait.

Les cas de sécurité sont absolument distincts des cas d'utilisation ; ils ne produisent pas une valeur ajoutée fonctionnelle mais ils recouvrent en effet tous service de sécurité dont un utilisateur bénéficie.

La figure 3 présente un exemple de modèle des cas de sécurité.

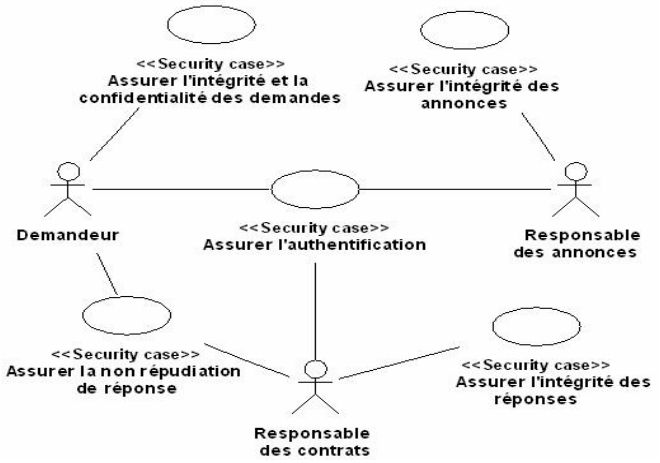


Fig. 3. Modèle des cas de sécurité

7. Scénarios critiques

Un scénario critique représente une succession particulière d'enchaînement (séquence d'actions d'interactions entre les acteurs et le système) qui incluent un risque en terme de sécurité informatique. Pour souligner ce risque, nous allons associer les propriétés de sécurité spécifiées par les cas de sécurité sur les interactions entre le système considéré comme une boîte noire et les différents acteurs. Par exemple : les scénarios qui permettent d'assurer la non répudiation dans les transactions électronique et les scénarios qui spécifient les interactions avec échange des informations critiques.

Pour la description des scénarios critiques, nous avons utilisé deux diagrammes dynamiques d'UML ; le diagramme d'activité qui est très utile en cas des actions parallèles [1] et le diagramme de séquence qui permet de bien visualiser les actions critiques.

La figure dans la page suivante présente un exemple de modèle qui souligne des échanges critiques.

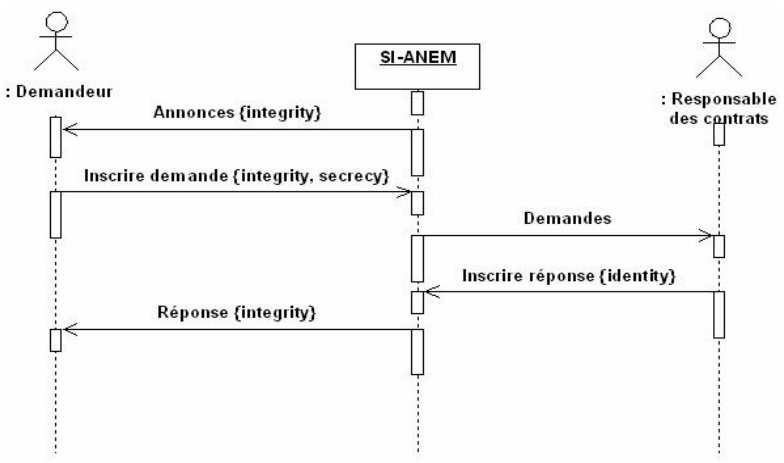


Fig. 4. Modèle des interactions critiques

Nous avons utilisé trois contraintes pour les interactions entre le système et les acteurs :

- La contrainte {secrecy} pour assurer la confidentialité des interactions.
- La contrainte {integrity} pour assurer l'intégrité des interactions.
- La contrainte {identity} pour assurer l'identité des parties lors de l'exécution d'une action d'interaction entre un acteur et le système.

La figure suivante présente un exemple de modèle qui exprime la non répudiation d'une transaction.

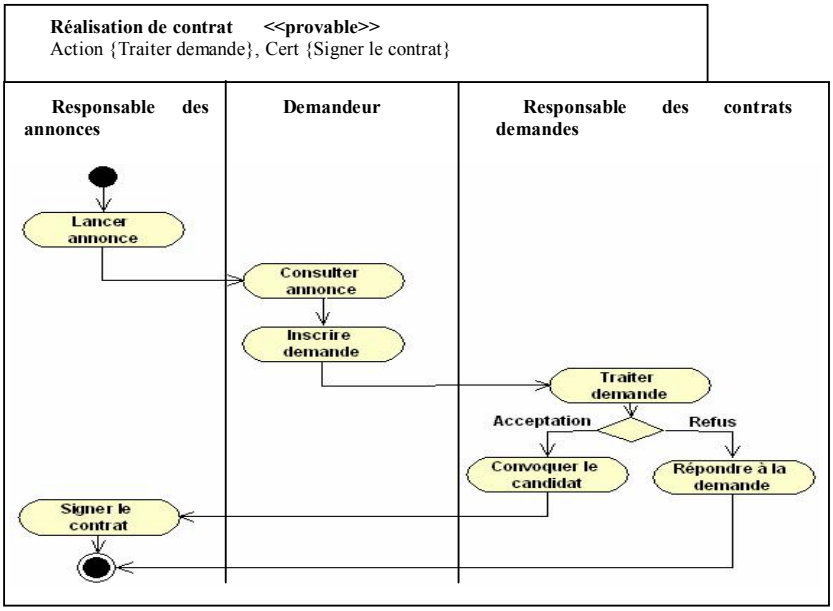


Fig. 5. Le stéréotype <<provable>>

Ce stéréotype d'*UMLsec* permet d'assurer la non répudiation dans les transactions de e-commerce ; il garantit que si une action est exécutée, elle ne peut pas être niée. Le package << provable >> définit deux étiquettes {action} et {cert} [10]. Pour tous scénario d'un diagramme d'activité : Un état d'activité contenu dans l'étiquette "cert" n'est atteint que si l'état d'activité contenu dans l'étiquette "action" est atteint avant lui.

8. Conclusion

UML n'est pas une notation fermée : elle est générique, extensible et configurable par l'utilisateur [17]. En cas de besoin, des précisions peuvent être apportées au moyen de mécanismes d'extension. Cet article présente des nouvelles profils d'UML pour la spécification des contraintes de la sécurité informatique, mais la porte reste ouverte pour d'autres extensions dans les différentes phases de développement.

La démarche proposée se fait par itération. Après l'identification des cas de sécurité, On valide ces cas. Si l'ensemble des exigences assurées par les cas de sécurité ne répond pas aux besoins de cahier des charges sécurité, on doit reprendre la spécification et corriger les éventuelles erreurs. La spécification des exigences de sécurité, quant à elle, produit des modèles pour le court terme, afin de définir les contraintes de sécurités imposées aux systèmes d'information après la capture des menaces provenant de l'environnement de ce système. Si l'environnement exige de nouvelles donnes en matière de sécurité, il convient d'intégrer la spécification des nouvelles exigences pour améliorer la sécurité de système. Les points importants qui reste à développer : Elaboration des nouvelles extensions d'UML pour les autres phases de développement ainsi que l'intégration de ces extensions dans un processus de développement.

9. Références

[1] P. Roques et F. Vallee, « *UML en action* », Eyrolles, (2002).

[2] Robert Longeon et Jean-Luc Archimbaud ; *« Guide de la sécurité des systèmes d'information a l'usage des directeurs »*, Cours CNRS, Site : http://www.cnrs.fr/Infosecu

[3] I. Jacobson, G. Booch, J. Rumbaugh, « *Le processus unifié de développement logiciel* », Eyrolles, (2000).

[4] A.Cockburn, *« Rédiger des cas d'utilisation efficaces »*, Eyrolles, (2001).

[5] P.Roques, « *UML par la Pratique* », Eyrolles, 2ème édition (2003).

[6] R.Medina, « *L'Extreme Programming* », Cours Crossbow Labs, (2008).

[7] P. kruchten , « *The Rational Unified Process : An Introduction* », Addison-Wesley, Second Edition (2000).

[8] S. Meng « *Security Requirements Analysis and Modeling of Distributed Systems* », Thèse de Master , Munich University of Technology Department of Informatics, Software & Systems Engineering, (2004).

[9] E.Maiwald, « *Sécurité des réseaux* », Campus Press, (2001).

[10] J. Jurjens, *« Secure Systems Development with UML: a Foundation »*, Thèse de doctorat, Munich University of Technology, (2003).

[11] S.Chehida, « Modélisation sécurisée des systèmes d'information Etude de cas: ANPE », Mini-projet Université d'Oran Es-Sénia – Ecole doctorale STIC, (2007).

[12] B.Debbabi, M.S.Boudjelda, « Le processus unifié de développement logiciel RUP », cours. (2007).

[13] G. Picard, « *le processus unifié*», Cours ENS Mines Saint-Etienne, (2008).

[14] P. Roques, « *Modéliser un site e-commerce*», Eyrolles, (2002).

[15] K.Scott, «*Unified Process Explained* », Addison-Wesley, (2002).

[16] C.Larman, « *UML et les Design Patterns* », Campus Press, (2002).

[17] Alain Muller et Nathalie Gaertner , *« Modélisation objet avec UML »*, Eyrolles (2004).

[18] Gerson Sygné, *« UML ï Mécanismes dêextension »*, cours (2005).

[19] Baroudi Rachid, *« Sécurité dûn système de remboursement par UMLsec »*, Mini projet Université d'Oran Es-Sénia – Magister, (2006).

[20] I. Jacobson, G. Booch, J. Rumbaugh, « *Unified Modeling Language Users Guide*», Addison Wesley Longman, (1999).

[21] G. Booch, *« Object Solutions: Managing the Object-Oriented Project»*, Addison Wesley, (1996).

[22] CNRS, « Sécurité informatique : numéro 31,… ,35 », Site : http://www.cnrs.fr/Infosecu, Revue (2001).

Optimisation III

Vers un Nouveau Protocole Pour Contrer l'Inversion de Priorité

DOUKHANI AMEL & GHOUALMI.NACIRA
ameldoukhani@yahoo.fr – ghoualmi@yahoo.fr

Université Badji Mokhtar
Faculté des sciences de l'ingénieur
Département d'Informatique, ANNABA, ALGÉRIE
Informatique Industriel

Résumé
Dans un ordonnancement temps réel préemptif, la présence simultanée de priorités fixes et de ressources partagées à accès exclusif peut entraîner un phénomène appelé ***inversion de priorité***. Devant ce problème, nous présentons dans cet article un nouveau protocole d'allocation de ressources nommée *protocole à plafond dynamique* « ***CDP : Ceiling Dynamic Protocol*** » pour systèmes temps réels où la survenue du phénomène d'inversion de priorités et le blocage des tâches prioritaires sur une ressource partagée à accès exclusive sont très exceptionnels, des fois n'aura pas lieu. Le cadre de l'étude porte sur un ordonnancement préemptif de tâches périodiques avec des ressources partagées à plafond de priorité dynamique. Les objectifs de cette proposition sont : contrer l'inversement des priorités, éliminer l'inter-blocage, minimiser le temps de blocage et assurer l'exécution de la tâche la plus prioritaire sans aucune interruption.

Mots-clés : Ordonnancement temps réel, protocoles d'inversion de priorité, Plafond de priorité dynamique, Héritage de priorité, Temps mort.

1. Introduction

Souvent, on distingue des tâches utilisant des ressources partagées. A cet effet l'accès simultané à ces ressources doit être contrôlé afin de garder un état cohérent. Le partage de ressources en mutuelle exclusion peut engendrer des *inter-blocages* ou des *inversions de priorités*. La situation d'inversion de priorités survient lorsque l'allocation du CPU est basée sur des priorités, mais que les processus ne sont pas indépendants. Alors, lorsqu'une tâche T_1 demande une ressource déjà allouée à une autre tâche T_2, T_1 se met en attente de la ressource même si elle est prioritaire (figure1).

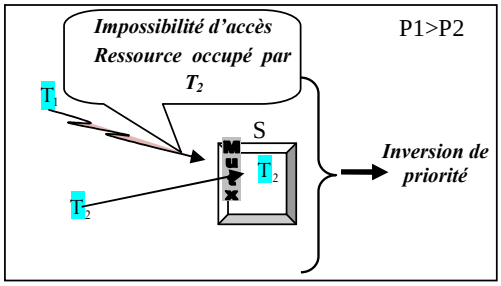


Figure 1 : *illustration de l'inversion de priorité*

Des fois ce problème mène à des dégâts inattendus. Ainsi, pour limiter les problèmes des deux phénomènes pré-cités, des protocoles tels que PIP **[1]**, OCPP et ICPP **[1,5]**, SRP **[2, 3, 4]** ont été introduits.

2. L'inversion de priorité

C'est une situation indésirable due au partage de ressource où l'accès d'une tâche à une ressource partagée est empêché par une tâche moins prioritaire, qui possède la ressource **[2]**. Les protocoles d'inversion de priorité ont été définis pour répondre à des problèmes supplémentaires ou non résolus. Ainsi, les risques d'inter-blocage et de chaînes de blocages avec PIP ont été résolus par PCP (Priority Ceiling Protocol : OCPP et ICPP) **[1, 3]**. De même, le nombre important de changements de contexte entres les tâches avec PCP a conduit à définir le protocole SRP. Ce dernier, fonctionnant aussi bien avec un ordonnanceur à priorité fixe que dynamique, se distingue de plus par sa facilité d'implémentation **[4]**, en partie due au fait que les blocages se font au niveau de la préemption et non au niveau de l'accès aux ressources. L'inconvénient de cette méthode est qu'une tâche qui n'utilise aucune ressource partagée peut se retrouver bloquée par le test de préemption. Lorsqu'une tâche préempte une autre, on est sûr que les ressources dont elle a besoin seront disponibles **[3, 4]**.

3. Etude comparatif

Le but du tableau 1 est de comparer les différents protocoles permettant de limiter les effets de l'inversion de priorités, par l'analyse de leurs caractéristiques de fonctionnement et leur difficulté d'implémentation. Dans ce tableau en notant $\pi(R_k)$ la priorité maximum des tâches accédant à R_k et $D_{j,k}$ la durée de la section critique associée. Concernant les deux indices ***n*** et ***m*** : selon **[1]** on a :

1) s'il y a **n** tâches de basse priorité pouvant bloquer une tâche T_i alors T_i peut être bloquée pour une durée maximale de **n** sections critiques;
2) s'il y a **m** ressources distinctes qui peuvent bloquer une tâche T_i alors T_i peut être bloquée pour une durée maximale de **m** sections critiques.

Le temps de blocage est alors égal à la durée de min {n, m} sections critiques.

	PIP	PCP	SRP
Blocages	Accès aux ressources	Accès aux ressources	Préemption
Types de blocage	Direct, héritage de priorité	Direct, héritage de priorité, plafond de priorité	Direct, Plafond de préemption
Nombre de blocages	Min{n, m}	1	1
Temps de blocage B_i	$B_i = \min\{ B_i^l, B_i^s \}$ $B_i^l = \sum_{j=i+1}^{n} \max_k \{D_{j,k} : \pi(R_k) \geq P_i \}$ $B_i^s = \sum_{k=1}^{m} \max_{j<i} \{D_{j,k} : \pi(R_k) \geq P_i \}$	$B_i = \max_{j,k} \{D_{j,k}: P_j < P_i$ et $\pi(R_k) \geq P_i \}$	$B_i = \max_{j, k} \{D_{j,k} :$ $P'_j < P'_i$ et $\pi(R_k) \geq P'_i \}$
Inter-blocage	Oui	Non	Non
Chaîne de blocage	Oui	Non	Non
Algorithme d'ordonnancement	RM	RM	RM/EDF
Priorité	Fixe	Fixe	Fixe/dynamique
Implémentation	Dur	Moyen	Facile

Tableau 1 : *Présentation d'une comparaison entre PIP, PCP et SRP*

4. Le protocole proposé pour contrer le phénomène d'inversion de priorité

Dans ce papier nous offrons un nouveau protocole « *protocole à plafond dynamique : **C**eiling **D**ynamic **P**rotocol (**CDP**)* » basé sur la réduction maximale du temps de blocage d'une tâche prioritaire. Le phénomène d'inversion de priorités est minimisé voir éliminé. Le protocole CDP pourrait engendrer un gaspillage du temps CPU (*temps morts*). La notion de temps mort est apparu dans le cas où les tâches à s'exécuté ont des dates d'activations proches dans le temps et leurs sections critiques sont de longue durée d'exécution.

Dans la suite du papier nous rappelons que la tâche i est notée $\mathbf{T_i}$, la tâche en attente est notée $\mathbf{T_a}$, la ressource k est notée $\mathbf{R_k}$, la priorité de la tâche i est notée $\mathbf{P_i}$, La priorité plafond courante (la plus haute priorité des tâches à l' instant t) est notée π', le nombre de tâche qui sollicite la section critique R_k est notée $Nb_{T_{user}(R_k)}$ et la l'instant d'arrivé de la tâche i est notée **PDM(Ti)**, le nombre de périodes successives requises à l'utilisation de la ressource R_k est notée D_{R_k}.

4.1-Principe du CDP

L'idée de base de CDP est de rendre dynamique le plafond de priorité d'une ressource R_k interdisant une tâche T_i d'entrer en section critique R_k si celle-ci sera sollicitée ultérieurement par une autre tâche de priorité supérieure. Dans le cas où T_i peut libérer la section critique désirée avant qu'une tâche prioritaire n'atteigne sa date d'activation, elle peut la prendre. Chaque ressource R_k a une valeur dynamique qui est la priorité plafond, notée $\pi(R_k)$ et nommée *priorité plafond* de R_k égale à la plus haute priorité des tâches nécessitant R_k pour accomplir leurs exécutions ; c'est-à-dire que $\pi(R_k)$ change (diminue) dès que une tâche prioritaire courante quitte définitivement la ressource R_k . De manière plus formelle :

$$\pi(R_k) = \begin{cases} max_{T_i \text{ nécessite } (R_k)} P_i \\ \\ \varphi \end{cases} \qquad (1)$$

Où : P_i : Priorités des tâches nécessitent R_k pour accomplir leurs exécutions. φ : est une priorité plus basse que la priorité de la tâche la moins prioritaire des tâches qui ont utilisé R_k

Remarque : Si toutes les tâches nécessitent R_k sont terminées leurs exécutions de dans alors $\pi(R_k) = \varphi$.

Ce protocole suppose que chaque tâche possède une priorité fixe. Les ressources utilisées, les périodes de départ minimal, les séquences d'exécutions ainsi que les durées dont les ressources sont utilisées sont connues avant le début de l'exécution.

4.2- Protocoles d'ordonnancements proposés

- Si deux tâches ou plus veulent commencer (ou continuer) leur exécution à un instant t alors c'est la tâche prioritaire qui prend la main pour s'exécuter.
- Une tâche T_i de priorité statique P_i peut préempter l'exécution de la tâche courante T_j de priorité statique P_j et commence son exécution si et seulement si $P_i > P_j$.
- Lorsqu'une tâche T_i de priorité statique P_i veut prendre une ressource R_k à un instant t, on a l'une des situations suivantes :
  - Si $Nb_{T_{user}(R_k)} = 1$ alors T_i peut prendre la ressource R_k et entre en section critique.
  - Si $(Nb_{T_{user}(R_k)} \geq 1) et (P_i = \pi(R_k))$ alors T_i peut prendre la ressource R_k.
  - Si $(Nb_{T_{user}(R_k)} \geq 1) et (P_i < \pi(R_k))$ alors on a trois cas possibles :
    - **(a)** Si l'exécution de T_i dans R_k se termine avant ou avec les périodes de départ minimale de toutes les tâches prioritaires à Ti alors T_i peut prendre la ressource R_k et entre en section critique. De manière Formelle :

$$P_j > P_i : D_{R_k} \leq \left[min\left(PDM(T_j)\right) - t \right] \qquad (2)$$

(b) s'il y a des tâches T_g de priorités inférieures ou égales à P_i qui sont :

- interrompues par T_i ou par des tâches de priorité inférieures à P_i ;
- bloquées sur une ressource R_k ;

Alors on commence le test d'allocation pour la tâche la plus prioritaire entre eux :

- Si $T_a.etat = prête$ alors T_a peut commencer son exécution.
- Si $T_a.etat = préemptée$ alors T_a peut reprendre son exécution.
- Si $T_a.etat = bloquée\ sur\ R_{k'}$ (k' peut être égal à k) alors Si $D_{R_{k'}} \leq [min(PDM(T_j)) - t]$ alors la tâche T_a rehausse sa priorité à celle de π'(héritage de la priorité π'). Elle est alors exécutée avec la priorité hérité jusqu'à ce que une tâche de priorité supérieure à π' atteigne sa date d'activation, dans ce cas sa priorité redevient celle qu'elle était.

(c) La non prise de R_k par T_i provoque un temps mort qui dure :
[Le minimum des périodes de départ minimales des tâches qui n'ont pas encore déclenché leur exécution - l'instant où T_i veut prendre R_k]. De manière plus formelle :

$$P_j > P_i : D_{TM} = \left[min\left(PDM(T_j)\right) - t\right] \quad (3)$$

Tel que : T_j n'ont pas encore atteint leurs dates d'activation.

4.3-Caractéristiques du CDP

- Le CDP empêche les situations d'inter-blocage, qui peuvent survenir lorsqu'il y a plusieurs ressources partagées. Il peut provoquer le cas de blocage sur la ressource R_k, lorsque la durée de la section critique de la tâche courante est assez élevée et peut dépasser la l'instant d'arrivé d'une tâche prioritaire.
- Le CDP accompli l'exécution de la tâche la plus prioritaire sans aucune interruption.
- Le temps de blocage c'est la durée pendant laquelle une tâche de priorité moyenne T_m ne peut plus progresser à cause de la condition (*La durée d'exécution de Tm dans R_k > l'instant d'arrivé d'une tâche prioritaire*), une tâche de moindre priorité bénéficiant de cette situation de blocage et reprend son exécution avec une priorité plus élevé (hérité de π'). On peut calculer le temps de blocage B_m de T_m sur R_k comme suit : à chaque fois que le protocole d'ordonnancement (b) ou (c) est exécutée on compte la durée de cette exécution, le temps de blocage de T_m est la somme de ces durés. De manière plus formelle :

$$P_j > P_m : B_m = B_m + [min\left(PDM(T_j)\right) - t] \quad (4)$$

4.4-Étude Comparative Complémentaire

Le tableau 2 montre les différentes particularités de fonctionnements associées au protocole CDP.

	CDP (*Ceiling Dynamique Protocol*)
Blocages	Accès aux ressources
Types de blocage	Section critique longue
Nombre de blocage de la tâche prioritaire	0
Nombre de blocage des tâches moyennes	Variable (en fonction des PDM et de durées de section critique)
Temps de blocage de Tm	$P_j > P_m : B_m = B_m + [min(PDM(T_j)) - t]$
Inter-blocage	Non
Chaîne de blocage	Non
Algorithme d'ordonnancement	En perspective
Priorité	Fixe
Temps d'exécution de la tâche prioritaire	Nombre de périodes requises à l'exécution
Implémentation	En perspective

Tableau 2 : *les caractéristiques attribuées au protocole CDP*

4.5- Avantages

- ✓ CDP ne mène jamais à un inter-blocage ;
- ✓ CDP assure l'exécution de la tâche la plus prioritaire sans aucune interruption ;
- ✓ Bien adapté lorsque les PDM des tâches sont trop proches dans le temps et les sections critiques sont de courte duré d'exécution;
- ✓ Temps de blocage très réduit ;
- ✓ Minimise le nombre de commutation de contexte.

4.6- Inconvénients

- × Le gaspillage de temps CPU (*temps mort*) implique un temps d'exécution total trop élevé.
- × Les sections critiques de longue durée d'exécution provoquent souvent des cas de blocage.

5- Exemple Proposé

L'idée de l'exemple a été prise de la référence **[1,3]**. On a sélectionné quatre tâches avec leurs priorités, périodes de départ minimal ainsi que leurs séquences d'exécution. Le tableau 3 propose cet exemple :

Tâches	*Priorités*	*Périodes de départ minimal*	*Séquences d'exécution*
T_0	25	4	ER_0R_1E
T_1	15	2	ER_1R_1E
T_2	10	2	EE
T_3	4	0	$ER_0R_0R_0E$

Tableau 3 : *les 4 tâches avec leurs Pi, PDM et séquences d'exécution associées*

Dans les figures qui suivent les différentes couleurs signifient :
Exécution normale, Tâche bloquée, che préemptée,
Section critique R_0, ection critique R_1.

5.1- L'exemple avec les priorités assignées aux tâches :

La figure 2 montre l'ordre d'exécution des 4 tâches avec leurs priorités assignées:

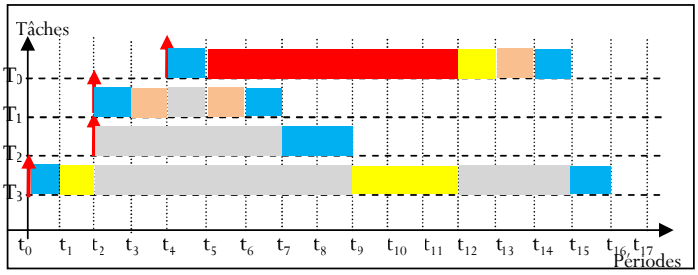


Figure 2 : l'ordonnancement des 4 tâches avec leurs priorités assignées

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 peut prendre R_0 et entre en section critique. A l'instant t_2 : T_1 préempte T_3 et commence son exécution car $P_1>P_3$. A l'instant t_3 : T_1 peut prendre R_1 et entre en section critique. A l'instant t_4 : T_0 est réveillé et préempte T_1 car $P_0>P_1$, T_0 commence son exécution. A l'instant t_5 :T_0 ne peut prendre R_0 car la ressource n'est pas libre. T_1 reprend son exécution dans la section critique R_1 et T_0 se retrouve bloquée. A l'instant t_6 : T_1 libère R_1 et continue son exécution dans aucune section critique durant une période. A l'instant t_7 :T_1 termine son exécution. T_2 s'exécute pendant deux périodes dans aucune section critique. A l'instant t_9 : c'est la tâche T_3 qui continue son exécution dans R_0 durant 3 périodes car T_0 se retrouve bloquée sur cette ressource. A l'instant

t_{12} : T_3 libère la ressource R_0, T_0 préempte T_3 et entre en section critique en prenant la ressource R_0. A l'instant t_{13} : T_0 libère la ressource R_0, prend la ressourcé R_1 et entre en section critique pendant une période. La durée du blocage dû à l'inversion de priorité est égale à t_{12}-t_5. A l'instant t_{14} : T_0 libère R_1 et continue son exécution dans aucune section critique durant une période. A l'instant t_{15} : T_0 termine son exécution. T_3 s'exécute pendant une période dans aucune section critique. A l'instant t_{16} : T_3 termine son exécution.

Conclusion :
Temps d'exécution de $T_0 = t_{15} - t_4 = 11$ périodes

On se serait attendu à ce que T_0 puisse faire sa séquence d'exécution en 4 périodes (ER_0R_1E), car c'est la tâche la plus prioritaire, mais 11 périodes ont été requises. Pire encore, T_1 qui est moins prioritaire que T_0 a terminé plus rapidement sa séquence d'exécution, d'où le nom inversion de priorité.

5.2- L'exemple Avec PIP (Priority Inheritance Protocol) :

La figure 3 suivante montre l'ordre d'exécution des 4 tâches selon le protocole PIP :

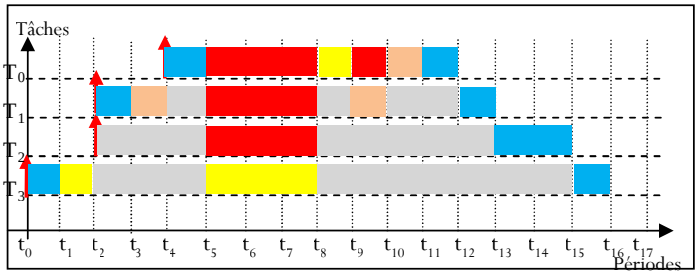


Figure 3 : l'ordonnancement des 4 tâches avec PIP

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 peut prendre R_0 car elle est libre et entre en section critique. A l'instant t_2 : T_1 préempte T_3 et commence son exécution car $P_1 > P_3$. A l'instant t_3 : T_1 peut prendre R_1 car elle est libre et entre en section critique. A l'instant t_4 : T_0 est réveillé et préempte T_1 car $P_0 > P_1$, T_0 commence son exécution. A l'instant t_5 : T_0 ne peut prendre R_0 car la ressource n'est pas libre. T_3 reprend son exécution mais avec la priorité P_0 de T_0 ($P_3 = P_0$) durant 3 périodes. A l'instant t_8 : T_3 libère la ressource R_0, reprend sa priorité initiale P_3 et est préemptée par T_0. T_0 peut prendre la ressource libre R_0 et entre en section critique. A l'instant t_9 : T_0 ne peut prendre R_1 car la ressource n'est pas libre. T_1 reprend son exécution mais avec la priorité P_0 de T_0 ($P_1 = P_0$) durant une période. A l'instant t_{10} : T_1 libère la ressource R_1, reprend sa priorité initiale P_1 et est préemptée par T_0. T_0 peut prendre la ressource libre R_1 et entre en section critique. A l'instant t_{11} : T_0 libère R_1 et continue son exécution dans aucune section critique durant une période. A l'instant t_{12} : T_0 termine son exécution. T1 s'exécute pendant une période dans aucune section critique. A l'instant t_{13} : T_1 termine son exécution. T2 s'exécute pendant deux périodes dans aucune section critique. A l'instant t_{15} : T_2 termine son

exécution. T_3 reprend la sienne pendant une période dans aucune section critique. A l'instant t_{16} : T_3 termine son exécution.

Conclusion :
Temps d'exécution de $T_0 = t_{12} - t_4 = 8$ périodes

5.3- L'exemple avec OCPP (Original Ceiling Priority Protocol) :

Sachant que *plafond de priorité* courant du système (current priority ceiling), noté π', est égal au maximum des plafonds de priorité des ressources utilisées, ou Ω si aucune ressource n'est utilisée (Ω est une priorité plus basse que n'importe qu'elle priorité).

La figure 4 suivante montre l'ordre d'exécution des 4 tâches selon le protocole OCPP:

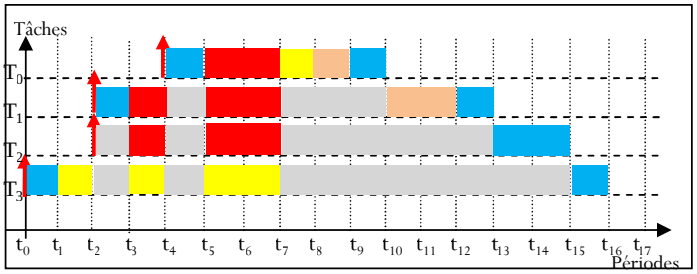


Figure 4 : l'ordonnancement des 4 tâches avec OCPP

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 peut prendre R_0 car $P_3 > \pi'$ et $\pi' = \Omega$ et entre en section critique. A l'instant t_2 : T_1 est réveillée, préempte T_3 et commence son exécution car $P_1 > P_3$. A l'instant t_3 : T_1 ne peut prendre R_1 car P_1 n'est pas supérieur à π' et $\pi' = \pi(R_0)$, T_3 hérite de la priorité P_1 de T_1 et reprend son exécution. A l'instant t_4 : T_0 est réveillée et préempte T_3 car $P_0 > P_3$, T_0 commence son exécution. A l'instant t_5 : T_0 ne peut prendre R_0 et se retrouve bloquée car P_0 n'est pas supérieure à π' et $\pi' = \pi(R_0)$, T_3 reprend son exécution avec la priorité P_0 durant deux périodes consécutives. A l'instant t_7 : T_3 libère R_0 et reprend sa priorité P_3. T_3 est alors préemptée par T_0 et T_0 prend R_0 car $P_0 > \pi'$ et $\pi' = \Omega$. A l'instant t_8 : T_0 libère R_0 et peut prendre R_1 car $P_0 > \pi'$ et $\pi' = \Omega$. A l'instant t_9 : T_0 libère R_1 et s'exécute pendant une période dans aucune section critique. A l'instant t_{10} : T_0 termine son exécution. T_1 reprend la sienne et peut prendre R_1 pendant deux périodes consécutives car $P_1 > \pi'$ et $\pi' = \Omega$. A l'instant t_{12} : T_1 libère R_1 et s'exécute pendant une période dans aucune section critique. A l'instant t_{13} : T_1 termine son exécution. T2 s'exécute pendant deux périodes dans aucune section critique. A l'instant t_{15} : T_2 termine son exécution. T_3 reprend la sienne pendant une période dans aucune section critique. A l'instant t_{16} : T_3 termine son exécution.

Conclusion :
Temps d'exécution de $T_0 = t_{10} - t_4 = 6$ périodes

5.4- L'exemple avec ICPP (Immediate Ceiling Priority protocol) :

La figure 5 suivante montre l'ordre d'exécution des 4 tâches selon le protocole ICPP :

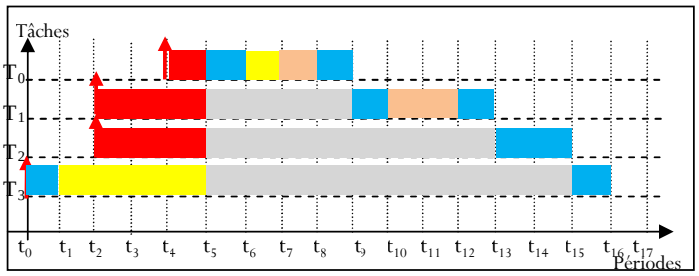


Figure 5 : l'ordonnancement des 4 tâches avec ICPP

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 peut prendre R_0 car $P_3>\pi$'et $\pi'= \Omega$ et entre en section critique avec $P_3=\pi(R_0)$ et $\pi(R_0)=P_0$. A l'instant t_2 : T_1 est réveillée, T_3 peut continuer à s'exécuter car $P_3=P_0$ et $P_0>P_1$ de T_1. A l'instant t_4 : T_0 est réveillée mais T_3 peut continuer à s'exécuter car $P_3 = P_0$. A l'instant t_5 : T_3 libère R_0 et reprend sa priorité initiale P_3. T_3 est alors préemptée par T_0 qui commence son exécution. A l'instant t_6 :T_0 prend R_0 car $P_0 > \pi$' et $\pi' = \Omega$. A l'instant t_7 : T_0 libère R_0 et peut prendre R_1 car $P_0 > \pi$' et $\pi' = \Omega$. A l'instant t_8 : T_0 libère R_1 et s'exécute pendant une période dans aucune section critique. A l'instant t_9 : T_0 termine son exécution. T_1 commence son exécution normale. A l'instant t_{10}: T_1 peut prendre R_1 pendant deux périodes consécutives car $P_1>\pi$' et $\pi'= \Omega$. A l'instant t_{12} : T_1 libère R_1 et s'exécute pendant une période dans aucune section critique. A l'instant t_{13} : T_1 termine son exécution. T_2 s'exécute pendant deux périodes dans aucune section critique. A l'instant t_{15} : T_2 termine son exécution. T_3 reprend la sienne pendant une période dans aucune section critique. A l'instant t_{16} : T_3 termine son exécution.

Conclusion *:*

Temps d'exécution de $T_0 = t_9 - t_4 = 5$ périodes

5.5- L'exemple avec SRP (Stack Resource Policy) :

La figure 6 suivante montre l'ordre d'exécution des 4 tâches selon le protocole SRP :

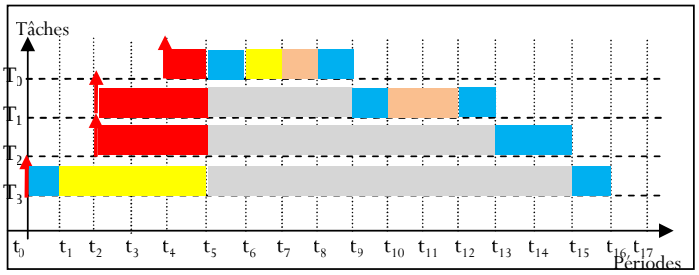


Figure 6 : l'ordonnancement des 4 tâches avec SRP

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 peut prendre R_0 et entre en section critique, le plafond de préemption du système devient $\pi' = \pi(R_0) = P'_0$. A l'instant t_2 : T_1 et T_2 sont activées, mais ($P'_1 < P'_3$) et ($P'_2 < P'_3$) donc ni T_1 ni T_2 ne peut préempter T_3. A l'instant t_4 : T_0 est réveillée, mais T_3 peut continuer à s'exécuter car $P'_3 > P'_0$. A l'instant t_5 : T_3 libère R_0 ; T_0, T_1 et T_2 sont les tâches prêtes à s'exécuter. Comme $P_0 > P_1 > P_2$ alors T_0 commence son exécution. A l'instant t_6 : T_0 prend R_0, on a alors ($\pi'=\pi(R_0)$) et ($\pi(R_0)=P'_0$). Le test de préemption sur T_1 et T_2 n'est pas satisfaisant puisque ($P_1 < P_0$) et ($P_2 < P_0$), donc T_0 continue son exécution. A l'instant t_7 : T_0 libère R_0 et prend R_1, on a alors $\pi' = \pi(R_1) = P'_1$. Le test de préemption sur T_1 et T_2 n'est pas satisfaisant puisque ($P_1 < P_0$) et ($P_2 < P_0$), donc T_0 continue son exécution. A l'instant t_8 : T_0 libère R_1 et continue son exécution dans aucune section critique durant une période. A l'instant t_9 : T_0 termine son exécution, T_1 peut alors s'exécuter. A l'instant t_{10} : T_1 prend R_1 durant deux périodes de suites, on a alors ($\pi'=\pi(R_1)$) et ($\pi(R_1)=P'_1$). Le test de préemption sur T_2 n'est pas satisfaisant puisque $P_2 < P_1$, donc T_1 continue son exécution. A l'instant t_{12} : T_1 libère R_1 et continue son exécution dans aucune section critique durant une période. A l'instant t_{13} : T_1 termine son exécution. T_2 s'exécute pendant deux périodes dans aucune section critique. A l'instant t_{15} : T_2 termine son exécution, T_3 reprend la sienne durant une période dans aucune section critique. A l'instant t_{16} : T_3 termine son exécution.

Conclusion *:*
Temps d'exécution de $T_0 = t_9 - t_4 = 5$ périodes

5.6- L'exemple avec CDP (Ceiling Dynamic Protocol):
La figure 7 suivante montre l'ordre d'exécution des 4 tâches selon le protocole CDP :

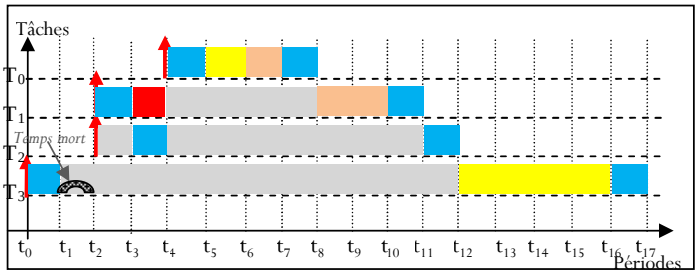


Figure 7 : l'ordonnancement des 4 tâches avec CDP

A l'instant t_0 : T_3 est activée et commence son exécution. A l'instant t_1 : T_3 ne peut pas prendre la ressource R_0 car : $(Nb_{T_{user}(R_0)} > 1)$ et $P_3 < \pi(R_0)$ et $(D_{R_0} = 4 \text{ périodes}) > ([PDM(T_2 \text{ ou } T_3) - t_1] = [2 - 1] = 1 \text{ période})$, selon la règle d'ordonnancement (c) le processeur reste inactif pendant un temps mort qui dure $D_{TM} = [PDM(T_2) - t_1] = t_2 - t_1 = 1 \text{ période}$. A l'instant t_2 : T_1 est activée et commence son exécution et T_2 reste en attente car $P_1 > P_2$. A l'instant t_3 : T_1 ne peut pas prendre la ressource R_1 car : $(Nb_{T_{user}(R_1)} > 1)$ et $P_1 < \pi(R_1)$ et $(D_{R_1} = 2 \text{ périodes}) > ([PDM(T_0) - t_3] = t_4 - t_3 = 1 \text{ période})$, selon la règle d'ordonnancement (b) on a *T_2.état=prête* alors T_2 hérite de π', $P_2 = \pi' = P_1$ et commence son exécution. A l'instant t_4 : T_0 est réveillé et préempte T_2 car $P_0 > P_1$, T_0 commence son exécution et T_2 reprend sa priorité initiale P_2. A l'instant t_5 : T_0 continue à s'exécuter et prend la ressource R_0 et entre en section critique parce que $P_0 = \pi(R_0)$. A l'instant t_6 : T_0 quitte R_0 ($\pi(R_0) = P_3$) et prend R_1 pendant deux périodes car $P_0 = \pi(R_1)$. A l'instant t_7 : T_0 quitte R_1 ($\pi(R_1) = P_1$) et continue son exécution durant une période dans aucune section critique. A l'instant t_8 : T_0 termine son exécution. T_1 reprend la sienne et prend la ressource R_1 pendant deux périodes successives car $P_1 = \pi(R_1)$. A l'instant t_{10} : T_1 quitte R_1 ($\pi(R_1) = \varphi$) et continue son exécution dans aucune section critique durant une période. A l'instant t_{11} : T_2 reprend son exécution dans aucune section critique durant une période car $P_2 > P_3$. A l'instant t_{12} : T_2 termine son exécution et T_3 peut reprendre la sienne et prend R_0 car $P_3 = \pi(R_0)$. A l'instant t_{16} : T_3 quitte R_0 ($\pi(R_0) = \varphi$) et continue son exécution dans aucune section critique. A l'instant t_{17} : T_3 termine son exécution.

Conclusion :

Temps d'exécution de $T_0 = t_8 - t_1 = 4$ périodes.

On constate que la tâche la plus prioritaire T_0 est exécutée dans 4 périodes comme indique leur séquence d'exécution. Donc, T_0 n'était pas bloquée et termine son exécution dans les normes.

6. Comparaison des résultats

Le but du tableau 4 est de comparer les résultats de l'application des quatre protocoles d'inversion de priorités sur l'exemple proposé.

	PIP	**PCP**	**SRP**	**CDP (proposé)**
Types de blocage	héritage de priorité	héritage de priorité plafond de priorité	Plafond de préemption	Section critique longue
Nombre de blocage de la tâche prioritaire	2	1	1	0
Nombre de blocage des tâches moyennes	1	1	1	1
Gaspillage de temps CPU	Non	Non	Non	Oui
Temps de blocage de la tâche prioritaire	4 périodes	OCPP : 2 périodes ICPP : 1 période	1 période	0 période
Interblocage	Non	Non	Non	Non
Chaîne de blocage	Oui	Non	Non	Non
Priorité	Fixe	Fixe	Fixe/dynamique	Fixe
Temps d'exécution de la tâche prioritaire	8 périodes	OCPP : 6 périodes ICPP : 5 périodes	5 périodes	4 périodes
Temps total d'exécution	16 périodes	16 périodes	16 périodes	17 périodes

Tableau 4 : *présentation d'une étude comparative des résultats obtenus par l'exemple proposé*

7. Conclusions & Perspectives

Les différentes techniques de résolution du problème d'inversion de priorité (PIP, PCP, SRP) atteignent certain seuil de réussite dans la réduction de ce problème. De même les risques de la situation d'inter-blocage et des chaines de blocage ont été éliminés notamment par le SRP. Mais, malgré tout ça le phénomène d'inversion de priorité reste un cauchemar pour quelques systèmes temps réel et particulièrement les systèmes temps réel à contrainte dure. Le protocole d'ordonnancement proposé « CDP» apporte plusieurs améliorations par rapport aux anciens protocoles. Il minimise essentiellement la production du phénomène d'inversion de priorité, lorsque la tâche la plus prioritaire commence son exécution, elle ne sera jamais bloquée sur une ressource ou interrompue par une tâche de moindre priorité. Ce protocole peut fournir des résultats intéressants, notamment sur des tâches de sections critiques courtes. Mais, le gaspillage du temps CPU dû à ce protocole implique un temps d'exécution total trop élevé et les sections critiques de longue durée d'exécution provoquent souvent des cas de blocage.

En perspective, nous envisageons de contrer le phénomène d'inversion de priorité en évitant d'exploiter les temps mort.

RÉFÉRENCES

[1] L. Sha, R. Rajkumar, J. P. Lehoczhy, « *Priority inheritance protocols : an approach to real-time synchronization* », IEEE Transactions on Computers, Vol. n°39, n°9, sept, 1990.

[2] Samia Bouzefrane, « *Étude temporelle des Applications Temps Réel Distribuées à Contraintes Strictes basée sur une Analyse d'Ordonnançabilité* », Thèse de Doctorat, LISI, ENSMA, 1998.

[3] S. Ramamurthy, « *A lock-free approach to object sharing in real time systems* », University of North Carolina, USA, 1997.

[4] T.P. Baker, « *Stack-Based Scheduling of Realtime Processes* », the Journal of Real-Time Systems, 3, pp. 67-99 (1991).

[5] M.I. Chen, K.J. Lin, « *Dynamic Priority Ceilings* », « *A Concurrency Control Protocol for Real-Time Systems* », The Journal of Real-Time Systems, 2, pp. 325-346 (1990).

Problème d'Assemblage Orthogonal Rectangulaire, Approche Algorithmique

Rachid Ouafi[1] et Isma Dahmani[1]

1Département de Recherche Opérationnelle
Faculté de Mathématiques, BP 32
EL Alia Bab Ezzouar, 16111, Alger
ALGERIE.
dahmani.isma@gmail.com
rachid_ouafi@hotmail.com

RÉSUMÉ. Le problème traité dans cet article est le problème d'assemblage orthogonal rectangulaire. Il consiste à regrouper un ensemble de pièces rectangulaires dans un rectangle final de surface minimale. Dans cette étude, une contrainte est imposée, à savoir l'orientation fixe des pièces. Nous proposons dans cet article, une version modifiée de la méthode exacte BFBB (Best First Branch and Bound) [8]. En vue d'améliorer la méthode en temps d'exécution et en place mémoire. A cet effet nous utilisons trois stratégies. Nous commençons par le développement d'une nouvelle borne supérieure afin de réduire l'espace de recherche. Ensuite, nous introduisons une nouvelle représentation [2] de la liste fermée de l'algorithme pour une meilleure réorganisation. Enfin, la notion d'ordre lexicographique est utilisée afin d'éliminer les modèles de découpe dupliqués[9].La performance de notre algorithme a été évaluée sur un ensemble d'instances générées aléatoirement et les résultats obtenus sont comparés à l'algorithme BFBB.

Mots-clés: Séparation et évaluation, optimisation combinatoire, assemblage rectangulaire.

1 Introduction

Les problèmes de Découpe et d'Assemblage (D&A) ont été posés dès les années 60 (Kantorovich, 1960 ; Dyckhoff, 1990; Hifi. 1994, G. Wäscher, H. et al., 2004). De leur intérêt, plusieurs axes de recherche ont suscités l'attention de beaucoup de chercheurs et divers papiers ont fait leur apparition. Le problème de placement (découpe) consiste a placer (découper) un ensemble de pièces dans un rectangle de largeur et hauteur fixes (***packing problem***) ou largeur fixe et de hauteur non fixe (***strip packing***, Hinxman, 1980; Coffman et Lagarias, 1989; Jacobs, 1996 ; Zhang, Kang , Deng 2006 ; Cui, Y. et al., 2008). Les problèmes SP et RP (***Square Packing*** and ***Rectangular Packing***) ont été beaucoup étudiés par plusieurs chercheurs (Kleitman et Krieger, 1975; Duijvestijn, 1978; Hifi, 1994, Hifi et Ouafi 1998 ; Clautiaux F. et al., 2006). Le SP (RP) consiste à regrouper des pièces de formes carrées (rectangulaires) dans un carré (rectangle) final de surface minimale.

Nous nous intéressons dans cet article, au problème d'assemblage orthogonal rectangulaire (PAOR). Le PAOR consiste à regrouper des pièces rectangulaires dans un rectangle final de surface minimale. Son dual[12] est un problème de découpe de stock à 2D (Two-Dimensional Cutting Stock Problem(TDCS)). Une contrainte guillotine est imposée au problème TDCS afin de réduire les modèles de découpes faisables. La découpe guillotine consiste à couper un rectangle d'un bord à l'autre. De même cette contrainte apparait dans le cas du PAOR sous forme de construction orthogonale [8]. Elle se fait en combinant les pièces avec leurs copies sous forme de constructions horizontales et verticales (voir figure 1).

Hifi & Ouafi[8] ont proposé une méthode exacte pour la résolution du PAOR appelée Best First Branch and Bound (BFBB). Nous proposons une version modifiée de BFBB basée sur l'introduction de nouvelles techniques permettant d'améliorer la performance de celle ci.

Notre papier est structuré comme suit : dans la deuxième section nous commençons par la présentation du PAOR [8]. Ensuite, dans la troisième section nous développons la nouvelle version de BFBB par : (i) développement d'une nouvelle borne supérieure (heuristique gloutonne), (ii) adaptation de la nouvelle représentation de la liste fermée [2]. (iii) introduction de la stratégie d'élimination des modèles dominés et dupliqués [9].

Enfin, dans la quatrième section, la performance de la nouvelle version est évaluée sur des instances de différentes tailles générées aléatoirement.

2 Problème d'Assemblage Orthogonal Rectangulaire (*PAOR*)

2.1 Présentation du PAOR

Étant donné un ensemble de n pièces $S=\{(l_1, w_1), (l_2, w_2),\ldots, (l_n, w_n)\}$, où l_i (resp. w_i) est la longueur (resp. la largeur) et b_i désigne le nombre de copies de la pièce i, $i=1,..,n$.

Une solution réalisable T est un modèle d'assemblage et elle est représentée par un rectangle qui regroupe l'ensemble des pièces de S avec leurs duplications.

Une solution optimale T^* est une solution réalisable de surface minimale.

Nous utilisons une stratégie de construction orthogonale pour réduire les modèles d'assemblage faisables. Elle se fait en combinant les pièces avec leurs copies sous forme de constructions horizontales et verticales. A cet effet, les définitions ci-dessous sont adoptées.

Définition 1. Soient p et p' deux pièces de S de dimensions respectives (l, w) et (l',w'). On appelle $R= (l+l', \max\{w, w'\})$ (resp. $R= (\max\{l,l'\},w+w')$), un assemblage rectangulaire, noté r-assemblage, obtenu en joignant p et p' par une construction horizontale (resp. verticale), sachant que le nombre d'occurrences d'une pièce donnée dans chaque construction ne doit pas excéder la borne supérieure b_i (voir

Fig.1. ci-dessous). Considérons S_{rem} sous-rectangle restant, de longueur l_{rem} et de largeur w_{rem}, après la combinaison de deux pièces p et p'.

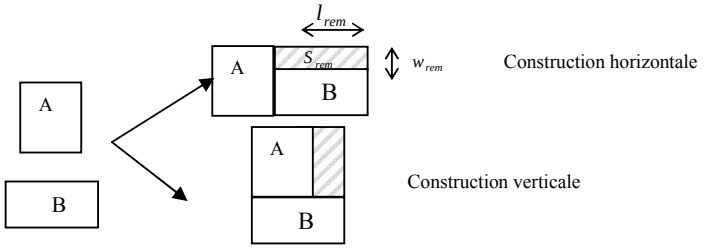


Fig. 1. Constructions horizontale et verticale.

Définition 2. On appelle R un assemblage terminal (noté t-assemblage), s'il contient toutes les pièces avec leurs copies.

Remarque. Dans cette étude, seuls les pièces et les r-assemblages d'orientation fixées sont pris en considération

3 La nouvelle version de l'algorithme BFBB

3.1 La nouvelle borne supérieure

Nous développons un algorithme glouton (Algo BS) pour le calcul de la borne supérieure du PAOR. L'algorithme commence par le calcul de la borne initiale supérieure de départ (Esup[8]), qui sera améliorée par la suite. On considère à cet effet une liste ξ contenant l'ensemble des pièces et leurs copies ordonnées selon l'ordre décroissant de largeur. On construit, en premier lieu, l'assemblage R_{rec} constitué uniquement de pièces de largeur maximale. Dans l'étape principale de la procédure, on effectue des constructions horizontales successives de la manière suivante : on introduit une liste intermédiaire ξ' constituée de pièce de largeur maximale parmi les pièces restantes, puis on sélectionne R de ξ' de longueur maximale ; pour la placer à droite de R_{rec} et on effectue par la suite un remplissage avec les pièces restantes sur le sous-rectangle S_{rem}, tant que possible. La procédure s'arrête si $E_{rec} \leq E_{sup}$ tel que E_{rec} représente la surface de R_{rec} et on affecte la valeur de E_{rec} à E_{sup}. Dans ce cas, E_{sup} est la meilleure borne supérieure de départ par construction horizontale. Sinon cette étape est répétée jusqu'à ce que $\xi=\varnothing$.

Algo BS : Une procédure gloutonne pour le PAOR

Entrée: ensemble des pièces rectangulaires borné par bi, $1 \leq i \leq n$

Sortie: solution sous-optimale notée E_{sup}

L'étape initiale:

1) Calculer E_{sup} tel que : $E_{sup}= \min\{E_h, E_v\}$ où

$E_h = (\sum_{i=1}^{n} b_i l_i)(\max_{1\le i\le n}\{w_i\})$ et $E_v = (\sum_{i=1}^{n} b_i w_i)(\max_{1\le i\le n}\{l_i\})$

2) ξ: ensemble des pièces avec leurs copies triées selon l'ordre décroissant de leurs largeurs : $w1>w2\geq\ldots\geq w_{\sum_{i=1}^{n} b_i}$

Soit R_{rec} le rectangle initial obtenu par construction horizontale de la pièce R_1 avec ses copies $R_{rec}=(b_1 l_1, w_1)$, *avec :* $\xi=\xi \setminus \{R_1 / l_{R_k} = l_{R_1}$ et $w_{R_k} = w_{R_1}\}$

L'étape principale:

Répéter

1. Soit ξ' l'ensemble des pièces dont la largeur est la plus grande :

$$\xi' = \{R_k : w_{R_k} = \max_{R_j \in \xi}\{w_{R_k}\}, \forall\, l_{R_k}\}$$

2. Choisir la pièce: $R' \in \xi' / l_{R'} = \max_{R_k \in \xi'}\{l_{R_k}\}$,
3. faire la construction horizontale de R' avec R_{rec} puis, Poser : $\xi=\xi\setminus\{R'\}$
4. Calculer l'espace vide S_{rem}, résultat de la construction précédente, tel que: $EV=(l_{R'}, w_{R_{rec}} - w_{R'})$.

   Tant que (EV :$(l_{EV}, w_{EV})\neq(0,0)$) et (il existe une pièce qui rentre dans S_{rem})

   Faire

   i. Placer la pièce R_i dans l'espace S_{rem}

   ii. Poser $\xi = \xi\setminus\{R_i\}$

   iii. Calculer le nouvel espace vide S_{rem}

   Fait;

Jusqu'à: ($\xi = \varnothing$) ou ($E_{rec}\leq E_{sup}$);

L'étape terminale:

Si $E_{rec}<E_{sup}$ alors l'ensemble $E_{sup} = E_{rec}$.

Fin.

3.2. Nouvelle représentation de la liste fermée

L'algorithme recherche la plus petite surface qui contient toutes les pièces avec leurs copies. Il est basé sur deux listes principales ξ_2 fermée et $\xi_1$ouverte. Ces dernières occupent une place mémoire très importante au cours de l'exécution. La nouvelle représentation de la liste fermée [2] permet de réorganiser la liste fermée d'une façon intelligente pour éviter les calculs inutiles.

Pour se faire, on fixe L_H et W_V d'après la solution initiale réalisable de Algo1[8] telles que :

L_H : La longueur du rectangle solution, obtenue par Algo1[8] en utilisant la construction horizontale,

W_V: La largeur du rectangle solution, obtenue par Algo1[8] en utilisant la construction verticale.

A chaque itération, on sélectionne R de ξ_1, ayant la plus petite borne inférieure f' et on le place dans ξ_2. La liste ξ_3 contient les r-assemblage obtenus par les combinaisons horizontales de R avec tout R' de l'ensemble ϑ_{l_R} et les combinaisons verticales de R avec tout R'' de l'ensemble ϑ_{w_R} où ϑ_{l_R} et ϑ_{w_R} sont des sous-ensembles de ξ_2 tel que :

$$\begin{cases} \vartheta_{l_R} = \{q \,/\, l_q = l_R + l_p \le L_H,\ p \in \xi_2\} \\ \vartheta_{l_w} = \{q/\, w_q = w_R + w_P \le W_V,\ p \in \xi_2\} \\ \text{où } l_p \text{ et } w_p \text{ sont la longueur et la largeur de l'élément } p. \end{cases}$$

L'algorithme s'arrête dès que $f'(R)$ de R sélectionné de ξ_2 est supérieure ou égale à la meilleure valeur de la solution courante. Sinon, ξ_1 est réduite à l'ensemble vide.

3.3. Notion de modèle dominé et d'ordre lexicographique

3.3.1. Notion de modèle dominé

La notion de modèle dominé [9] est adaptée à BFBB pour stériliser des r-assemblage comme suit :

Soient R et R' deux r-assemblages. La construction orthogonale entre R et R' est dite modèle dominé s'il existe $R'' \in \xi_1$ qui occupe l'espace vide S_{rem} obtenu par la construction orthogonale entre R et R'.

Cette technique est introduite dans BFBB de façon qu'à chaque nouveau modèle produit par combinaison orthogonale, on calcule l'espace vide. S'il existe une pièce de ξ_1 qui peut occuper cet espace sans dépasser les contraintes exigées, alors ce modèle est un modèle dominé à éliminer

3.3.2. Ordre lexicographique

L'ordre lexicographique [9] est appliqué à BFBB, afin d'éliminer les r-assemblages dupliqués modèles de symétrie. Il est introduit dans ξ_1 initiale, tel qu'à chaque R_i ($i \in I$), on associe deux ordres lexicographiques, un horizontal et un vertical, où $i=\theta_h=\theta_v=1,2,\ldots, n$.

Pour chaque r-assemblage A, obtenu par construction horizontale entre K et Q, on introduit un nouvel ordre lexicographique tel que $\theta_{h(A)} = \min\{\theta_{h(K)}, \theta_{h(Q)}\}$ et $\theta = \theta_{v(A)} = \max_{E \in \xi_1 \cup \xi_2} (\theta_{v(E)})+1$

Test du r-assemblage dupliqué. Soient R et R' deux r-assemblages, $\theta_{h(R)}$ et $\theta_{h(R')}$ les ordres lexicographiques horizontaux de R et R' respectivement et $\theta_{v(R)}$ et $\theta_{v(R')}$ les ordres lexicographiques verticaux de R et R' respectivement.

Si $R \in$ liste ouverte composée d'au moins deux pièces et $R' \in$ liste fermée composée de pièces du même type, alors :

- La construction horizontale entre R et R' est un modèle dupliqué si : $\theta_{h(R)} < \theta_{h(R')}$
- La construction verticale entre R et R' est un modèle dupliqué si : $\theta_{v(R)} < \theta_{v(R')}$

Algorithme exact BFBB amélioré (BFBBA) :

ξ_1 : Ensemble des sous problèmes ;

ξ_2 : Liste de mémorisation des meilleurs sous problèmes ;

R, R' et R'' : r-assemblages ;

f'(R) : Borne inférieure du sous problème contenant R ;

Opt : Valeur de la meilleure solution actuelle ;

S_{rem} : Espace vide obtenu par la construction horizontale ou verticale entre R et R ;

$\theta_{h(R)}$: Ordre lexicographique horizontal de R ;

$\theta_{v(R)}$: Ordre lexicographique vertical de R.

Entrée: Une instance du problème d'assemblage rectangulaire orthogonal

Sortie: La valeur de la solution optimale Opt

L'étape initiale:

$\xi_1 = \{R_1, \ldots, R_n\}$; $\xi_2 = \varnothing$ et fin = faux ;

Soit E_{sup} la borne supérieure obtenue par l'application de Algo BS ; Opt = E_{sup} ;

L'étape principale:

Répéter

Choisir le r-assemblage R avec la plus petite valeur de f' ; (notée f'_{min})

Si Opt- $f'_{min} \leq 0$; alors fin = vrai

Sinon

Début

1. Transférer R de ξ_1 à ξ_2
2. construire ξ_3 tel que :
   a. Chaque R'' de ξ_3 obtenu par la construction horizontale entre R et les éléments ϑ_{l_R} de ξ_2 tel que $\vartheta_{l_R} = \{q / l_R + l_p \leq L_H, p \in \xi_2\}$
   b. Chaque R'' de ξ_3 obtenu par la construction verticale entre R et les éléments ϑ_{w_R} de ξ_2 tel que $\vartheta_{l_w} = \{q / w_q = w_R + w_p \leq W_V, p \in \xi_2\}$
   c. Tester R avec les éléments K de ξ_2. Si le modèle obtenu est un modèle dupliqué, alors le nouveau modèle n'est pas construit.
   d. Chaque élément de ξ_3 satisfaisant les contraintes b_i ($1 \leq i \leq n$) et f'<Opt ;
   e. Chaque modèle de ξ_3 qui satisfait le test de dominance est éliminé.
   f. Chaque élément de ξ_3 est étiqueté par un ordre lexicographique.
3. S'il $\exists$ un t-assemblage $R' \in \xi_3 \backslash f'(R') < Opt$, alors Opt = f'(R'); mettre à jour ξ_1 par $\xi_1 \cup \xi_3$ et remplacer ξ_1 par ξ_1/{assemblage non terminal avec l'évaluation $f' \geq$ Opt};
4. Si $\xi_1 = \varnothing$ alors fin = vrai ;

Fin de si

Jusqu'à fin = vrai ;

Fin.

4. Implémentation & Résultats

l'algorithme proposé a été codé en C et testé sur un ordinateur avec processeur Pentium 4 CPU 3.00 GHz, et 1 GO de RAM. La performance de notre algorithme (ABFBB) est évaluée sur plusieurs instances générées aléatoirement. Nous considérons un groupe de 50 instances, le nombre de pièces utilisées est pris dans l'intervalle [3, 16]. Les dimensions (l_i, w_i) de toutes les pièces sont fixées dans l'intervalle [1, 80], et la borne b_i i = 1, ..., n, est générée aléatoirement dans l'intervalle [1, 9].

Performance de l'algorithme amélioré ABFBB

Table 1. Performances de l'algorithme BFBB amélioré

Borne Supérieure	RM (Opt/BS) 1.21	RM (Opt/NBS) 1.15
BFBB	N. N. M 3638,58	T. M (s) 86.88
BFBBA	N.N. M 2532,24	T. M (s) 13.75

Table 2. Gain de l'algorithme BFBB amélioré

%Gain (BFBBA/BFBB)	N. N. M 87,17	T. M 30,40

RM (Opt/BS) : Rapport Moyen entre Opt et Borne Supérieure (Algo1)[7] ;
RM(Opt/NBS) : Rapport Moyen entre Opt et la Nouvelle Borne Supérieure (Algo BS) ;
N.N.M : Nombre moyen de nœuds (construction) ;
T.M(s) : Temps moyen d'exécution (en second).

Les résultats obtenus dans le tableau 1 indique que la nouvelle borne supérieure AlgoBS est meilleure que la borne supérieure Algo1[7], elle passe de 1,21 à 1,15 en moyenne. Les solutions obtenues de l' Algo BS permettent de réduire encore les constructions inutiles au niveau des nœuds internes.

Les résultats de la table 2 montrent clairement l'amélioration apportée par l'algorithme Best First Branch and Bound Amélioré (ABFBB). Le gain en moyenne sur le temps d'exécution est considérablement augmenté, il est égal à 84,17% et produit une réduction moyenne de 30,40% en terme de nombre de nœuds (construction) générés.

5. Conclusion

Nous avons développé une version améliorée de l'algorithme BFBB proposé par Hifi & Ouafi (1996) pour la résolution du problème d'assemblage orthogonal rectangulaire.
Dans la nouvelle version proposée, nous nous sommes basé sur trois techniques qui consiste en :

1. Développement d'une nouvelle borne supérieure (heuristique gloutonne)
2. Adaptation de la nouvelle représentation de la liste fermée (structure de données) afin d'accélérer la construction de la nouvelle configuration (VD Cung et al, 2000)
3. Introduction des stratégies des modèles dominés et modèles symétriques afin d'éliminer certains modèles et de restreindre la recherche de l'arbre développé.

Les résultats expérimentaux obtenus montrent que l'algorithme ABFBB est meilleur que l'algorithme BFBB en termes de temps d'exécution et nombre total de nœuds explorés.

Références

1. F. Clautiaux, A. Jouglet, J. Carlier, A. Moukrim (2006b), A new constraint programming approach for the orthogonal packing problem, Computers and Operations Research, doi:10.1016/j.cor.2006.05.012.
2. E. G. Coffman, and J. C. Lagarias (1989). Algorithms for packing squares: a probabilistic analysis. SIAM Journal of Computing 18 (1), 166-185.
3. V.D. Cung, M. Hifi and B.L. Cun (2000): Constraind two-dimensional cutting stock problems a best-first branch-and-bound algorithm. International Transactions in Operational Research.7 pp 185-210
4. Y. Cui, Y. Yang, X. Cheng, and P. Song,(2008) "A recursive branch-and-bound algorithm for the rectangular guillotine strip packing problem," Computers & Operations Research, vol. 35, no. 4, pp. 1281–1291.
5. A. J. W. Duijvestijn (1978). Simple perfect squared square of lowest order. Journal of Combinatorial Theory 25 (B), 260-263.
6. H. Dyckhoff (1990): A typology of cutting and packing problems. European Journal of Operational Research 44 145-159.
7. P. Gilmore, and R. Gomory (1965):Multistage cutting problems of two and more dimensions. Ops Res. 13, 94-119
8. M. Hifi (1994). Study of some combinatorial optimization problems: cutting stock, packing and Set covering problems, Ph. D. Thesis, Université de Paris 1 Panthéon-Sorbonne (in French).
9. M. Hifi and R. Ouafi (1998): A best-first branch and bound algorithm for orthogonal rectangular packing problem. International Transactions in Operational Research Vol.5, No.5, pp. 345-356
10. M. Hifi and R. M'hallah (2004): An exact algorithm for constraind two dimensional two-staged cutting problems. Operations Research Vol.53, No.1, pp 140-150

11. A.I. Hinxma. (1980). The tri –loss and assortment problems: a survey. European Journal of Operational Research 5 8-18.
12. S. Jakobs (1996). On genetic algorithm for the packing of polygons. European Journal of Operational Research 88, 165-181.
13. L. K. Kantorovich (1960): Mathematical methods of organizing and planning production. Management Science 6,363-422.
14. D.J. Kleitman and M.K. Krieger (1975): An optimal bound for two dimensional bin packing. Proceedings of the 16 th Annual Symposium on foundations of computer Science, pp. 163-168.
15. P. Sweeney, E. Paternoster (1992). Cutting and Packing problems: a categorized application-oriented research bibliography. Journal of Operation Research Society 43(7), 691-706.
16. G. Wäscher, H. Haußner, H. Schumann, (2004): An Improved Typology of Cutting andPacking Problems, 1st ESICUP Meeting, Lutherstadt Wittenberg, Germany, 18-20 March 2004
17. D. Zhang, Y. Kang, A. Deng (2006), A new heuristic recursive algorithm for the strip rectangular packing Problem, Computers and Operations Research 33(8); 2209-2217

Commande optimale de processus thermiques de grande dimension

Pierre Spiteri[1]

[1] IRIT-ENSEEIHT – 2 rue Camichel – BP 7122 – F-31 071 Toulouse (France)
Pierre.Spiteri@enseeiht.fr

Abstract. La présente étude est consacrée à la détermination de la loi de commande optimale pour la régulation de processus thermiques de grandes dimensions. Différents algorithmes de résolution du problème de commande optimale, tels que les méthodes de relaxation, de plus profonde descente et du gradient conjugué sont présentés et implémentés. Leurs efficacités sont comparées expérimentalement.

Keywords: Commande optimale, principe de Pontryaguin, méthode de relaxation, méthode de plus profonde descente, méthode du gradient conjugué, systèmes dynamiques.

1 Introduction

Les nécessités de commande en ligne de processus dynamiques complexes continus et soumis à des perturbations, exigent parfois la résolution extrêmement rapide de systèmes différentiels avec conditions aux deux bouts, ces systèmes étant issus des conditions d'optimalité de Pontryaguin. La résolution efficace de ces systèmes algébro – différentiels permet ainsi de réagir en temps réel aux perturbations éventuelles agissant sur le processus à contrôler de manière optimale. L'intérêt majeur de cette problématique est donc de réduire de façon sensible les temps de calculs et d'autoriser la conduite en temps réel de processus, de grande dimension, interconnectés entre eux.

Dans le cas de problèmes à horizon fixe et à état final libre, on présente dans cette étude, un algorithme de relaxation pour la conduite en ligne d'un processus thermique dans lequel la commande est soumise à des contraintes de type inégalité. L' algorithme de relaxation considéré se déduit directement des conditions d'optimalité de Pontryaguin. À cause des contraintes sur la commande, ces dernières conditions d'optimalité conduisent à la résolution d'un problème multivoque, ce qui classiquement revient à projeter les valeurs calculées de la commande sur l'ensemble convexe définissant ces contraintes. L'algorithme de relaxation étant de nature itérative, sous certaines conditions, l'application de point fixe associée au problème à résoudre est contractante dans un espace produit de fonctions continues par morceaux, ce qui permet de mettre en évidence un critère de convergence simple. De plus, il convient de préciser que les conditions de convergence sont indépendantes de la longueur de l'horizon dans lequel les équations d'évolution sont définies.

Par ailleurs, les performances de l'algorithme de relaxation sont comparées avec celles de méthodes plus classiques, comme la méthode de plus profonde descente ou celle du gradient conjugué. Les expérimentations numériques montrent clairement que l'algorithme de relaxation est particulièrement intéressant :

- lorsque le coût du contrôle est important,

- lorsque les contraintes sur la commande sont saturées où dans ce cas particulier les performances de la méthode du gradient conjugué s'effondrent.
Le présent papier se décompose en 6 sections. Au paragraphe 2, le problème de commande optimale considéré est présenté dans une forme la plus générale et les équations d'optimalité sont établies. Le paragraphe 3 est consacré à la présentation de la modélisation du processus thermique à contrôler. Le paragraphe suivant est consacré à la description des méthodes de relaxation, de plus profonde descente et celle du gradient conjugué ; un résultat de convergence pour la méthode de relaxation est présenté. Enfin, le dernier paragraphe présente brièvement le résultat des essais numériques, lorsque les équations d'état sont respectivement de dimension 6 et 24 ; en particulier les vitesses de convergence et les temps de calculs des différents algorithmes de résolution sont comparés entre eux. L'annexe présente la définition et les propriétés des M-matrices, notion largement utilisée dans le présent travail.

2 Position du problème et caractérisation de la solution

Soit le système physique dont l'état est décrit par l'équation différentielle ordinaire suivante

$$\begin{cases} \dfrac{dy}{dt} = f(y,u,t),\ \mathrm{u(t)} \in \mathrm{U_{ad}},\ \mathrm{t_0} < t \leq T \\ \mathrm{y(t_0)} = y_0 \end{cases} \qquad (1)$$

où U_{ad} représente l'ensemble de commandes admissibles, ensemble convexe fermé des applications bornées, continues par morceaux sur $[t_0,T]$. On désire conduire le système d'un état initial $y(t_0)$ à un état final y(T) en minimisant la fonction coût suivante

$$J = S(y(T),T) + \int_{t_0}^{T} r(y,u,t).dt \qquad (2)$$

où S(y(T),T) représente le coût terminal. Il s'agit donc d'un problème de commande optimale à horizon fixe et à état final libre. Pour respecter des conditions de régularité, on suppose que les applications f et r sont deux fois continûment dérivables par rapport à y et u. La solution d'un tel problème est classiquement caractérisée en utilisant le principe de Pontryaguin, qui compte tenu des contraintes sur la commande s'écrit ici

$$\begin{cases} \dfrac{dy}{dt} = \nabla_p H = f(y,u,t), & \mathrm{y(t_0)} = y_0 \\ -\dfrac{dp}{dt} = \nabla_y H = \nabla_y r(y,u,t) + \left[\dfrac{\partial f}{\partial y}\right]^T .p(t), & \mathrm{p(T)} = \nabla_{\mathrm{y}} \mathrm{S(y(T),T)} \\ \nabla_u H = \dfrac{\partial r}{\partial u} + \left[\dfrac{\partial f}{\partial u}\right]^T .p(t) + \partial\Psi_{U_{ad}} \ni 0 \end{cases} \qquad (3)$$

où H est l'Hamiltonien défini par

$$H(y(t),p(t),u(t),t) = S(y(T),T) + r(y,u,t) + p^t.f(y,u,t),$$

p est l'état adjoint, $\Psi_{U_{ad}}$ est la fonction indicatrice du convexe et $\partial\Psi_{U_{ad}}$ est le sous – différentiel de la fonction indicatrice du convexe U_{ad}. Rappelons que, classiquement, les conditions d'optimalité précédentes expriment le fait que le Hamiltonien *H(y,p,u,t)* possède un minimum absolu. Notons qu'en l'absence de coût terminal, i.e. si $S(y(T),T) \equiv 0$, alors $p(T)=0$. La prise en compte des contraintes sur la commande conduit à la résolution d'un problème multivoque ; ceci provient de la perturbation de la troisième équation par le sous – différentiel de la fonction indicatrice du convexe U_{ad} application multivoque dont nous rappelons ci – dessous la définition ainsi que les principales propriétés.

Définition 1 : Soit E un espace vectoriel normé et E' le dual topologique de E; soit ϕ une application non – différentiable ; l'ensemble des $\xi \in E'$ noté $\partial\phi(\mu)$ tel que

$$\phi(\mu) - \phi(\sigma) - <\xi, \mu - \sigma> \geq 0, \forall\ \mu \in E, \xi \in \partial\phi(\mu)$$

est le sous différentiel de ϕ en μ et ξ est le sous – gradient.

Remarque : si ϕ est une application différentiable alors $\partial\phi(\mu)$ se réduit à $\{\phi'(\mu)\}$.

Pour fixer les idées, la fonction $\phi = |\mu|$ n'est pas différentiable à l'origine ; le sous – différentiel de cette application est donc $\partial\phi(\mu) \equiv \text{sign}(\mu)$, soit

$$\partial\phi(\mu) = sign(u) = \begin{cases} -1 \text{ si } \mu < 0 \\ [-1,+1] \text{ si } \mu = 0 \\ +1 \text{ si } \mu > 0 \end{cases}$$

On peut énoncer les propriétés suivantes

Propriété fondamentale n°1 : Un élément μ_0 vérifie $\phi(\mu_0) = \min_{\mu \in E} (\phi(\mu))$, s.s.i.

$$0 \in \partial\phi(\mu_0)$$

Preuve : $\mu_0 \in E$ vérifie $\phi(\mu_0) \leq \phi(\mu)$, $\forall\ \mu \in E$, s.s.i. on a $\phi(\mu) \geq \phi(\mu_0) + <0, \mu - \mu_0>$; soit $0 \in \partial\phi(\mu_0)$.

Propriété fondamentale n°2 : $\partial\phi$ est un opérateur monotone multivoque de E dans E'

Preuve : Soit $\xi \in \partial\phi(\mu) \Rightarrow \phi(\sigma) \geq \phi(\mu) + <\xi, \sigma - \mu>$, $\forall\ \sigma \in E$

Soit $\zeta \in \partial\phi(\pi) \Rightarrow \phi(\sigma) \geq \phi(\pi) + <\zeta, \sigma - \pi>$, $\forall\ \sigma \in E$

On considère dans la première inégalité que $\sigma = \pi$ et que dans la seconde inégalité $\sigma = \mu$; on additionne terme à terme d'où $<\xi - \zeta, \mu - \pi > \geq 0$, ce qui établit que $\partial\phi$ est une application monotone.

Conséquence : il en résulte que chercher à minimiser ϕ sur un ensemble convexe $U \subset E$ se ramène à résoudre une équation multivoque $0 \in \mathbf{A}(\mu)$, (avec $\mathbf{A} = \partial(\phi + \Psi_U)$), où Ψ_U est la fonction indicatrice du convexe U, dont on rappelle ci dessous la définition

Définition 2 : Soit U un convexe fermé, E un e.v.n., $U \subset E$; la fonction indicatrice Ψ_U du convexe est alors définie par

$$\Psi_U(z) = \begin{cases} 0 \text{ si } z \in U \\ +\infty \text{ sinon} \end{cases}$$

Propriété : sous - différentiel de Ψ_U

$\partial\Psi_U(\mu) = \{\xi \in E' \mid <\xi, \mu - \sigma> \geq 0, \forall\ \mu \in U\} \Rightarrow \text{Dom}(\partial\Psi_U) = \text{Dom}((\Psi_U) = U$

$\partial\Psi_U(\mu) = \{0\}$, $\forall\ \mu \in U$ et si $\mu \in \partial U$ (frontière de U) $\partial\Psi_U(\mu)$ coïncide avec le cône des normales à U au point μ

Exemple : $U = \{ u \mid u_{Max} \geq u \geq u_{min} \}$; alors

$$\partial\Psi_U(v) = \begin{cases}]-\infty,0] \text{ si } v = u_{\min} \\ 0 \text{ si } u_{Max} \geq v \geq u_{\min} \\ [0,+\infty[\text{ si } v = u_{Max} \\ \varnothing \text{ sinon} \end{cases}$$

application qui est bien monotone.

Remarque : ces rappels étant effectués, la formulation (3) des équations d'optimalité traduit la projection de la commande sur le convexe U_{ad} définissant les contraintes ; précisons que nous cherchons à minimiser $\partial(H+\Psi_{U_{ad}})$, et que, compte tenu de la continuité de H, cela conduit finalement à la résolution de l'équation (3). Par ailleurs, en l'absence de contraintes, il est clair que $\partial\Psi_{U_{ad}} \equiv 0$, et les équations d'optimalité se réduisent aux formules classiques

$$\begin{cases} \dfrac{dy}{dt} = \nabla_p H = f(y,u,t), & y(t_0) = y_0 \\ -\dfrac{dp}{dt} = \nabla_y H = \nabla_y r(y,u,t) + \left[\dfrac{\partial f}{\partial y}\right]^T .p(t), & p(T) = \nabla_y S(y(T),T) \\ \nabla_u H = \dfrac{\partial r}{\partial u} + \left[\dfrac{\partial f}{\partial u}\right]^T .p(t) = 0 & \end{cases} \quad (5)$$

3 Régulation d'un processus thermique

Le problème étudié consiste à amener l'état d'un barreau situé dans une cheminée à une température voisine d'une consigne $z_d \in R^n$ en un temps fini T. Dans la suite $x \in R^n$ représente la température de la cheminée en n points, $u \in R^n$ est l'intensité des courants envoyés dans chacun des n enroulements de chauffage ; pour des raisons d'homogénéité de notation, $z \in R^n$ représente la température du barreau en n points. Le problème revient donc à déterminer la commande u telle qu'au bout du temps T la température du barreau soit uniformément égale à z_d. La modélisation mathématique du problème a été obtenue par linéarisation de l'équation de la chaleur autour d'un point de fonctionnement et identification de paramètre (voir [7]). Soit $y=(z_1, x_1,..,z_i, x_i, ..., z_n, x_n) \in R^{2n}$ le vecteur décrivant l'état du système, défini par l'équation d'état suivante

$$\begin{cases} \dfrac{dy}{dt} = Ay + Bu + f(t), \quad t \in [0,T] \\ y(0) = y_0 \end{cases}$$

où y_0 est l'état initial donné du système, f(t) est un terme source donné, A et B deux matrices, telles que A est l'opposée d'une M – matrice, ce qui garantie la stabilité des solutions y(t) puisque $Re(\lambda(A)) < 0$. On renvoie à l'annexe pour un bref rappel de la notion de M – matrice (voir [11]). L'observation du système est constituée par une partie du vecteur y à savoir le vecteur z ; on pose z = C y, où C est la matrice

d'observation. Par exemple, si n = 3 (four à 3 tranches), la matrice d'observation C est définie par

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

Le critère J(u) à minimiser est la combinaison linéaire de deux critères J_1 et J_2, soit

$$J(u) = \alpha\, J_1(u) + \beta\, J_2(u),$$

le dosage entre les deux composantes du critère étant effectué par les coefficients α et β. Pour J_1 et J_2, on retiendra les expressions suivantes

$J_1(u) = \int_0^T \|z - z_d\|_2^2 dt$, définissant un critère de précision

et

$J_2(u) = \int_0^T \|u - u_d\|_2^2 dt$, correspondant à la minimisation de la consommation d'énergie, le terme u_d correspondant à la commande conduisant asymptotiquement à la température prescrite z_d et donnée par $u_d = -(CA^{-1}B)^{-1} z_d$.

Le problème se résume donc à

$$\begin{cases} \text{Déterminer } u \in U_{ad} \text{ t.q. } J(u) = \min\limits_{v \in U_{ad}} J(v) \\ \text{sous la contrainte} \\ \begin{cases} \dfrac{dy}{dt} = Ay + Bu + f(t),\ t \in [0,T] \\ y(0) = y_0 \end{cases} \end{cases}$$

l'ensemble U_{ad} étant défini par $U_{ad} = \{\, u \mid u_{Max} \geq u \geq u_{min} \,\}$.

Dans le cas général où les contraintes sont prises en compte, les équations d'optimalité s'écrivent

$$\begin{cases} \begin{cases} \dfrac{dy}{dt} = Ay + Bu + f(t),\ t \in [0,T] \\ y(0) = y_0 \end{cases} \\ \begin{cases} -\dfrac{dp}{dt} = A^t p + C^t C y - C^t z_d \\ p(T) = 0 \end{cases} \\ \dfrac{\alpha}{\beta} B^t p + u - u_d + \partial \Psi_{U_{ad}} \ni 0 \end{cases}$$

où $\partial\Psi_{Uad}$ est le sous-différentiel de la fonction indicatrice du convexe U_{ad}; dans le cas sans contrainte elles se réduisent à

$$\begin{cases} \begin{cases} \dfrac{dy}{dt} = Ay + Bu + f(t), \ \ t \in [0,T] \\ y(0) = y_0 \end{cases} \\ \begin{cases} -\dfrac{dp}{dt} = A^t p + C^t C y - C^t z_d, \ t \in [0,T] \\ p(T) = 0 \end{cases} \\ \dfrac{\alpha}{\beta} B^t p + u - u_d = 0, \ t \in [0,T] \end{cases}$$

4 Algorithmes de résolution

4.1 Algorithme de relaxation

La caractérisation précédente des solutions fournit directement un algorithme de résolution du problème de commande optimale du processus thermique. Soit $u^{(0)}(t)$, $t \in [0,T]$, donnée une approximation initiale de la commande. On effectue alors

(a) la détermination de l'état $y^{(r+1)}(t)$ par intégration numérique de l'équation d'état

$$\begin{cases} \dfrac{dy^{(r+1)}(t)}{dt} = Ay^{(r+1)}(t) + Bu^{(r)}(t) + f(t), \ \ t \in [0,T] \\ y^{(r+1)}(0) = y_0 \end{cases}$$

(b) la détermination de l'état adjoint $p^{(r+1)}(t)$ par intégration numérique dans le sens rétrograde de l'équation d'état adjoint, avec $p^{(r+1)}(T) = 0$, en l'absence de coût terminal dans ce problème

$$\begin{cases} -\dfrac{dp^{(r+1)}(t)}{dt} = A^t p^{(r+1)}(t) + C^t C y^{(r+1)}(t) - C^t z_d, \ t \in [0,T] \\ p^{(r+1)}(T) = 0 \end{cases}$$

(c) la détermination de la commande $u^{(r+1)}(t)$, $t \in [0,T]$

$$u^{(r+1)}(t) = \underset{U_{ad}}{\mathrm{Proj}} \ (u_d - \frac{\alpha}{\beta} B^t p^{(r+1)}(t)), \ t \in [0,T].$$

Le processus de calcul (a), (b) et (c) est recommencé jusqu'à convergence. On indique ci dessous un résultat de convergence (voir [7] pour la preuve)

Proposition 3 : Si la matrice A est l'opposée d'une M – matrice et si le rapport $\dfrac{\beta}{\alpha} > k_O$, alors la méthode de relaxation décrite par (a), (b) et (c) est convergente.

Remarque : la convergence de la méthode de relaxation est assurée dés que le poids de la commande est plus important que celui de la précision sur l'observation dans le critère J(u) à minimiser.

À titre indicatif de comparaison de performances, nous donnons la formulation des algorithmes de plus profonde descente et de gradient conjugué.

4.2 Algorithme de plus profonde descente

Comme précédemment soit $u^{(0)}(t)$, $t \in [0,T]$, donnée une approximation initiale de la commande. On effectue alors

(a') la détermination de l'état $y^{(r+1)}(t)$ par intégration numérique de l'équation d'état

$$\begin{cases} \dfrac{dy^{(r+1)}(t)}{dt} = Ay^{(r+1)}(t) + Bu^{(r)}(t) + f(t), \ t \in [0,T] \\ y^{(r+1)}(0) = y_0 \end{cases}$$

(b') la détermination de l'état adjoint $p^{(r+1)}(t)$ par intégration numérique dans le sens rétrograde de l'équation d'état adjoint

$$\begin{cases} -\dfrac{dp^{(r+1)}(t)}{dt} = A^t p^{(r+1)}(t) + C^t C y^{(r+1)}(t) - C^t z_d, \ t \in [0,T] \\ p^{(r+1)}(T) = 0 \end{cases}$$

(c') la détermination de la commande par la méthode de plus profonde descente

$$u^{(r+1)}(t) = \underset{U_{ad}}{\Pr oj}(u^{(r)}(t) - \theta_r G^{(r)}), \ t \in [0,T]$$

où

$$G^{(r)} = u^{(r)} + \frac{\alpha}{\beta} B^t p^{(r+1)}(t) - u_d, \ t \in [0,T]$$

et

$$\theta_r = \frac{\left\| G^{(r)} \right\|^2}{\frac{\alpha}{\beta} < B^t \hat{p}^{(r+1)}, G^{(r)} > + \left\| G^{(r)} \right\|^2}$$

le produit scalaire <. ., ..> étant défini par $< .,. >= \int_0^T << .,. >>_{R^n} dt$, où $<< .,. >>_{R^n}$ est le produit scalaire standard dans R^n, l'intégrale étant évaluée par voie numérique.

4.3 Algorithme du gradient conjugué

Soit $u^{(0)}(t)$, $t \in [0,T]$, donnée une approximation initiale de la commande. La méthode du gradient conjugué se décompose en deux parties : une partie dévolue à l'initialisation de l'algorithme et une partie itérative constituée d'un corps de boucle.

Initialisation de l'algorithme

calcul de $q^{(0)}(t) = \nabla J(u^{(0)}(t)) = \text{grad } J(u^{(0)}(t))$, $t \in [0,T]$ par résolution des équations suivantes

$$\begin{cases} \dfrac{dy^{(0)}(t)}{dt} = Ay^{(0)}(t) + Bu^{(0)}(t) + f(t), \ t \in [0,T] \\ y^{(0)}(0) = y_0 \end{cases}$$

$$\begin{cases} -\dfrac{dp^{(0)}(t)}{dt} = A^t p^{(0)}(t) + C^t C y^{(0)}(t) - C^t z_d, \ \mathrm{t} \in [0,\mathrm{T}] \\ p^{(0)}(T) = 0 \end{cases}$$

puis

$$q^{(0)} = G^{(0)} = u^{(0)} + \frac{\alpha}{\beta} B^t p^{(0)}(t) - u_d, \ \mathrm{t} \in [0,\mathrm{T}]$$

ainsi que

$$\begin{cases} \dfrac{d\hat{y}^{(0)}(t)}{dt} = A\hat{y}^{(0)}(t) + Bq^{(0)}(t), \ \ \mathrm{t} \in [0,\mathrm{T}] \\ \hat{\mathrm{y}}^{(0)}(0) = 0 \end{cases}$$

$$\begin{cases} -\dfrac{d\hat{p}^{(0)}(t)}{dt} = A^t \hat{p}^{(0)}(t) + C^t C \hat{y}^{(0)}(t), \ \mathrm{t} \in [0,\mathrm{T}] \\ \hat{p}^{(0)}(T) = 0 \end{cases}$$

$$\mathsf{A}\, q^{(0)} = q^{(0)} + \frac{\alpha}{\beta} B^t \hat{p}^{(0)}(t), \ \mathrm{t} \in [0,\mathrm{T}]$$

et

$$\theta_0 = \frac{< G^{(0)}, q^{(0)} >}{< \mathsf{A} q^{(0)}, q^{(0)} >}$$

Corps de boucle

(a'') détermination de l'état $y^{(r+1)}(t)$ par intégration numérique de l'équation d'état

$$\begin{cases} \dfrac{dy^{(r+1)}(t)}{dt} = Ay^{(r+1)}(t) + Bu^{(r)}(t) + f(t), \ \ \mathrm{t} \in [0,\mathrm{T}] \\ \mathrm{y}^{(r+1)}(0) = \mathrm{y}_0 \end{cases}$$

(b'') détermination de l'état adjoint $p^{(r+1)}(t)$ par intégration numérique dans le sens rétrograde de l'équation d'état adjoint

$$\begin{cases} -\dfrac{dp^{(r+1)}(t)}{dt} = A^t p^{(r+1)}(t) + C^t C y^{(r+1)}(t) - C^t z_d, \ \mathrm{t} \in [0,\mathrm{T}] \\ p^{(r+1)}(T) = 0 \end{cases}$$

Soit

$$G^{(r)} = u^{(r)} + \frac{\alpha}{\beta} B^t p^{(r+1)}(t) - u_d, \ \mathrm{t} \in [0,\mathrm{T}]$$

et

$$\beta_r = -\frac{\left\| G^{(r)} \right\|^2}{\left\| G^{(r-1)} \right\|^2} \qquad q^{(r)} = G^{(r)} - \beta_r q^{(r-1)}$$

(c'') détermination de $\hat{y}^{(r+1)}$ et $\hat{p}^{(r+1)}$ par intégration de

$$\begin{cases} \dfrac{d\hat{y}^{(r+1)}(t)}{dt} = A\hat{y}^{(r+1)}(t) + Bq^{(r)}(t), \ \ \mathrm{t} \in [0,\mathrm{T}] \\ \hat{\mathrm{y}}^{(r+1)}(0) = 0 \end{cases}$$

$$\begin{cases} -\dfrac{d\hat{p}^{(r+1)}(t)}{dt} = A^t \hat{p}^{(r+1)}(t) + C^t C\hat{y}^{(r+1)}(t), \ \mathrm{t} \in [0,\mathrm{T}] \\ \hat{p}^{(r+1)}(T) = 0 \end{cases}$$

et

$$\mathsf{A}q^{(r)} = q^{(r)} + \frac{\alpha}{\beta} B^t \hat{p}^{(r+1)}(t), \ \mathrm{t} \in [0,\mathrm{T}]$$

Soit

$$\theta_r = \frac{< G^{(r)}, q^{(r)} >}{< \mathsf{A}q^{(r)}, q^{(r)} >}$$

(d'') Détermination de la commande par la méthode du gradient conjugué

$$u^{(r+1)}(t) = \underset{\mathrm{U_{ad}}}{\mathrm{Pr}\, oj} \ (u^{(r)}(t) - \theta_r q^{(r)}), \ \mathrm{t} \in [0,T],$$

le produit scalaire <. ., ..> étant défini comme au paragraphe 4.2.

Remarque : l'implantation des algorithmes précédents, nécessite la résolution d'une équation différentielle ordinaire. Parmi les méthodes d'intégration de ce type d'équation, nous avons choisi d'utiliser la méthode d'Euler – modifiée, d'ordre 2, qui se formule comme suit pour la résolution de l'équation différentielle ordinaire suivante

$$\begin{cases} \dfrac{dy(t)}{dt} = f(y,t), \ \mathrm{t}_0 < t \le T \\ y(\mathrm{t}_0) = \mathrm{y}_0 \end{cases} ;$$

on discrétise le temps en les points $t_m=t_0+m.\delta t$, où δt est le pas de temps et on approxime $y(t_m)$ par la suite récurrente y_m calculée comme suit

$$\begin{cases} K0 = \delta t.f(y_m, t_m) \\ K1 = \delta t.f(y_m + K0, t_m + \delta t) \\ y_{m+1} = y_m + \dfrac{1}{2}(K0 + K1) \end{cases}$$

Ce procédé est particulièrement commode dans le cas de notre application puisqu'il ne nécessite pas d'utilisation de valeurs en des points intermédiaires compris dans l'intervalle $[t_m, t_{m+1}]$.

5 Expérimentations numériques

Les expérimentations numériques ont portés sur la régulation de deux processus thermiques de grande dimension, à savoir

- un four à n = 3 zones de chauffage, baptisé Horace
- un four à n = 12 zones de chauffage, baptisé Hercule.

On trouvera dans [7] les valeurs des coefficients des matrices A, B et C, ainsi que celles des valeurs des paramètres intervenant dans l'identification. On indique simplement que, dans chaque cas, A est une matrice carrée de dimension 2n, et il a été vérifié que c'était bien une M – matrice, ce qui permet d'appliquer le résultat théorique énoncé au paragraphe 4 ; de plus les matrices B et C sont des matrices rectangulaires de dimensions respectives (2nxn) et (nx2n).

On a constaté expérimentalement que pour une valeur du rapport $\frac{\beta}{\alpha} > k_O$, correspondant à un coût élevé du contrôle par rapport à la précision sur l'observation z = Cy, la convergence des méthodes présentées précédemment est bonne quelle que soit la donnée initiale ; on a pu donner une estimation de k_0 pour chacun des processus thermiques. Ainsi $k_0 \approx 2$ pour le processus Horace et $k_0 \approx 4$ pour le processus Hercule. Cependant, si le rapport $\frac{\beta}{\alpha} \leq k_O$, les méthodes itératives précédentes convergent également. Les principaux résultats expérimentaux sont résumés dans les tables 1, 2, 3 et 4 et nous conduisent à formuler les commentaires suivants

- dans le cadre théorique envisagé ici, c'est à dire lorsque le coût du contrôle est plus élevé que le coût de la précision, la méthode de relaxation est plus performante que les deux autres méthodes. Par contre lorsque le rapport $\frac{\beta}{\alpha} \leq k_O$, la méthode du gradient conjugué s'avère efficace ; il faut cependant préciser que si cette dernière méthode s'applique bien dans le cas sans contraintes, elle est nettement moins performante lorsque celles-ci sont saturées. Par contre la méthode de relaxation est moins sensible dans ce cas,
- pour tous les essais numériques effectués, la méthode des directions conjuguées donne de meilleurs résultats que ceux obtenus avec la méthode de plus profonde descente ; néanmoins, ces deux algorithmes ont des performances sensiblement équivalentes,
- la méthode de relaxation est simple à programmer et nécessite peu de place mémoire ; c'est une méthode particulièrement robuste. Ce type de méthode est plus efficace lorsque les interactions entre les diverses composantes sont faibles, ce qui ne semble pas être le cas des méthodes de descente ou de direction conjuguées.

Remarque : on peut envisager une extension de cette étude au cas de problèmes de commande optimale plus généraux, tels que l'étude de systèmes singulièrement perturbés, correspondant à des problèmes à plusieurs échelles de temps, ainsi qu'à des problèmes multi – critères (voir [8]).

Table 1. Temps de calcul en secondes pour la regulation du four Horace.

Rapport $\frac{\beta}{\alpha}$	0,25	0,25 [1]	1	2	4	10
Algorithme						
Méthode de relaxation	317	171, 8	64,5	45,5	39,3	33
Méthode de plus profonde descente	114	163,6	88,7	76,3	63,8	63,8
Méthode de gradient conjugué	77,5	1 828	64,8	64,8	64,8	52,2

[1] Cas de contraintes saturées

Table 2. Nombre d'itérations pour la convergence pour la regulation du four Horace.

Rapport $\frac{\beta}{\alpha}$	0,25	0,25 [2]	1	2	4	10
Algorithme						
Méthode de relaxation	50	27	10	7	6	5
Méthode de plus profonde descente	10	13	7	6	5	5
Méthode de gradient conjugué	6	59	5	5	5	4

Table 3. Temps de calcul en secondes pour la regulation du four Hercule.

Rapport $\frac{\beta}{\alpha}$	3	4	10
Algorithme			
Méthode de relaxation	1009	854	545
Méthode de plus profonde descente	1077	1077	924
Méthode de gradient conjugué	931	931	776

Table 4. Nombre d'itérations pour la convergence pour la regulation du four Hercule.

Rapport $\frac{\beta}{\alpha}$	3	4	10
Algorithme			
Méthode de relaxation	13	11	7
Méthode de plus profonde descente	7	7	6
Méthode de gradient conjugué	6	6	5

6 Annexe : notion de M – matrice

On rappelle ci-dessous des définitions et résultats classiques (voir [11]).

Définition 4 : A est une M – matrice si A est inversible, $A^{-1} \geq 0$ et $a_{i,j} \leq 0$ dès que $i \neq j$.

Théorème 5 : Soit A une matrice t.q. $a_{i,j} \leq 0$ dés que $i \neq j$. A est une M – matrice ssi

(1) les éléments diagonaux $a_{i,i}$ sont strictement positifs

(2) la matrice de Jacobi $J = I - D^{-1} A$, D diagonale de A, est t.q. $\rho(J) < 1$ (rayon spectral de J)

Théorème 6 : Soit A une M – matrice et Δ une matrice diagonale non négative. Alors $(A+\Delta)$ est une M – matrice et $(A+\Delta)^{-1} \leq A^{-1}$

[2] Cas de contraintes saturées

Définition 7 : Une matrice A est diagonale fortement dominante si $\left|a_{i,i}\right| \geq \sum_{j\neq i}\left|a_{i,j}\right|$, $\forall i \in \{1,..,\dim(A)\}$ (A est diagonale dominante) et s'il existe au moins un indice k t.q. $\left|a_{k,k}\right| > \sum_{j\neq k}\left|a_{k,j}\right|$.

Définition 8 : Soit A une matrice réelle ou complexe. A est réductible, s'il existe une matrice de permutation P, de même dimension, telle que

$$P^t A\, P = \begin{pmatrix} B_{1,1} & B_{1,2} \\ 0 & B_{2,2} \end{pmatrix},$$

où les matrices $B_{i,i,}$ pour i=1,2, sont carrées. Si une matrice n'est pas réductible, elle est irréductible.

Lemme 9 : Une C.N.S. pour qu'une matrice *A* soit irréductible est que pour tout couple d'indices (i,j), $i \neq j$, il existe au moins un ensemble d'indices $i_1, i_2, \ldots i_k$ ($i_k = j$), avec $k \geq 1$, tels que les éléments $a_{i,i1}, a_{i1,i2}, \ldots.., a_{ik\text{-}1,j}$ soient tous différents de zéro.

Définition 10 : Une matrice A est diagonale dominante irréductible si A est irréductible et diagonale fortement dominante.

Théorème 11 : Une matrice strictement diagonale dominante ou diagonale dominante irréductible est inversible.

Théorème 12 : Soit A une matrice strictement ou irréductible diagonale dominante t.q. $a_{i,j} \leq 0$ dès que $i \neq j$ et $a_{i,i} > 0$. Alors A est une M – matrice.

Théorème 13 : Si A est une M – matrice alors la partie réelle de ses valeurs propres est positive.

Références

1. Athans, M., Falb, M.L. : Optimal Control. Mc Graw Hill (1966)
2. Bernhard, P. : Commande optimale, décentralisation et jeux dynamiques. Dunod (1976)
3. Ciarlet, P.G. : Introduction à l'analyse numérique matricielle et à l'optimisation. Collection Mathématiques appliquées pour la maîtrise. Masson (1982)
4. Faure, P. : Analyse numérique et notes d'optimisation. Ellipses (1988)
5. Feldbaum, A.: Principes théoriques des systèmes asservis optimaux. Edition MIR de Moscou (1973)
6. Gien, D., Lang, B., Miellou, J.C., Raffort, L., Spiteri, P., : Commande optimale de systèmes complexes, RAIRO, 18 (2), 209-224 (1984)
7. Lang, B., Miellou, J.C., Spiteri, P., : Asynchronous relaxation algorithms for optimal control problem, Mathematics and Computers in simulations, 28, 227-242 (1986)
8. Lang, B., Spiteri, P., : Decomposition and coordination using asynchronous iterations in optimal control, Encyclopedia of Systems and Control, Ed. Singh, 3475-3481 (1987)
9. Lascaux, P., Theodor, R.: Analyse numérique matricielle appliquée à l'art de l'ingénieur. Masson (1987)
10. Naslin, P.: Théorie de la commande et conduite optimale. Dunod (1969)
11. Ortega, J.M., Rheinboldt, W.C.: Iterative solution of nonlinear equations in several variables. Academic Press (1970)
12. Pontryaguin, L., Boltianski, V., Gamkrelidze, R, Michtchenk, E. : Théorie Mathématiques des processus optimaux. Edition MIR de Moscou (1974)
13. Zoubov, V. : Théorie de la commande. Edition MIR de Moscou (1978)

Réseaux de Neurones Récurrents Appliqués à l'Automatisation du Marché à Terme : cas Producteur-Consommateur

Salima KENDI*, Fodil LAIB**, and Mohammed Said RADJEF***

Laboratoire de Modélisation et d'Optimisation
des Systèmes (LAMOS), Université de Béjaia, Algérie.

salima_kendi@yahoo.fr,fodil.laib@cevital.com,radjefms@yahoo.fr

Résumé Dans ce travail, nous proposons une approche d'automatisation du processus de négociation du prix sur un marché à terme simplifié, en occurrence le marché producteur-consommateur. Cette approche consiste à représenter le système par un réseau de neurones récurrent capable de réagir au flux de prévisions de l'offre et de la demande dans l'émission des ordres d'achat et de vente, l'interaction de ces ordres aboutira à la génération d'une courbe de Prix du Marché (PM). Pour mesurer la performance de notre système de négociation automatique, nous avons d'abord émis des hypothèses sur les propriétés d'une courbe de Prix de Référence (PR), puis nous proposons des mesures analytiques permettant de calculer la distance entre les deux courbes (PM) et (PR). L'objectif du réseau de neurones est d'apprendre à générer une courbe (PM) qui soit la plus proche possible de la courbe (PR).

Mots clés : Négociation automatique, Marchés à terme, Réseaux de Neurones Récurrents, Apprentissage, Stratégies de négociation.

Introduction

La majorité des crashs financiers et des bulles spéculatives est due au comportement irrationnel de l'homme (comportement moutonnier, la panique, le désir du gain et la peur de perdre). Il est probable qu'en remplaçant l'homme par un automate dans la fixation des prix, il y'aurait moins de crashs financiers car les automates poursuivent des stratégies rationnelles tenant compte de la situation réelle du marché. Cette rationalité peut être un facteur de stabilité économique si ces automates sont capables de générer un prix accomplissant son rôle de régulateur de l'offre et de la demande.

Avant l'âge des ordinateurs, les négociants échangeaient des actions et des matières premières en s'appuyant sur l'intuition pour la fixation du prix. Avec l'augmentation du niveau de l'investissement et du commerce, les négociants recherchaient des outils augmentant leurs gains et minimisant leurs risques. L'analyse technique et fondamentale, les statistiques et la régression linéaire sont autant d'outils employés pour prévoir la direction du marché. Aucune de ces techniques ne s'est avérée être l'outil uniformément correct de prévision ; beaucoup d'analystes argumentent sur l'utilité de beaucoup d'approches [1].

L'analyse technique repose sur l'hypothèse que l'histoire se répète et que la direction du prix peut être déterminée en examinant les prix passés. Cette technique est employée par la majorité des opérateurs en bourse [2].

L'analyse fondamentale nécessite une description des facteurs explicatifs de l'offre et de la

* Département de recherche Opérationnelle, Université de Béjaia, Algérie (corresponding author).
** Groupe CEVITAL, Garidi II, Kouba, Alger, Algérie.
*** Laboratoire de Modélisation et d'Optimisation des Systèmes (LAMOS), Université de Béjaia, Algérie.

demande d'un produit sur une période donnée. L'ensemble des relations de causalité entre les variables constitue un modèle économique qui est formalisé en un modèle économétrique [3].

Dans la grande variété des techniques de modélisation des marchés, chaque approche a son propre ensemble de défenseurs et de détracteurs. Le but commun de toutes ces méthodes est de prévoir les mouvements de prix à partir de l'information passée. Ces méthodes fonctionnent mieux lorsqu'elles sont utilisées simultanément. L'avantage principal d'employer un réseau de neurones demeure dans le fait qu'il permet d'apprendre à employer efficacement ces méthodes en association.

Différentes études empiriques ont testé l'efficience des marchés à terme en montrant que les prix à terme sont des prévisions non biaisées des prix au comptant. Les premières ont été réalisées en 1974 sur les marchés à terme de bétail sur pied et par Bigman, Goldfarb et Schechtman (1983) [4] sur les marchés à terme de grains à Chicago. Les deux études concluent à l'inefficience des marchés à terme.

La capacité des réseaux de neurones de traiter avec l'incertain et de découvrir des rapports non-linéaires dans des données d'entrée les rend plus aptes à modéliser les systèmes dynamiques non-linéaires tels que le marché boursier [1]. Selon Wong, Bodnovich et Selvi [5], les domaines d'application les plus fréquents des réseaux de neurones durant les dix dernières années sont les opérations de production (53,5%) et les finances (25,4%).

Il y a une littérature étendue sur les applications financières des réseaux de neurones (Trippi et Turban, 1993 [6] ; Refenes, 1994 [7] ; Odom et Sharda (1990) [8] ; Coleman, Graettinger et Lawrence (1991) [9] ; Salchenkerger, Cinar et Lash (1992) [10] ; Tam et Kiang (1992) [11] ; Wilson et Sharda (1994) [12] ; Weigend, Rumelhart et Hubermann (1991) [13] ; ...).

1 Définition du contrat à terme

Un contrat à terme est une promesse de vente d'un produit à une date ultérieure et pour un prix immédiatement fixé. Sur un marché de contrats à terme, l'opérateur achète ou vend des contrats sans nécessairement posséder la marchandise servant de support aux contrats. Un opérateur peut s'engager en mars à livrer pour le mois de septembre une marchandise qu'il ne possède pas. De la même manière, l'opérateur s'engageant à acheter sans être assuré que son co-contractant possède les produits physiques ou sans avoir lui même la volonté d'en prendre réellement livraison. Le marché à terme est donc avant tout un *marché financier* [14].

2 Formulation mathématique du mécanisme du marché à terme

Soit $\mathcal{N} = \{1, \ldots, n\}$ un ensemble de traders dans un marché à terme. Ces derniers estiment les niveaux de l'offre et de la demande à travers les informations provenant de différentes sources. Les traders envoient leurs ordres $\mathbf{u}_i$, $i \in \mathcal{N}$, sur la plate-forme du marché comme montré sur la figure 1. Les ordres de ventes sont mis dans la liste des ordres de vente, en occurrence LSO[1], et ceux d'achat dans la liste des ordres d'achat, LBO[2]. Le meilleur ordre de vente (celui ayant le prix le plus bas) est toujours à la tête de LSO et le meilleur ordre d'achat (celui ayant le prix le plus haut) est toujours à la tête de LBO. Une session de contrats à terme se déroule dans un intervalle de temps $[0,\ T]$ réparti en un ensemble de périodes $\mathbb{T} = \{t_0,\ \ldots,\ t_m\}$, tels que

$$t_0 = 0,\ t_m = T,\ t_j = t_{j-1} + h,\quad j = 1,\ \ldots,\ m,$$

où h est le pas de la discrétisation : $h = \frac{T}{m}$ [15].

[1] List of Selling Orders.
[2] List of Buying Orders.

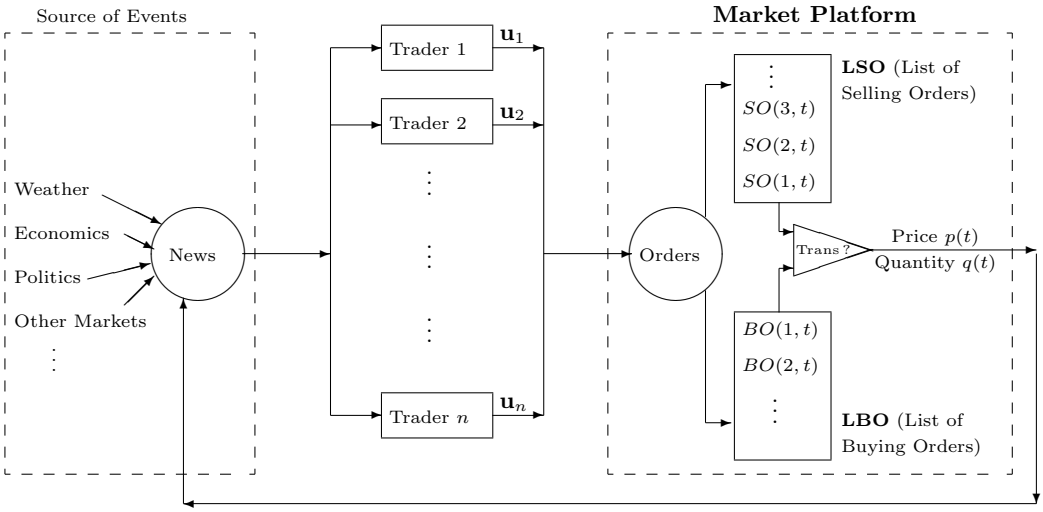


Fig. 1. Traders envoyant leurs ordres vers la plate-forme du marché

3 Définitions et hypothèses relatives à la courbe du prix

Il est généralement admis que la balance de l'offre et de la demande est le conducteur principal du prix. Afin de pouvoir évaluer la performance d'une stratégie de négociation, nous présentons ci-dessous quelques hypothèses sur les propriétés que devrait avoir une courbe de prix idéal.

Definition 1. ***(Prix nominal).*** *Pour produire une unité d'un produit donné C, nous avons besoin d'utiliser c_k unités du composant C_k coûtant $p_k(t_{j-1})$ à l'instant t_{j-1}, avec $k = 1, \ldots, K$ et $j = 1, \ldots, m$. Ainsi, en incluant un ratio de marge bénéficiaire r, le* prix nominal *du produit C à l'instant t_j est*

$$p^*(t_j) = (1 + r) \sum_{k=1}^{K} c_k p_k(t_{j-1}).$$

Hypothese 1. ***(S&D[3] détermine la tendance du prix actuel).*** *La courbe du prix doit inversement suivre la courbe de la balance (S&D) de l'offre S et de la demande D. Si (S&D) décline alors le prix doit augmenter et vice-versa.*

Hypothese 2. ***(Le prix conduit le prochain mouvement de** $S\&D$**).*** *Une augmentation significative du prix encourage l'investissement , ce qui provoquera une augmentation du niveau de l'offre, au même temps la consommation sera réduite. La réciproque est vraie.*

Hypothese 3. ***(Le prix nominal détermine le niveau actuel du prix).*** *Dans le cas d'un surplus, le prix de marché devrait baisser au-dessous du prix nominal dans le but de décourager la production et encourager la consommation. Inversement, dans le cas d'un déficit, le prix de marché devrait être au-dessus du prix nominal dans le but d'encourager la production et décourager la consommation. Si à n'importe quel instant t_j, le niveau de l'offre est égal au niveau de la demande, alors le prix du marché $p(t_j)$, à l'instant t_j devrait être égal au prix nominal, $p^*(t_j)$.*

[3] S&D : abréviation de Supply and Demand (offre et demande).

Hypothese 4. ***(La volatilité de S&D transférée au prix).*** *La volatilité de S&D devrait induire une volatilité équivalente sur la courbe du prix. Explicitement, dans tout sous-ensemble de temps* $\{t_k,\ \ldots,\ t_{k+h}\} \subset \{t_1,\ \ldots,\ t_m\}$*, la relation suivante est satisfaite :*

$$\sigma_{S\&D}(t_k,\ t_{k+h}) \simeq \sigma_p(t_k,\ t_{k+h}),$$

où $\sigma_{S\&D}(t_k,\ t_{k+h})$ *et* $\sigma_p(t_k,\ t_{k+h})$ *sont les écart-types de S&D et du prix respectivement sur la période* $(t_k,\ t_{k+h})$*, avec* $h \in \mathbb{N}$*,* $t_1 \leq t_k < t_{k+h} \leq t_m$*.*

Hypothese 5. *Une bonne courbe de prix est celle où les transactions prennent lieu dans la majorité des périodes, c'est-à-dire la quantité transactionnelle* $q(t_j)$ *devrait être positive la plupart du temps,* $j = 1,\ \ldots,\ m$*.*

Hypothese 6. ***(Volumes homogènes).*** *La volatilité des quantités transactionnelle* $q(t_j)$ *devrait être maintenue au minimum dans tout sous-ensemble de temps* $\{t_k,\ \ldots,\ t_{k+h}\} \subset \{t_1,\ \ldots,\ t_m\}$*,*

$$\sigma_q(t_k,\ t_{k+h}) \simeq 0,$$

où $\sigma_q(t_k,\ t_{k+h})$ *est l'écart-type de la variable* q *sur la période* $(t_k,\ t_{k+h})$*, avec* $h \in \mathbb{N}$*,* $t_1 \leq t_k < t_{k+h} \leq t_m$*.*

3.1 Critères de performance économique

Pour mesurer l'efficacité du réseau de neurones (RPQ) suivant les hypothèses 1-6 respectivement, les critères suivants sont utilisés :

$$z_1(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{[sign(G(t_j)-G(t_{j-1}))=-sign(p(t_j)-p(t_{j-1}))]}, \tag{1}$$

$$z_2(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m-1}\sum_{j=1}^{m-1} \mathbf{1}_{[signe(G(t_{j+1})-G(t_j))=signe(p(t_j)-p(t_{j-1}))]}, \tag{2}$$

$$z_3(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{[sign(G(t_j))=-sign(p(t_j)-p^*(t_{j-1}))]}, \tag{3}$$

$$z_4(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m-h}\sum_{k=1}^{m-h} \mathbf{1}_{[|\sigma_G(t_k,\ t_{k+h})-\sigma_p(t_k,\ t_{k+h})|\leq\epsilon]}, \tag{4}$$

$$z_5(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}_{[q(t_j)>0]}, \tag{5}$$

$$z_6(\gamma,\ \mathbf{S},\ \mathbf{D}) = \frac{1}{m-h}\sum_{k=1}^{m-h} \mathbf{1}_{[\sigma_p(t_k,\ t_{k+h})\leq\epsilon]}, \tag{6}$$

où $G(t_j) = S(t_j) - D(t_j)$ représente la différence entre l'offre $S(t_j)$ et la demande $D(t_j)$, $\mathbf{1}_{[\cdot]}$ est la fonction conditionnelle définie par $\mathbf{1}_{[C]} = 1$ si la condition C est vérifiée, sinon $\mathbf{1}_{[C]} = 0$, $0 < \epsilon \ll 1$ et z_i est un ratio prenant ses valeurs dans $[0,\ 1]$ et mesurant l'efficacité du réseau de neurones proposé selon l'hypothèse i $(i = 1,\ \ldots,\ 6)$.
La performance moyenne est

$$\bar{z}(\gamma,\ \mathbf{S},\ \mathbf{D}) = \sum_{k=1}^{6} w_k z_k, \tag{7}$$

où $w_1,\ \ldots,\ w_6$ sont les poids associés aux critères ci-dessus, avec $0 \leq w_k \leq 1$, $k = 1,\ \ldots,\ 6$ et $\sum_{k=1}^{6} w_k = 1$.

4 Réseaux de neurones

De façon générale, on situe le début des réseaux de neurones artificiels à 1943 avec les travaux de McCulloch et Pitts [16]. Aujourd'hui, on retrouve les réseaux de neurones solidement implantés dans diverses industries.

Le modèle mathématique d'un neurone artificiel est essentiellement composé d'un intégrateur effectuant la somme pondérée de ses R entrées. Le résultat n de cette somme est ensuite transformé par une fonction de transfert f qui produit la sortie a du neurone. Les R entrées du neurone correspondent au vecteur $p = [p_1\ p_2\ \ldots p_R]^T$, alors que $w = [w_{1,1}\ w_{1,2}\ \ldots\ w_{1,R}]^T$ représente le vecteur des poids ou "*poids synaptiques*". La sortie n de l'intégrateur est donnée par l'équation suivante :

$$n = \sum_{j=1}^{R} w_{1,j} p_j - b, \tag{8}$$

b est appelé "*biais*" ou "*seuil d'activation*" du neurone. Le résultat n s'appelle "*niveau d'activation*" du neurone.

L'intérêt des neurones réside dans les propriétés qui résultent de leur association en réseaux. Un réseau de neurones est un ensemble de neurones interconnectés suivant une topologie de connexion. Il existe deux types de réseaux de neurones : les réseaux *statiques* (ou non bouclés) et les réseaux *dynamiques* (récurrents ou bouclés). Le graphe de ces derniers contient au moins un cycle, d'où le terme bouclage.

5 La configuration du marché producteur-consommateur

Le marché producteur-consommateur est un marché à terme avec seulement un producteur et un consommateur. Un automate vendeur est désigné à vendre la production du producteur et un automate acheteur est désigné à couvrir les besoins du consommateur.

La décision du vendeur automatisé dans la période t_j possède la forme suivante :

$$\mathbf{u_1}(t_j) = (u_{11}(t_j),\ u_{12}(t_j),\ u_{13}(t_j)), \tag{9}$$

L'ordre de l'acheteur automatisé a la forme suivante :

$$\mathbf{u_2}(t_j) = (u_{21}(t_j),\ u_{22}(t_j),\ u_{23}(t_j)), \tag{10}$$

où

- $u_{11}(t_j)$ et $u_{21}(t_j)$: le prix de vente et le prix d'achat respectivement.
- $u_{12}(t_j)$ et $u_{22}(t_j)$: la quantité offerte et la quantité demandée respectivement.
- $(u_{13}(t_j) \in [-1,\ +1])$ et $(u_{23}(t_j) \in [-1,\ +1])$: les facteurs d'ajustement de l'offre et de la demande respectivement (on note par $S_a(t_j)$ et $D_a(t_j)$ l'offre ajustée et la demande ajustée respectivement).

Une transaction aura lieu si les deux conditions suivantes sont vérifiées en même temps :
i) $u_{12}(t_j) \neq 0$ et $u_{22}(t_j) \neq 0$,
ii) $u_{21}(t_j) \geq u_{11}(t_j)$.
Dans ce cas, le prix transactionnel sera :

$$p(t_j) = \frac{u_{11}(t_j) + u_{21}(t_j)}{2}, \tag{11}$$

et la quantité transactionnelle sera :

$$q(t_j) = \min\{|u_{12}(t_j)|,\ u_{22}(t_j)\}. \tag{12}$$

En cas de non satisfaction des conditions, aucune transaction n'aura lieu à la période t_j et on pose conventionnellement :

$$p(t_j) = p(t_{j-1}) \text{ et } q(t_j) = 0. \tag{13}$$

6 Modélisation du marché à terme Producteur-Consommateur par les réseaux de neurones

Notre ultime objectif est de concevoir des automates capables de négocier les prix sur les marchés à terme, remplaçant ainsi les traders humains dans cette mission fastidieuse. Dans cette section, nous supposons que les deux automates, acheteur et vendeur, utilisent les mêmes paramètres dans leur formulation :

- du prix-désiré (u_{11} et u_{12} respectivement) à chaque instant t_j, c-à-d $u_{11}(t_j) = u_{12}(t_j)$, et d'après la relation (11), nous aurons $p(t_j) = u_{11}(t_j) = u_{21}(t_j)$;
- de la quantité désirée (u_{12} et u_{22} respectivement) à chaque instant t_j, c-à-d $u_{12}(t_j) = u_{22}(t_j)$, et d'après la relation (12), nous aurons $q(t_j) = |u_{12}(t_j)| = u_{22}(t_j)$.

Nous proposons une approche de modélisation du processus de négociation automatisée des prix et des quantités [4] pour un marché à terme de type producteur-consommateur. Cette approche consiste à représenter le système par un réseau de neurones récurrent capable de réagir aux variations de l'offre et de la demande dans la fixation des prix et des quantités à terme.
L'architecture du réseau proposé est fixée par tâtonnements, elle reste une parmi une infinité d'architectures qui peuvent être proposées pour le même réseau ; le but est d'approcher au mieux le comportement du système réel.

6.1 Présentation du réseau générateur des prix et des quantités à terme

Nous avons construit un réseau de neurones récurrent (RPQ) avec le toolbox "*Neural Network*" de MATLAB. Le réseau est constitué de neuf sous-réseaux MLP (SR(1)-SR(9)) dont chacun est composé de deux couches. Les premières couches de chacun des sous-réseaux possèdent trente neurones chacune et des fonctions d'activation tangente sigmoïde (tansig), sauf celle du premier sous-réseau qui possède une fonction d'activation radiale ou RBF (Radial Basis Function). Les secondes couches de chacun des sous réseaux possèdent un neurone chacune et des fonctions d'activation linéaires.

Les entrées externes du réseau sont : le prix nominal à l'instant $t_{j-1} : p^*(t_{j-1})$, l'offre à l'instant $t_j : S(t_j)$, la demande à l'instant $t_j : D(t_j)$.

Le réseau retourne deux sorties qui sont : le prix transactionnel à l'instant $t_j : p(t_j)$ et la quantité transactionnelle à l'instant $t_j : q(t_j)$.
où : $y_1(t_j)$ est le nombre de contrats que le producteur a vendu depuis l'instant t_1 jusqu'à t_j. $y_2(t_j)$ est le nombre de contrats que le consommateur a acheté depuis l'instant t_1 jusqu'à t_j.

Le réseau générateur des prix et des quantités à terme est représenté sur la figure 2.

[4] On note par quantité le nombre de contrats échangés lors d'une transaction donnée.

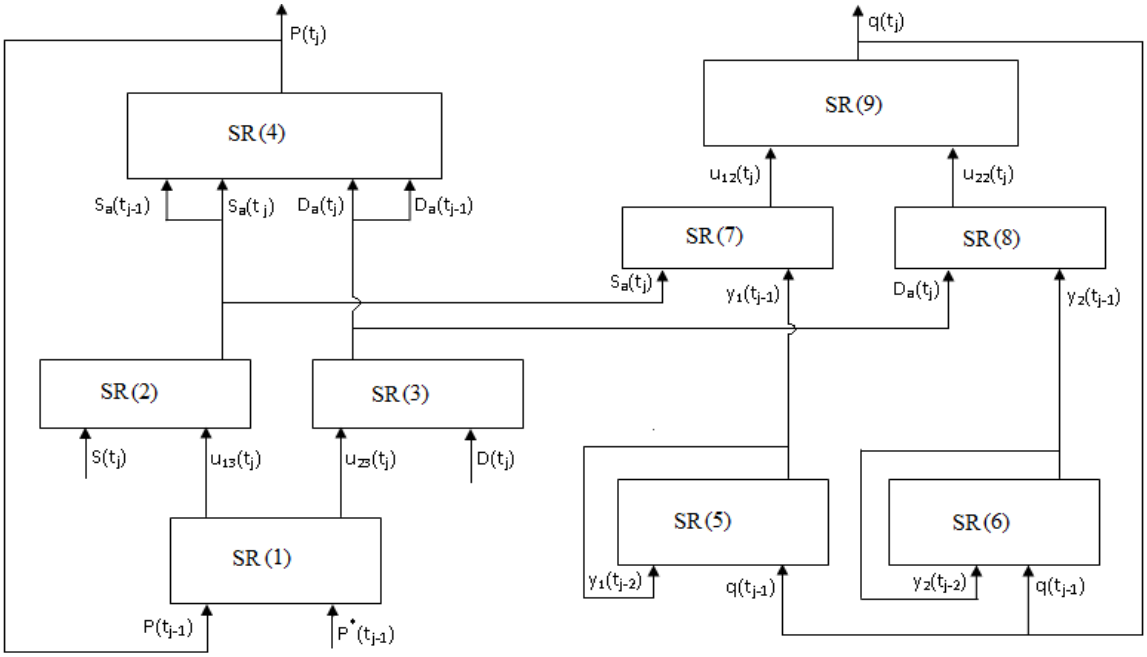


Fig. 2. Modèle de réseau de neurones générant les prix et les quantités à terme

7 Génération des données de l'apprentissage

7.1 Génération des entrées

Dans les marchés à terme, l'offre et la demande sont estimées par des personnes renvoyant leurs prévisions à chaque période. Ces estimations ne seront connues qu'une fois réalisées. Ces prévisions sont donc des variables aléatoires.

En pratique, une étude statistique est nécessaire pour détecter le type de loi à assigner au phénomène aléatoire de l'offre et de la demande et les valeurs de ses paramètres. Dans notre travail, nous utilisons un échantillon des prévisions de l'offre : $S(t_0),\ \ldots,\ S(t_m)$ et un échantillon des prévisions de la demande : $D(t_0),\ \ldots,\ D(t_m)$ comme des données d'apprentissage. En raison de la non connaissance des lois exactes qui régissent l'offre et la demande dans les marchés à terme, nous avons été amenés à supposer que ces dernières suivent, par exemple, une lois normale.

Nous supposons que : le nombre de périodes $m = 100$, l'offre suit une loi normale de moyenne $\mu_S = 5000$ et d'écart type $\sigma_S = 0.5$ et la demande suit la même loi de moyenne $\mu_D = 5000$ et d'écart type $\sigma_D = 0.3$. L'évolution des prévisions de l'offre et de la demande durant les m périodes est représentée sur la figure 3.

7.2 Génération des sorties désirées

Dans notre travail, nous avons d'abord cherché des prix et des quantités de transactions (en fonction de l'offre, de la demande et du prix nominal) maximisant la performance moyenne $\bar{z}$. Puis, nous les avons utilisés comme des sorties désirées pour le réseau de neurones proposé. Pour maximiser $\bar{z}$, nous avons opté pour les algorithmes génétiques en utilisant le toolbox "*Genetic Algorithm and Direct Search*" de MATLAB. Les paramètres utilisés dans l'algorithme génétique sont ceux qui existent par défaut : la taille de la population a été fixée à 100 ; la probabilité de croisement est de 0.8 ; le nombre de générations fixé à 100 ; le type de codage utilisé est le binaire.

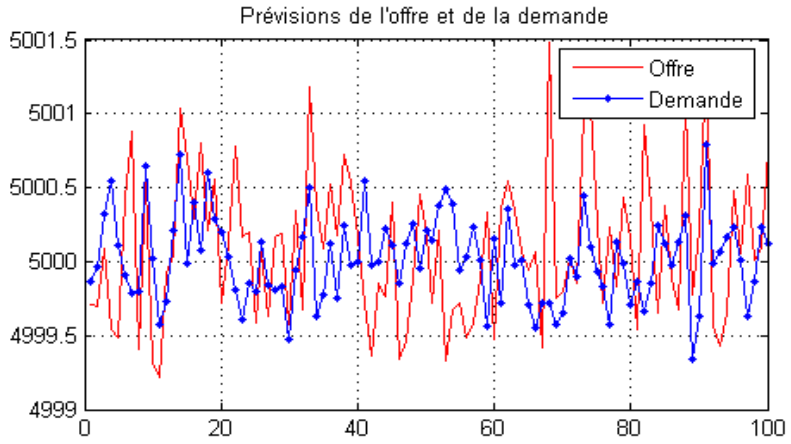


Fig. 3. Prévisions de l'offre et de la demande

8 Phase d'apprentissage

Les entrées sont injectées au réseau séquentiellement dans le temps (chaque séquence représente une période de transaction dans le marché à terme). L'algorithme d'apprentissage utilisé est le "*TRAINS*" (*Sequential order incremental training w/learning functions*). Nous avons défini cet algorithme en mettant : net.adaptFcn='trains'.
Le critère d'erreur utilisé est l'erreur quadratique moyenne : "*MSE*" (*Mean Squared Error performance function*). Cette fonction est définie dans le toolbox par la commande : net.performFcn='mse'. Le taux d'apprentissage choisi est celui par défaut : 0.1.

Les poids associés au réseau ont été initialisés par l'algorithme de Nguyen-Widrow (dans le toolbox, l'initialisation d'une couche i donnée suivant cet algorithme se fait par la commande : net.layersi.initFcn='initnw'). La courbe des prix générés par le réseau (accompagnée de la courbe des écarts entre les prévisions de l'offre et celles de la demande) est représentée sur la figure 4.

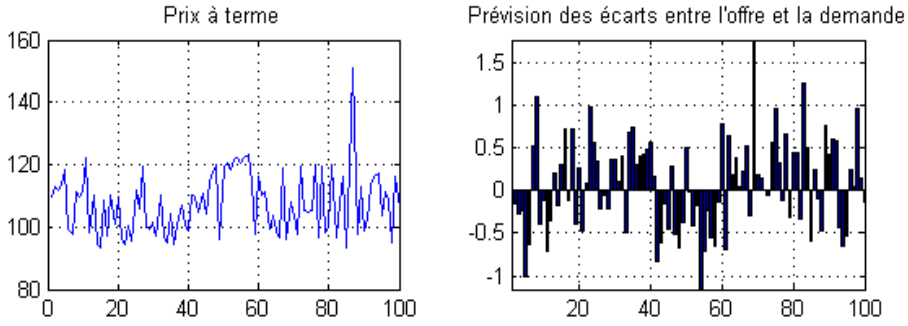


Fig. 4. Prix à terme générés par le réseau de neurones

Nous constatons bien que les variations des prix sont inversement proportionnelles aux variations des écarts entre les prévisions de l'offre et celles de la demande.

La courbe des quantités transactionnelles générées par le réseau est représentée sur la figures 5.

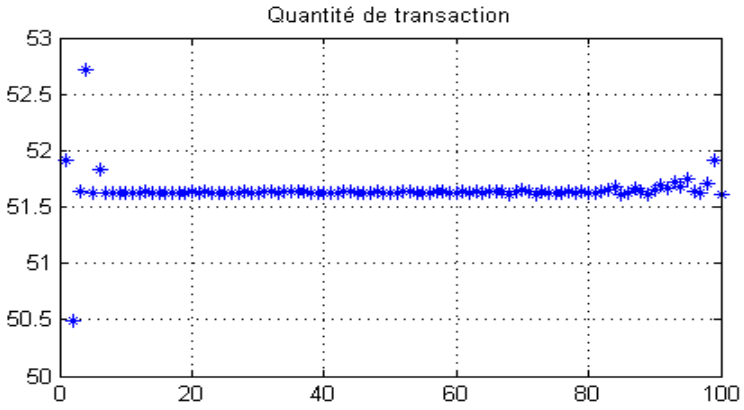


Fig. 5. Quantités de transactions générées par le réseau de neurones

9 Comparaison des sorties prix du réseau aux sorties désirées d'apprentissage

À la fin de l'apprentissage, nous avons obtenu le graphe de la figure 6.

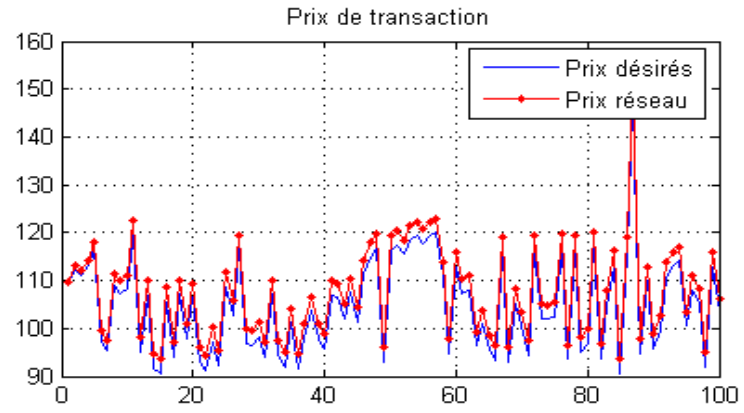


Fig. 6. Sorties du réseau prix et sorties désirées

Nous constatons que les deux courbes se rapprochent, l'erreur quadratique moyenne est égale à 0.0348.

10 Etude de performances

Soient les notations suivantes :

- Z_g : performances des résultats obtenus à travers les algorithmes génétiques en maximisant la moyenne $\bar{z}$ des critères de performances suggérés dans les formules (1)-(6).
- Z_{rg} : performances des résultats obtenus par les réseaux de neurones dont les sorties désirées sont celles obtenues à travers les algorithmes génétiques en maximisant la moyenne $\bar{z}$.

Un tableau comparatif sur les performances des résultats obtenus par les sorties du réseau proposé et les sorties désirées est représenté sur la table 1.

	z_1	z_2	z_3	z_4	z_5	z_6	$\bar{z}$
Z_g	0.9495	0.7245	1.0000	0	1.0101	1.0000	0.7807
Z_{rg}	0.9495	0.7245	0.8889	0	1.0101	1.0000	0.7622
Z_g	0.9192	0.7449	1.0000	0	1.0101	0.9888	0.7772
Z_{rg}	0.9192	0.7449	0.9394	0	1.0101	0.9888	0.7671
Z_g	0.8788	0.7449	1.0000	0	1.0101	1.0000	0.7723
Z_{rg}	0.8788	0.7449	1.0000	0	1.0101	1.0000	0.7723
Z_g	0.9394	0.7551	1.0000	0	1.0101	1.0000	0.7841
Z_{rg}	0.9394	0.7551	0.9596	0	1.0101	1.0000	0.7774
Z_g	0.8889	0.7959	1.0000	0	1.0101	1.0000	0.7825
Z_{rg}	0.8889	0.7959	0.9091	0	1.0101	1.0000	0.7673
Z_g	0.8889	0.7755	1.0000	0	1.0101	1.0000	0.7791
Z_{rg}	0.8889	0.7755	0.9596	0	1.0101	1.0000	0.7723
Z_g	0.9192	0.7857	1.0000	0	1.0101	1.0000	0.7858
Z_{rg}	0.9192	0.7857	0.8485	0	1.0101	1.0000	0.7606
Z_g	0.9091	0.7653	1.0000	0	1.0101	1.0000	0.7807
Z_{rg}	0.9091	0.7653	0.9394	0	1.0101	1.0000	0.7706
Z_g	0.9091	0.7653	1.0000	0	1.0101	1.0000	0.7807
Z_{rg}	0.9091	0.7653	0.9394	0	1.0101	1.0000	0.7706
Z_g	0.9192	0.7857	1.0000	0	1.0101	1.0000	0.7858
Z_{rg}	0.9192	0.7857	0.8485	0	1.0101	1.0000	0.7606
Z_g	0.9091	0.7653	1.0000	0	1.0101	1.0000	0.7807
Z_{rg}	0.9091	0.7653	0.9394	0	1.0101	1.0000	0.7706

Tab. 1. Performances obtenues par les sorties du réseau proposé et les sorties désirées

11 Analyse des résultats du tableau

Nous remarquons que les performances les plus élevées sont obtenues directement à travers les algorithmes génétiques en maximisant la moyenne $\bar{z}$. Mais ce qui nous intéresse est un modèle fixe capable d'apprendre le mécanisme du marché à terme, c'est-à-dire : le modèle d'un réseau de neurones. Nous constatons que les performances des résultats du réseau de neurones sont légèrement plus baisses que les précédentes ; cela s'explique car l'apprentissage peut être bon mais pas parfait.

12 Test et généralisation

À la fin de la phase d'apprentissage, nous avons testé si le réseau (pour la courbe des prix, par exemple) peut se généraliser à des données n'appartenant pas à l'ensemble d'apprentissage. Nous avons généré un échantillon de taille 20 (de même loi de probabilité que l'ensemble d'apprentissage). Les résultats sont représentés sur la figure 7.

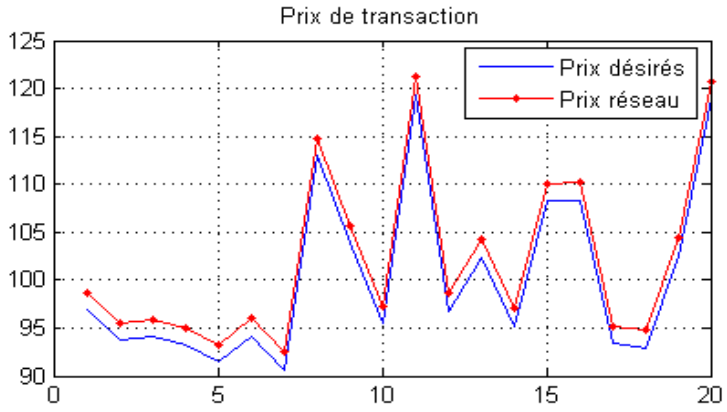


Fig. 7. Sorties prix du réseau et sorties désirées

L'erreur quadratique obtenue est égale à 0.4, nous remarquons que les résultats ne se rapprochent pas très bien des résultats attendus, donc nous concluons que le réseau ne possède pas une très bonne capacité de généralisation à d'autres données (de même loi).

Nous avons obtenu des résultats similaires pour la courbe des quantités transactionnelles.

13 Application d'autres algorithmes pour l'apprentissage

Nous avons utilisé d'autres algorithmes pour effectuer la phase d'apprentissage. Les résultats obtenus ne sont pas satisfaisants.

À titre d'exemple, après application de l'algorithme de rétro-propagation du gradient, la figure 8 montre le grand écart entre la courbe des prix à terme générés par le réseau et la courbe des prix désirés.

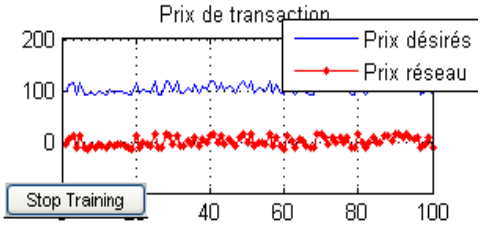


Fig. 8. Sorties du réseau prix et sorties désirées

Nous déduisons que l'algorithme séquentiel : "Sequential order incremental training" est le plus approprié pour l'apprentissage de notre réseau de neurones. Cela est dû à l'aspect séquentiel des données (dynamisme du système du marché à terme).

Conclusion

Après avoir réalisé notre étude et analysé les résultats, nous en tirons certaines conclusions :

– le mécanisme de la plate-forme du marché à terme peut être modélisé par les réseaux de neurones, ce qui donne la possibilité d'avoir un modèle capable d'apprendre son comportement (détermination des prix et des quantités transactionnels) ;

– le choix du modèle de réseaux de neurones influe considérablement sur les performances du modèle et, par conséquent, sur les performances des résultats ;

– la méthode que nous avons adoptée en maximisant d'abord les critères de performances suggérés en utilisant les algorithmes génétiques a donné de bons résultats pour les modèles de réseaux de neurones.

Références

1. R. Lawrence. Using neural networks to forecast stock market prices. Department of Computer Science, University of Manitoba, December 12 1997.
2. R J. Van Eyden. The Application of Neural Networks in the Forecasting of Share Prices. Finance and Technology Publishing, 1996.
3. J. Cordier. Les marchés à terme. Paris, 1984.
4. D. Bigman, D. Goldfarb and E. Schechtman, Futures markets efficiency and the time content of the information sets, Journal of Futures Markets, (1983), pp. 321–334.
5. B. K. Wong, T. A. Bonovich and Y. Selvi, Neural network applications in business : A review and analysis of the literature, Decision Support Systems, 19 (1997), pp. 301–320.
6. R. Trippi and E. Turban. Neural Networks in Finance and Investment : Using Artificial Intelligence to Improve Real-world Performance. Chicago : Probus, 1993.
7. A. N. Refenes. Neural Networks in the Capital Markets. Chicester : Wiley, 1995.
8. M. D. Odom and R. Sharda, A neural network model for bankruptcy prediction, In : Proceedings of the IEEE International Joint Conference on Neural Networks. San Diego, CA, 2 (1990), p. 163–168.
9. K. G. Coleman, T. J. Graettinger and W. F. Lawrence, Neural networks for bankruptcy prediction : The power to solve financial problems, AI Review, (1991), p. 48–50.
10. L. M. Salchenkerger, E. M. Cinar and N. A. Lash, Neural networks : A new tool for predicting thrift failures, Decision Science, 23(4) (1992), p. 899–916.
11. K. Y. Tam and M. Y. Kiang, Managerial applications of neural networks : The case of bank failure predictions, Management Science, 38(7) (1992), p. 926–947.
12. R. Wilson and R. Sharda, Bankruptcy prediction using neural networks, Decision Support Systems, 11 (1994), p. 545–557.
13. A. S. Weigend, D. E. Rumelhart and B. A. Huberman, Generalization by weight-elimination with application to forecasting, Advances in Neural Information Processing Systems, 3 (1991), p. 875–882.
14. Y. Simon and D. Lautier. Marchés dérivés de matières premières et gestion du risque de prix. ECONOMICA. Paris, 2^e édition, 2001.
15. M.S. Radjef and F. Laib. On the Mechanism of the Futures Market : a Formulation and Some Analytical Properties. Communication presented at the 13th International Symposium on Dynamic Games and Applications, Wroclaw (Poland), 30th June-3rd July, 2008
16. W. S. McCulloch and W. Pitts, A logical calculus of the ideas immanent in nervous activity, Bull. Math. Biophysics, (1943), pp. 5 :115–133.

Agents, ontologies et applications

Un Modèle SMA pour le Diagnostic Collectif

Khaled Allem[1], Ramdane Maamri[1] and Zaidi Sahnoun[1],

[1] Laboratoire LIRE, Université Mentouri de Constantine, Algérie
kallem9@yahoo.fr, rmaamri@yahoo.fr, zsahnoun@yahoo.fr

Abstract. La plupart des systèmes de diagnostic sont centralisés, le module du diagnostic est contrait d'avoir une vue globale du système physique et contient la description de ce dernier dans sa totalité ce qui laisse apparaître quelques problèmes liés principalement à la complexité du système physique à diagnostiquer d'un côté et de l'autre à une architecture figée et peu robuste du système de diagnostic. Dans ce cas il est évident de penser à distribuer la procédure de diagnostic. Les systèmes multi agents faisant l'objet d'intérêts croissants semblent particulièrement bien adaptés à la problématique posée. Notre travail consiste donc à proposer un modèle SMA pour le diagnostic collectif des systèmes complexes en utilisant la technique du diagnostic logique à base de cohérence, avec un intérêt particulier pour la distribution de l'étape d'analyse diagnostic.

Keywords: Diagnostic distribué, SMA, Diagnostic à base de cohérence, DX, FDI.

1 Introduction

La complexité croissante des systèmes industriels automatisés et les contraintes de compétitivité en terme de coût de production, disponibilité et sécurité des installations, ont mobilisé durant ces dernières années une large communauté de chercheurs pour améliorer la surveillance et le diagnostic de ce type de procédés, ainsi la complexité de la tâche de diagnostic a motivé la recherche de son automatisation. Le diagnostic est un thème de recherche fédérant différentes communautés scientifiques (Automatique, Informatique, Productique…), aujourd'hui au cœur des préoccupations industrielles. Il a pour but d'établir un lien entre un symptôme observé, la défaillance qui est survenue et ses causes.

Pour concevoir des systèmes de diagnostic, il existe deux approches assez différentes :
Dans la 1ière approche, souvent appelée *diagnostic à partir des principes premiers ou à base de cohérence*, on commence avec une description du système ainsi qu'une observation du comportement de ce dernier, si cette observation entre en conflit avec la façon dont le système est supposé se comporter, on est alors confronté au problème du diagnostic, à savoir, déterminer les composant du système qui quand supposés fonctionner anormalement, expliquera la contradiction entre le comportement observé

et le comportement correcte du système. Plusieurs travaux ont été réalisés dans ce domaine notamment ceux de [1], [2], [3] et plus récemment [4], [5], [6], [7], [8].

Pour la 2ième approche, décrite souvent comme une *approche expérimentale,* l'information heuristique joue un rôle dominant. Les systèmes de raisonnement diagnostic correspondant essaient de codifier les règles du domaine, des intuitions statistiques et l'expérience antérieure des experts humains. Les diagnostics réussis proviennent de l'expérience codifiée de l'expert humain plutôt de ce qui se réfère souvent à la connaissance « profonde » du système à diagnostiquer. Le système expert MYCIN est un parfait exemple pour cette approche.

La plupart des systèmes de diagnostic sont centralisés, le module de diagnostic centralisé est contraint d'avoir une vue globale du système physique et contient la description de ce dernier dans sa totalité. Ceci laisse apparaître quelques problèmes liés principalement à la complexité du système physique à diagnostiquer d'un côté et de l'autre à une architecture figée et peu robuste du système de diagnostic.

L'approche de diagnostic centralisé rencontre quatre problèmes principaux [12]:

Problèmes liés à la complexité du système : l'état du système est habituellement l'ensemble des états de ses composants.

Problèmes liés à une architecture peu robuste : un problème se produisant à l'intérieur d'un système centralisé de diagnostic peut conduire à l'échec total du système. En revanche, dans un système distribué, la défaillance d'un module (par exemple de détection, ou de localisation) qui participe au diagnostic ne met pas en péril toute la procédure de diagnostic.

Problèmes liés à une architecture figée : toute modification ou évolution structurelle du système nécessite une réécriture plus ou moins complète du programme de diagnostic.

Problèmes liés à la distribution des calculs : en effet, toutes les tâches d'une procédure de diagnostic sont exécutées sur un calculateur centralisé. Distribuer les différents calculs liés au diagnostic sur des machines différentes s'avère difficile dans une approche centralisée.

Il n'est donc plus possible de se contenter d'une approche centralisée et figée pour la conception d'un système de diagnostic destiné à des systèmes complexes.

Distribuer la procédure de diagnostic semble être une évidence dans ce cas, dès lors apparaît l'idée de multiplier les entités intelligentes et de les doter de facultés de communication afin qu'elles aient le moyen de communiquer entre elles pour que se construise par *coopération* une solution globale au problème du diagnostic. Dans une approche distribuée, chaque entité a une vue partielle ou locale du système à diagnostiquer. Par conséquent, pour construire une solution globale, logique et cohérente il faut pouvoir rassembler les solutions partielles. Donc, il faut que les entités coopèrent entre elles afin de partager leurs solutions et faire part de leurs problèmes et coordonner leurs activités.

Les *systèmes multi-agents*, faisant l'objet d'intérêts croissants, semblent particulièrement bien adaptés à la problématique, en effet, un agent peut être spécialiste dans un domaine de compétence dans lequel il peut intervenir sans pour autant trouver la solution globale, en revanche, l'interaction entre les différents agents fait qu'il y à émergence de la solution globale. L'approche multi-agents permet de mettre en œuvre une solution calquée sur l'organisation humaine, plutôt que d'envisager une seule entité omnipotente et omnisciente.

Le premier avantage à utiliser ce paradigme est de conduire à des systèmes modulaires et ouverts où le fait d'ajouter un agent ou de modifier la structure d'un système n'induit pas une re-conception d'une solution mais converge automatiquement vers une nouvelle solution globale. L'autre grand avantage est de permettre de résoudre des problèmes complexes en les décomposant en une multitude de problèmes plus élémentaires, en construisant des entités autonomes qui pourront, en coopérant, participer à la construction d'une solution globale.

Le diagnostic concerne les deux phases indissociables de détection et d'analyse pour lesquelles il existe une large panoplie de méthodes et de techniques proposées par différentes communautés de recherches (FDI, DX, SED)[1].

Notre objectif est de proposer un modèle SMA pour le diagnostic collectif des systèmes complexes, où avant même que les agents puissent établir un diagnostic global ils doivent d'abord calculer des diagnostics locaux utilisant leurs connaissances partielles du système. Pour ce faire, nous avons choisi d'adopter une approche qui vise à utiliser conjointement les méthodes de diagnostic issues de la communauté FDI pour la phase de détection, et les méthodes développées par la communauté DX pour l'étape de l'analyse diagnostic afin de pouvoir tirer profit des avantages de chacune et avoir ainsi une certaine complémentarité. Dans notre travail, nous nous sommes intéressés spécialement à la distribution de l'étape d'analyse diagnostic en utilisant la méthode dite du diagnostic à base de cohérence pour l'élaboration des diagnostics locaux.

Ainsi, notre travail s'inscrit dans le domaine du diagnostic à base de modèle et consiste à proposer un modèle SMA pour le diagnostic collectif en s'appuyant :

- d'une part sur une technique d'analyse formelle du diagnostic, en l'occurrence le diagnostic à base de consistance (cohérence) qui permet de garantir la justesse de l'analyse diagnostic, et
- d'autre part, sur le paradigme multi-agents permettant de renforcer et de compléter le résultat du diagnostic.

L'article est organisé comme suit :

La section 2 introduit la procédure de diagnostic, dans la section 3 nous présentons le modèle SMA proposé, une simulation fera l'objet de la section 4, la section 5 dresse une comparaison avec des travaux existants et enfin une conclusion et des perspectives sont présentées dans la section 6.

2 Procédure de Diagnostic

La détection et la localisation constituent les deux étapes essentielles d'une procédure de diagnostic, ceci dit, des auteurs proposent d'ajouter d'autres étapes afin de raffiner encore plus le diagnostic et ce par rapport à des objectifs fixés liés aux méthodes de diagnostic utilisées.

[1] FDI: Fault Detection and Isolation, communauté de l'automatique
DX : Communauté de l'Intelligence Artificielle
SED : Communauté des Systèmes à Evènement Discret

2.1 La Détection

Egalement appelée « génération de symptômes », il s'agit de vérifier, grâce à des tests, la consistance entre des informations sur le comportement réel d'un système physique tel qu'il peut être observé par l'intermédiaire de capteurs par exemple et son comportement attendu tel qu'il peut être prédit grâce aux modèles de bon ou mauvais comportement. Toute contradiction entre les observations et les prédictions déduites des modèles est nécessairement la manifestation d'un dysfonctionnement, c'est-à-dire de la présence d'un ou plusieurs défauts, la détection détermine donc le fonctionnement normal ou anormal du système

Différents types d'algorithmes de détection dédiés aux systèmes physiques ont été conçus par les chercheurs des communautés FDI, SED et DX. Dans la plupart des cas, les méthodes de diagnostic sont liées à la connaissance disponible sur le système et à sa représentation et sont classées de différentes façons par de nombreux auteurs [13], [14], [15] et [16].

2.2 L'Analyse

Connue aussi sous l'appellation de localisation ou raisonnement diagnostic, elle consiste à analyser les symptômes disponibles fournis par les tests de détection pour déterminer les états plausibles d'un système physique. La localisation permet de remonter à l'origine de l'anomalie et de localiser le ou les composants défectueux. Les méthodes d'analyse sont liées à la connaissance disponible sur le système à diagnostiquer et à sa représentation et sont classées de différentes façons par de nombreux auteurs [5], [7], [8]. La terminologie et la classification ne sont pas toujours homogènes, influencées par les contextes et les terminologies particulières à chaque communauté et domaine d'application. Les méthodes de diagnostic varient selon le type de connaissance du système à diagnostiquer, selon la façon de structurer cette connaissance et de l'utiliser lors de la génération d'un diagnostic. Il est donc possible de classer les méthodes de diagnostic selon l'un ou l'autre de ces trois aspects. La classification présentée dans [7] est basée sur le type de connaissance utilisé pour le diagnostic de défaillances. Trois grandes catégories de méthodes sont identifiées dans cette classification : les approches à base de règles, les approches à base de modèles et les approches à base de données.

Les techniques de diagnostic à base de modèles (DBM) issues de la communauté de l'intelligence artificielle sont fondées sur une théorie logique [1]. Dans ces approches la détection est considérée comme une tâche du diagnostic. Les premiers travaux s'appuyaient sur des associations de connaissances empiriques, comme ce qui est fait dans les systèmes experts. Ces approches utilisent des modèles basés sur la connaissance du système à diagnostiquer. De façon générale, deux types d'approches de DBM peuvent être distinguées : celles s'appuyant sur un modèle de comportement anormal (fautes) et celles qui reposent sur des modèles du comportement normal (bon) du système, dans ce dernier cas, le modèle décrit uniquement comment se comporte le système quand il fonctionne correctement. De nombreux travaux dans ce domaine sont connus sous l'appellation de *« diagnostic à partir des principes premiers »* ou à base de cohérence [1], [2], [3] et s'appuient sur un modèle de la

structure du système et du comportement de ses composants, pour effectuer des prédictions sur les états du système. Le diagnostic à base de cohérence et particulièrement *« la méthode de Reiter »* sera adopté dans la suite de notre travail.

3 Un Modèle SMA pour le Diagnostic Collectif

Le modèle SMA que nous proposons aura les caractéristiques suivantes :
- La connaissance sur le système est sémantiquement distribuée à travers les différents agents.
- Les agents utilisent des modèles de bon comportement dans l'étape d'analyse (localisation) et éventuellement des modèles de mauvais comportement dans l'étape d'identification.
- Les agents coopèrent pour le calcul des diagnostics locaux en garantissant la cohérence de leurs résultats pour établir le diagnostic global.

Ce choix est justifié par :
- L'idée derrière la distribution sémantique est l'intégration d'agents hétérogènes en provenance de différents groupes et possédant chacun sa propre vision sur le système (agent expert dans un domaine particulier), ainsi la structure interne de chaque agents n'aura guère d'importance dans la procédure de diagnostic.
- L'utilisation des modèles de bon comportement correspond par nature au type de raisonnement diagnostic adopté qui est en l'occurrence le diagnostic à base de consistance.
- Malgré l'autonomie dans le calcul des diagnostics locaux offerte par la distribution sémantique des connaissances sur les différents agents, ces derniers sont contraints de coopérer pour les raisons suivantes :
  a. augmenter l'efficacité du calcul,
  b. minimiser la probabilité d'avoir des diagnostics contradictoires,
  c. maximiser les chances d'avoir des diagnostics complémentaires.

Afin qu'elle puisse être utilisable comme méthode de raisonnement diagnostic dans le modèle SMA proposé, nous avons essayé d'adapter la théorie de Reiter à notre contexte de travail.

3.1 Théorie de Reiter dans un Contexte Multi-agents

Soit SD_i la description du sous système associé à l'agent A_i, $COMPS_i$ ses composants et OBS_i ses observations.

Définition 1. $\Delta_i \subseteq COMPS_i$ est un diagnostic associé à l'agent A_i pour $(SD_i, COMPS_i, OBS_i)$ ssi : $SD_i \cup OBS_i \cup \{\neg AB(c) / c \in COMPS_i - \Delta_i\}$ est consistante.

La coordination entre les agents est nécessaire afin de maintenir la cohérence entre les différents diagnostics locaux, ce qui est traduit formellement par la définition suivante:

Définition 2. $\Delta_i \subseteq COMPS_i$ est un diagnostic associé à l'agent A_i pour $(SD_i, COMPS_i, OBS_i)$ ssi la formule suivante est consistante :

$SD_i \cup OBS_i \cup \{\neg AB(c) / c \in COMPS_i - \Delta_i\} \bigcup_{\substack{j=1 \\ j \neq i}}^{k} \{AB(c) / c \in \Delta_j\}$ tel que Δ_j est le diagnostic associé à l'agent Aj.

La formule précédente peut être écrite comme suit :

$SD_i \cup OBS_i \cup \{\neg AB(c) / c \in COMPS_i - \Delta_i\} \cup \{AB(c) / c \in \Delta_j\} \cup ... \cup \{AB(c) / c \in \Delta_k\}$

ou plus concrètement sous la forme :

$SD_i \cup OBS_i \cup \{\neg AB(c) / c \in COMPS_i - \Delta_i\} \cup D_j ... \cup D_k$

Ainsi le diagnostic local Di obtenu par l'agent Ai est calculé en tenant compte des résultats (diagnostics locaux) des autres agents Aj, …, Ak.

Définition 3. Soient D_1, D_2, D_3,…, D_n des diagnostics locaux associés respectivement aux agents A_1, A_2, A_3,…, A_n, alors $D_G = D_1 \cup D_2 \cup ... \cup D_n$ est un diagnostic global pour (SD, COMPS, OBS) tel que :

$SD = \bigcup_{i=1}^{n} SD_i$, $COMPS = \bigcup_{i=1}^{n} COMPS_i$, $OBS = \bigcup_{i=1}^{n} OBS_i$.

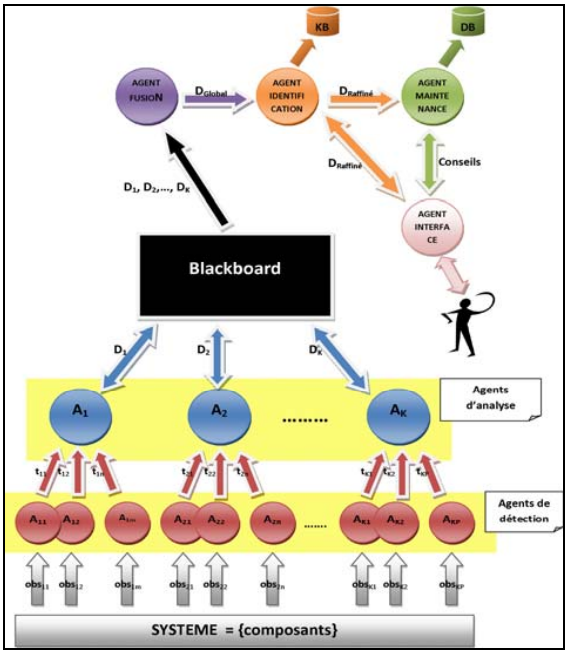


Fig. 1. Un Modèle SMA pour le Diagnostic Collectif.

3.2 Description du Modèle

La détection et la localisation sont généralement les deux étapes les plus importantes dans une procédure de diagnostic, ceci dit, d'autres étapes peuvent suivre afin de raffiner encore plus le diagnostic et proposer éventuellement une aide à l'opérateur en matière de pronostic et de conseils de maintenance. En se basant sur ce fait, nous avons choisi d'associer à chaque étape de diagnostic un type d'agent, comme suit :

1. Agents de Détection. Les agents de détection jouent les rôles suivants :

- Acquisition et traitement des données : les agents de détection permettent d'observer le système en récupérant les données de ce dernier par des capteurs, puis les traiter en éliminant les points aberrants et en les formatant afin de les préparer pour l'étape suivante.
- Génération des symptômes : chaque agent de détection encapsule un algorithme de détection (traitement du signal, espace de parité, observateurs d'état, …), qui grâce à des tests effectués sur un certain nombre de composants (support de test) et à partir des données récupérées précédemment permet de générer des symptômes.

2. Agents d'analyse. Les symptômes fournis par les agents de détection peuvent être regroupés dans une table de signatures qui servira de base pour le raisonnement diagnostic. Dans le modèle que nous proposons, le raisonnement diagnostic est totalement distribué à travers plusieurs agents d'analyse selon une distribution sémantique des connaissances. Ainsi, chaque agent d'analyse aura comme tâche de diagnostiquer un sous ensemble de composants du système et ce d'une manière autonome tout en gardant la possibilité d'avoir des agents d'analyse avec des connaissances dépendantes, ce qui implique donc une coopération des agents pour le calcul des diagnostics locaux. Cette coopération est traduite au niveau de la génération de l'arbre « pruned HS-Tree » selon la méthode de Reiter que nous avons légèrement modifiée en introduisant le paradigme multi agent et en exploitant les résultats partiels fournis par les différents agents d'analyse.

a. commencer la génération de l'arbre T en largeur d'abord et en fixant sa profondeur (niveau).
b. coopération par réutilisation des étiquettes des nœuds : soit l'étiquette $S \in F$ du nœud n calculée par l'agent A et soit n' un nœud tel que l'agent A' se charge du calcul de son étiquette, si $H(n') \cap S = \{\}$, alors l'agent A' utilisera l'étiquette calculée par l'agent A, c'est-à-dire S comme étiquette de n' et évitera d'une part l'accès inutile à F et d'autre part un appel coûteux à la fonction TP.
c. si le nœud n est étiqueté par √ et si n' est un nœud tel que : $H(n) \subseteq H(n')$ alors n' sera fermé et l'agent A' ne calculera pas les successeurs de n' (un nœud fermé sera marqué dans l'arbre par x).
d. si les nœuds n et n' sont étiquetés respectivement par S et S' de F, et si S' est un sous ensemble de S, alors pour chaque élément $\alpha \in S - S'$, le nœud sortant de n et étiqueté par α sera marqué comme redondant. L'arc redondant ainsi que le sous arbre généré à partir de cet arc devraient être supprimés afin de préserver la minimalité des ensembles « hitting sets » pour F.

3. Agent de fusion

- Calcul de la plausibilité circonstancielle : l'agent de fusion reçoit les différents diagnostics locaux fournis par les agents d'analyse et calcule pour chaque ensemble de diagnostic local la plausibilité circonstancielle correspondante.
- Résolution des conflits : Deux diagnostics locaux D_1 et D_2 sont en conflit lorsque la formule $D_1 \cup D_2$ est inconsistante. Le conflit est résolu en comparant la plausibilité circonstancielle pour les diagnostics conflictuels en optant pour celui ayant la plausibilité la plus élevée.
- Fusion des diagnostics locaux : après l'élimination des conflits éventuels, les diagnostics locaux D_i sont fusionnés en un seul diagnostic global représenté par la formule : $D_G = D_1 \cup D_2 \cup ... \cup D_n$.

4. Agent d'identification

- Raffinement du diagnostic : jusqu'à présent, le comportement des composants a été exprimé par les deux modes : AB et $\neg$ AB, dont seul le mode normal est modélisé. La disponibilité des modèles de mauvais comportement permet d'affecter un mode de défaut à chaque composant défaillant, rendant le diagnostic plus précis et plus significatif.
- Sauvegarde du diagnostic final dans une base de connaissances pour une éventuelle utilisation en temps que connaissances expertes.

5. Agent de maintenance

- Effectuer un pronostic : ça consiste à prédire l'évolution des défauts à partir des résultats du diagnostic, cette prédiction peut être faite par un graphe causal dans lequel on peut déterminer l'ensemble des variables liées causalement aux composants défaillants.
- Déterminer la politique de maintenance : grâce aux résultats fournis par le pronostic, l'opérateur humain décide de la politique de maintenance à appliquer. Pour un procédé industriel, il peut s'agir d'une vérification, d'une réparation ou d'un remplacement de composants défectueux, alors que dans un contexte médical, c'est plutôt, des conseils donnés au patient, une prescription de médicaments ou même une intervention chirurgicale. L'agent de maintenance utilise pour accomplir cette tâche des connaissances expertes préalablement stockées dans une base de données.

6. Agent d'interface

- Interaction avec les différents agents : l'agent d'interface implémente une interface homme machine permettant à l'opérateur de choisir, d'initialiser et d'interroger les différents agents du système.
- Visualiser les diagnostics : il permet de visualiser les diagnostics aux niveaux localisation et identification ainsi que les conseils de maintenance.

3.3 Communication dans le SMA

Le modèle que nous proposons supporte deux types de communication inter agents :

a. *Communication par partage d'information :* utilisée par les agents d'analyse pendant l'étape de localisation, où les agents partagent leurs résultats partiels à travers une structure de tableau noir contenant les diagnostics locaux ainsi que les symptômes sous forme de tables de signatures associés à ces diagnostics.

b. Communication par envoi de messages : c'est le mode de communication adopté par les agents pour le reste des étapes de la procédure de diagnostic. Le langage de communication des agents ACL-FIPA est utilisé pour cela.

4 Simulation avec JADE

Nous avons choisi de simuler le comportement des différents agents de notre modèle appliqué au diagnostic d'un bioprocédé [12] en utilisant la plateforme JADE pour le développement des SMA, et ce à travers l'IDE ECLIPSE.

L'enregistrement de nos différents agents dans le DF (Director Facilitor) est montré par la figure suivante :

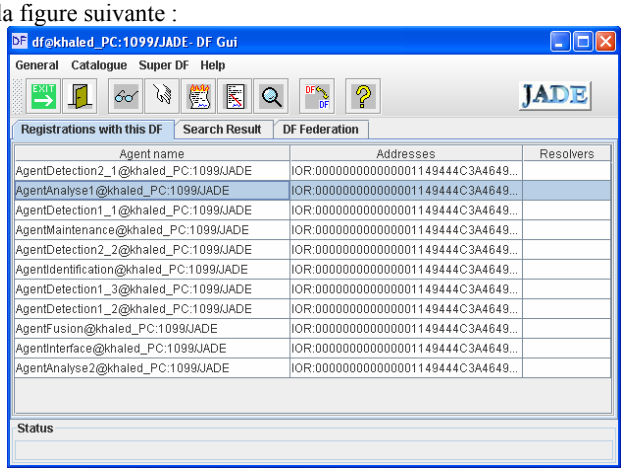


Fig. 2. Enregistrement des différents agents dans le DF.

Nous avons également utilisé un « behaviour composé » appelé « FSMBehaviour » pour implémenter les comportements de quelques agents, par exemple le comportement de « l'agent analyse 1 » dans son interaction avec « l'agent interface ».

```
public class AnalyseAgent1 extends Agent  {
 Boolean  stop = false;
protected void setup ()  {
...
FSMBehaviour AnalyseAgent1_beh= new FSMBehaviour ();
AnalyseAgent1_beh.registerFirstState(new
 attendreinstantiation (),"attendreinstanciation");
AnalyseAgent1_beh.registerState(new
 demandediagnostic(),"demandediagnostic");
AnalyseAgent1_beh.registerState (new fin (), "fin");
AnalyseAgent1_beh.registerTransition
 ("attendreinstanciation","demandediagnostic",0);
AnalyseAgent1_beh.registerTransition
 ("attendreinstanciation","fin",1);
```

```
AnalyseAgent1_beh.registerDefaultTransition
 ("demandediagnostic","attendreinstantiation");
addBehaviour(AnalyseAgent1_beh);
}
 private class attendreinstantiation extends
 OneShotBehaviour{
     Int valeurRetour = 0;
 public void action ()   {
 ACLMessage message = newACLMessage (ACLMessage.INFORM);
 Message.addReceiver ( new AID ("InterfaceAgent",
 AID.ISLOCALNAME));
           If (!stop)  {
                 Message.setContent("pret");
                 Send (message);
                 valeurRetour = 0;
                 block ();
           } else{
                 Message.setContent("arret");
                 Send (message);
                 valeurRetour = 1;
           } }
 public  int  onEnd ()  {
     Return  valeurRetour;
 } }
...
```

5 Comparaison avec d'autres travaux

Il existe dans la littérature plusieurs travaux concernant l'utilisation du paradigme multi agents dans la résolution du problème du diagnostic, et ce dans différents domaines allant du contrôle et du diagnostic des systèmes industriels complexes jusqu'à la détection des maladies et leur diagnostic en médecine.

Dans ce qui suit, nous allons comparer le modèle proposé avec des systèmes existants en l'occurrence : MAGIC *(Multi-Agent-based Diagnostic Data Acquisition and Management in Complex Systems)* [10], DIAMOND *(DIstributed Architecture for MONitoring and Diagnosis)* [17] et CMDS *(Contract Net Based Medical Diagnosis System)* [9] en se basant sur des critères que nous jugeons essentiels dans un système multi agents pour le diagnostic des systèmes complexes.

Table 1. Tableau Comparatif.

	MAGIC	DIAMOND	CMDS	NOTRE MODELE
Distinction entre les étapes de détection et d'analyse (localisation)	oui	oui	non	oui
Etape de Détection	Distribuée	Distribuée	X	Distribuée
Etape d'Analyse	Centralisée	Distribuée	X	Distribuée
Coopération entre agents pour la prise de décision collective	non	non	oui	Oui
Domaine d'application	industriel	industriel	médical	Industriel et médical

- *Distinction entre les étapes de détection et d'analyse :* L'idée derrière cette distinction est de pouvoir utiliser les méthodes les plus appropriées pour chacune des deux étapes de la procédure de diagnostic. En effet, les méthodes FDI sont plus efficaces pour la détection alors que les méthodes DX le sont pour la localisation. Alors que les systèmes MAGIC et DIAMOND distinguent ces deux étapes en leurs associant deux types différents d'agents, le système CMDS ne fait aucune distinction entre les étapes de diagnostic. Comme le système MAGIC, le modèle que nous proposons distingue clairement la détection de l'analyse (localisation) et utilise conjointement les méthodes FDI pour la détection et les méthodes DX pour l'analyse.

- *L'étape de détection* est parfaitement distribuée sur des agents dédiés appelés agents de diagnostic dans MAGIC et agents de contrôle dans DIAMOND. Elle est également distribuée dans notre modèle sur différents agents appelés agents de détection.

- *L'étape d'analyse :* dans MAGIC la localisation est centralisée dans l'agent de prise de décision diagnostic alors que dans DIAMOND elle est distribuée sur différents agents de diagnostic. Dans notre modèle, elle est également distribuée sur différents agents d'analyse.

- *Coopération entre agents pour la prise de décision collective :* dans MAGIC la prise de décision est effectuée par un seul agent alors que dans DIAMOND elle est distribuée sur plusieurs agents de diagnostic mais sans qu'il y est de coopération réelle pour l'établissement du diagnostic global. Pour le système CMDS, l'algorithme « Cooperative Medical Diagnosis Elaboration » [9] décrit une résolution coopérative du problème par des agents médicaux. Dans notre modèle, un agent d'analyse doit interagir avec d'autres agents du même type pour pouvoir calculer d'une manière coopérative son propre diagnostic local (construction de l'arbre HS-Tree).

- *Domaine d'application :* les systèmes MAGIC et DIAMOND sont dédiés au diagnostic des systèmes industriels alors que CMDS est réservé au diagnostic médial. Le modèle que nous proposons peut être utilisé aussi bien dans le domaine industriel que médical. Même si le raisonnement à base de modèle s'apprête mieux au domaine industriel, l'analyse diagnostic des agents peut être renforcée par un raisonnement abductif (voir perspectives) ce qui rendra le modèle encore plus approprié pour le diagnostic médical.

6 Conclusion et Perspectives

Dans ce papier nous avons proposé un modèle SMA pour le diagnostic collectif, où nous avons mis l'accent d'abord sur l'isolation des différentes étapes du diagnostic notamment la détection et l'analyse (localisation) qui constituent les deux étapes les plus importantes dans une procédure de diagnostic, en effet, notre modèle offre la possibilité d'utiliser conjointement les méthodes issues de la communauté FDI pour l'étape de détection et les méthodes DX pour l'analyse (raisonnement), puis nous nous sommes intéressés plus particulièrement à la distribution de l'étape d'analyse, pour cela nous avons adapté la théorie de Reiter à notre contexte de travail en modifiant la manière de générer l'arbre « HS-Tree » par différents agents d'analyse, en effet, dans le modèle proposé, le raisonnement diagnostic est totalement distribué à travers plusieurs agents d'analyse selon une distribution sémantique des connaissances. Ainsi, chaque agent d'analyse aura comme tâche de diagnostiquer un sous ensemble de composants du système et ce d'une manière autonome tout en gardant la possibilité d'avoir des agents d'analyse avec des connaissances dépendantes, ce qui implique donc une coopération des agents pour le calcul des diagnostics locaux. Le diagnostic global est calculé en fusionnant les différents diagnostics locaux.

Afin de valider notre modèle, nous avons testé son applicabilité sur un bioprocédé et simuler les interactions de ses agents sur la plateforme JADE.

Renforcer l'analyse diagnostic par un raisonnement abductif et doter les agents d'une capacité à apprendre selon des mécanismes d'apprentissage flou feront l'objet de nos futurs travaux.

References

1. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence, vol. 32, n°. 1, pp. 57--96 (1987).
2. Davis, R.: Diagnostic Reasoning based on structure and behaviour, Artificial Intelligence, vol. 24, pp. 347--410 (1984).
3. De Kleer, J., Williams, B. C.: Diagnosing multiple faults. Artificial Intelligence, vol. 32, pp. 97--130 (1987).
4. Cordier, M.O., Dague, P., Lévy, F., Montmain, J., Staroswiecki, M.,Travémassuyès, L.: Conflicts Versus Analytical Redundancy Relations: A comparative Analysis of the Model Based Diagnosis Approach From the Artificial Intelligence and Automatic Control Perspectives. Special Issue of the IEEE Transactions on Systems, Man, and Cybernetics, vol. 34, n°.5, pp. 2163-- 2177, October (2004).
5. Philippot, A., Sayed Mouchaweh, M., Carré-Ménétrier, V., Riera, B.: Decentralized Approach to Diagnose Manufacturing Systems, In Computational Engineering in Systems Applications CESA'06, Beijing, China (2006).
6. Wang, H., Zhang, M., Xu, D., Zhang, D.: A Framework of Fuzzy Diagnosis, IEEE Transactions on knowledge and data engineering, vol. 16, n°. 12, December (2004)
7. Lopez Varéla, C., Subias, A., Combacau, M. : Approche de détection basée cohérence : modèles pour le diagnostic, 3e Colloque International Francophone Performance et Nouvelles Technologies en Maintenance (2007).

8. Mokhtari, A., Le Lann, M.V., Hetreux, G., Le Lann, J.M : Diagnostic à base de modèle des systèmes dynamiques hybrides, SIMO 06 Systèmes d'Information, Modélisation, Optimisation et Commande en génie des procédés, 11-12 octobre, Toulouse (2006).
9. Iantovics, B.L.: Cooperative Medical Diagnoses Elaboration by Physicians and Artificial Agents, Petru Maior University of Tg. Mures, Romania (2008).
10. Köppen-Seliger, B., Marcu, T., Capobianco, M., Gentil, S., Albert, M., Latzel, S.: MAGIC: An integrated approach for diagnostic data management and operator support, IFAC SAFEPROCESS'03, Washington (2003).
11. Touaf, S., Ploix, S., Flaus, J.M.: A Logical Diagnostic Method For Complex Dynamic Systems, 5ème Congrès International Pluridisciplinaire Qualité et Sûreté de Fonctionnement, Qualita 2003, Nancy, France (2003).
12. Touaf, S.: Logical Diagnosis of the Complex and Dynamic Systems in Multi-Agent Context, PhD Thesis, Joseph Fourier- Grenoble 1 University, France (2005).
13. Frank, M. P., Köppen-Seliger, B.: New Developments Using AI in Fault Diagnosis, Engng Applic. Artif. Intell., vol. 10, n°.1, pp. 3--14 (1997).
14. Isermann, R.: Supervision, Fault-Detection and Fault- Diagnosis Methods - An Introduction, *Control Eng.Practice*, vol.5, pp. 639--652 (1997).
15. Chantler, M. J., Coghill, G. M., Shen, Q., Leitch, R. R.: Selecting tools and techniques for model-based diagnosis, *Artificial Intelligence in Engineering*, n°.12, pp. 81--98 (1998).
16. Venkatasubramanian, V., Rengaswamy, R., Kavuri, N., Yin K.: A review of process fault detection and diagnosis Part I: Quantitative model-based methods, Computers and Chemical Engineering, vol. 27, pp. 293-311 (2003).
17. Albert, M., Langle, T., Worn H.: Development Tool for Distributed Monitoring and Diagnosis Systems, *Institute for Process Control and Robotics*, University of Karlsruhe, Germany (2002).

Optimisation d'Alignement d'une Ontologie Multi-Points de Vue et une Ontologie Classique

Lynda Djakhdjakha[1], Mounir Hemam[2],
[1]Centre Universitaire de Souk Ahrass, Algérie
[2]Centre Universitaire de Khenchela, Algérie
[1]ldjakhdjakha@yahoo.fr
[2] Mounir.hemam@gmail.com

Résumé. Dans ce papier nous proposons d'utiliser les métaheuristiques pour optimiser le résultat d'un alignement entre une ontologie multi-points de vue et une ontologie classique. L'idée est de représenter les ontologies à comparer par des graphes et de chercher du meilleur appariement parmi les appariements multivoques des nœuds des graphes, où chaque nœud du graphe de l'ontologie classique peut être apparié à zéro, à un ou à plusieurs nœuds du graphe de l'ontologie multi-points de vue dans laquelle chaque nœud appartient à un point de vue différent.

Mots clés : Optimisation, alignement, ontologie, appariement graphes, colonies de fourmis, points de vue.

1 Introduction

A l'origine, la plupart des métaheuristiques ont été développées pour résoudre les problèmes d'optimisation combinatoire mais de nos jours, beaucoup de recherches ont pris comme objet l'adaptation de ces méthodes aux autres types de problèmes telle que le problème d'alignement d'ontologies.

L'*alignement des ontologies* ayant pour objectif de permettre une utilisation conjointe de plusieurs ontologies. Le résultat de cette tâche assure et facilite l'échange, le partage, la fusion des données et des informations entre systèmes ou des communautés dans le Web sémantique. Il s'agit généralement de construire des *appariements* entre les éléments décrits dans différentes ontologies. Dans la littérature, plusieurs méthodes d'alignement d'ontologies ont été proposées. Elles tirent parti des différents aspects des ontologies. Et elles s'intéressent à l'alignement des ontologies décrites dans différents langages ontologiques. En 2007, et dans le même contexte, une méthode d'alignement EDOLA a été proposée dans [1]. Cette méthode utilise l'algorithme de recherche tabou pour aboutir à un alignement optimal. Par conséquent, la majorité de des méthodes d'alignement permettent de détecter seulement des relations entre des ontologies classiques qui ne prennent pas en compte la notion de multiples points de vue.

Dans ce travail, nous nous intéressons au problème de développement d'ontologies dans une organisation hétérogène en prenant en compte différents points de vue, différentes terminologies des personnes, des groupes voire des communautés diverses au sein de cette organisation. Une telle ontologie, appelée *ontologie multi-points de vue*, permet de faire cohabiter à la fois l'hétérogénéité et le consensus dans une

organisation hétérogène. A la différence d'une ontologie classique, une ontologie multi-points de vue confère à un même univers de discours plusieurs représentations différentes telles que chacune est relative à un point de vue particulier [2]. Ce besoin de prise en compte de connaissances multi-points de vue, au sein d'une même ontologie, provient essentiellement d'un environnement multidisciplinaire où plusieurs groupes de personnes diversifiés coexistent et collaborent entre eux. Chaque groupe a ses intérêts particuliers et perçoit différemment les propriétés et les relations particulières des entités conceptuelles du même univers de connaissances à représenter.

L'objectif de cet article est double, nous proposons, dans un premier temps, une approche d'alignement entre une ontologie multi-points de vue et une ontologie classique. Ensuite, nous essayons d'optimiser le résultat d'alignement. De ce fait, nous proposons de prendre en entrée deux ontologies décrites en logiques de description étendues par le mécanisme d'estampillage et de les transformer en des structures de graphes, en particulier, en deux *graphes étiquetés et attribués*. Les graphes sont considérés comme d'excellents outils permettant la représentation de données structurées. Ainsi, l'idée est d'appliquer un modèle de calcul de similarité entre les deux ontologies qui se réduit à la comparaison des deux graphes. D'un point de vue mathématique, le calcul de similarité entre deux graphes est réalisé par une recherche de morphisme de graphes. Ce genre de problème s'appelle *appariement de graphes* et est un problème NP-complet.

Dans le processus d'alignement, nous nous intéressons à la recherche du *meilleur appariement* parmi les *appariements multivoques* des nœuds des graphes, où chaque nœud du graphe de l'ontologie classique peut être apparié à zéro, à un ou à plusieurs nœuds du graphe de l'ontologie multi-points de vue dans laquelle chaque nœud appartient à un point de vue différent. Cette recherche peut se traduire en un problème de *sélection de sous-ensemble (SS-problème)*, dans le but de trouver un sous-ensemble qui *satisfait certaines propriétés*. Pour résoudre ce problème, nous utilisons une méthode d'optimisation par métaheuristique. Les métaheuristiques sont des méthodes approchées qui traitent les problèmes d'optimisation difficile. Le but d'un problème d'optimisation est de trouver une solution maximisant ou minimisant une fonction objectif donnée. Ainsi, les métaheuristiques constituent une classe de méthodes approchées adaptables à un grand nombre de problèmes d'optimisation combinatoire.

Nous proposons d'utiliser les algorithmes incomplets *d'optimisation par les colonies de fourmis* qui est une métaheuristique récente qui s'inspire de l'intelligence collective des fourmis. Dans notre travail l'algorithme d'optimisation par colonies de fourmis est paramétré par un ensemble de caractéristiques pour être capable de retourner le meilleur sous-ensemble de l'ontologie multi-points de vue qui est apparié à l'ontologie classique.

L'article est organisé comme suit. Dans la section 2, nous décrivons l'algorithme d'optimisation par colonies de fourmis. Dans la section 3, nous clarifions le problème de sélection de sous-ensemble. Dans la section 4, nous détaillons la notion d'une ontologie décrite en logique de description étendue par le mécanisme d'estampillage. Dans la section 5, nous présentons le processus d'alignement d'une ontologie multi-points de vue et une ontologie classique et incluons l'étape d'optimisation d'alignement. Dans la section 6, nous concluons et donnons quelques perspectives pour améliorer notre travail.

2 L'Algorithme d'Optimisation par Colonies de Fourmis

L'algorithme d'optimisation par colonies de fourmis est inspiré du comportement des fourmis à la recherche de nourriture, et a été mis au point par Dorigo en 1992 [3]. Son principe repose sur le comportement particuliers des fourmis qui, elles sont capable de déterminer le chemin le plus court entre leur nid et une source de nourriture grâce à la phéromone qui est une substance que les fourmis déposent sur le sol lorsqu'elles se déplacent. Lorsqu'une fourmi doit choisir entre deux directions, elle choisit avec une plus grande probabilité celle comportant une plus forte concentration de phéromone. Le principe de l'algorithme consiste à reformuler le problème à résoudre en un problème de recherche d'un meilleur chemin dans un graphe appelé graphe de construction et à utiliser des fourmis artificielles pour trouver les bons chemins de ce graphe. A chaque cycle de l'algorithme, chaque fourmi de la colonie construit aléatoirement un chemin du graphe et la quantité de la phéromone est déposée sur les meilleurs chemins découverts lors de ce cycle. Lors des cycles suivants, les fourmis construisent de nouveaux chemins avec une probabilité dépendant de la phéromone déposée lors des cycles précédents et d'une heuristique propre au problème considéré. La colonie de fourmis converge alors peu à peu vers les meilleures solutions.

3 Problème de Sélection de Sous-Ensemble

Les problèmes de sélection de sous-ensembles (SS-problème) ont pour but de trouver un sous-ensemble consistant et optimal d'objets. Plus formellement, un SS-problème est défini par un triplet (S, $S_{consistant}$, f) tel que :

- S est l'ensemble d'objet ;
- $S_{consistant} \subseteq P(S)$ est l'ensemble de tous les sous-ensembles de S qui sont consistants, afin de pouvoir construire chaque sous-ensemble de $S_{consistant}$ de façon incrémentale (en partant de l'ensemble vide et en ajoutant à chaque itération un objet choisi parmi l'ensemble des objets consistants avec l'ensemble en cours de construction), on impose la contrainte suivante : pour chaque sous-ensemble consistant non vide $S' \in S_{consistant}$, il doit exister au moins un objet $o_i \in S'$ tel que $S' - \{o_i\}$ est aussi consistant.
- $f : S_{consistant} \rightarrow IR$ est la fonction objectif qui associe un coût $f(S')$ à chaque sous-ensemble d'objets consistant $S' \in S_{consistant}$.

Le but d'un SS-problème (S, $S_{consistant}$, f) est de trouver $S^* \in S_{consistant}$ tel que $f(S^*)$ soit maximal.

4 Ontologie Multi-Points de Vue en Logique de Description Etendue par le Mécanisme d'Estampillage

Le mécanisme d'estampillage consiste à utiliser un index de point de vue PV_i comme préfixe aux expressions du langage local du point de vue PV_i pour délimiter les éléments de connaissances (i.e. concepts, rôles et individus) définis au sein de ce point de vue. Par ailleurs, une ontologie, décrite selon le modèle multi-points de vue proposé dans [2], [4] et [5], est constituée de trois principales parties : **1)** la partie

terminologie (TBox) qui regroupe l'ensemble les concepts globaux, les concepts locaux et les rôles locaux des différents points de vue. Un concept global est un concept vu à partir de deux ou plusieurs points de vue avec certaines caractéristiques communes (i.e. attributs et/ou relations). Un concept local est un concept qui est vu et décrit localement selon un point de vue donné. Un rôle local permet de représenter le rapport entre deux concepts d'un même point de vue **2)** la partie assertions (ABox) qui contient un ensemble d'assertions et de faits sur les individus. Un individu est une instance d'un concept défini selon un point de vue donné et pouvant avoir des rôles (liens) avec d'autres individus du même point de vue ou bien avec des individus d'autres points de vue **3)** la partie liens intermédiaires (*LI*) qui regroupe l'ensemble des rôles globaux et des passerelles. Un rôle global permet de relier deux concepts de deux points de vue différents et permet ainsi d'exprimer un fait général à propos des membres des concepts qui participent à ce rôle. Par ailleurs, une passerelle décrit une règle entre les concepts de deux ou plusieurs points de vue différents.

5 L'Approche Proposée

L'originalité de notre approche réside dans le fait d'optimiser le résultat d'alignement d'une ontologie multi-points de vue et une ontologie classique en utilisant un algorithme d'optimisation par colonies de fourmis. L'objectif donc est de mettre en correspondances les deux ontologies représentées par des graphes et d'optimiser par la suite ces correspondances afin d'obtenir un alignement optimal et plus similaire. Elle se compose de trois phases successives : le pré-alignement, le processus d'alignement et la phase de post-alignement et d'optimisation (voir la Figure 1).

5.1Pré-Alignement

Dans la phase de pré-alignement, nous identifions d'abord les informations d'entrée (Input) qui constituent essentiellement les structures destinées à être alignées. Puis nous formalisons ces structures sous forme de graphe.

Format en entrée. Les deux ontologies en entrée sont décrites en logiques de description étendues par le mécanisme d'estampillage décrit précédemment.

Intervention humaine. Du fait qu'il est impossible d'automatiser complètement le processus d'alignement, une intervention de l'expert du domaine est absolument nécessaire pour effectuer les opérations suivantes :

Adaptation de l'ontologie classique. Un expert du domaine est sollicité pour identifier le nouveau point de vue correspondant à l'ontologie classique, les concepts globaux et les concepts locaux. Le préfixe PV_i est rajouté à chaque axiome de sa partie terminologique (TBox) et à chaque assertion de sa partie assertionnelle (ABox), pour avoir une ontologie *mono-point de vue* (i.e., un seul point de vue à considérer). La partie LI de l'ontologie mono-points de vue est vide, puisqu'il n'a y pas de relations globales ni de passerelles dans une ontologie mono-point de vue.

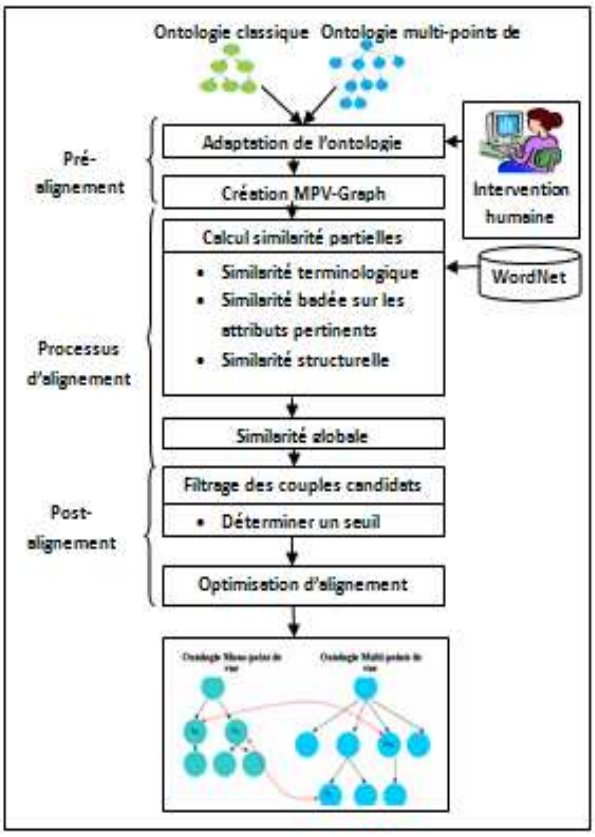


Fig. 1. Processus d'Alignement d'une Ontologie Multi-Points de Vue et une Ontologie Classique

Attribution et pondération des concepts locaux. Cette étape consiste à sélectionner pour chaque concept local les attributs les plus pertinents par rapport au points de vue visé, cette sélection peut se faire en évaluant l'importance des attributs un par un dans ce point de vue, puis trouver un vecteur réel de poids $W=(w_1, \ldots, w_n)$. Ce vecteur est utilisé par la suite dans le module de calcul de similarité. Le but de cette étape est d'améliorer les correspondances entre les concepts locaux.

Création du MPV-Graph. Cette étape consiste à prendre les deux ontologies (mono et multi-points de vue) en entrée et les convertir en de deux graphes étiquetés et attribués appelés respectivement MPV-Graph_1, MPV-Graph_2. Ces deux graphes permettent de représenter toutes les informations contenues dans les deux ontologies

(i.e., points de vue, concepts, rôles locaux et individus) et aussi les autres informations concernant l'ontologie multi-points de vue (i.e., rôles globaux et passerelles).

Les nœuds de chaque graphe représentent les trois types suivants : les concepts locaux, les concepts globaux et les individus. Les arcs qui existent dans MPV-Graph$_1$ représentent « les rôles locaux », et « les relations d'instanciation ». Par ailleurs, les arcs qui existent dans MPV-Graph$_2$ représentent en plus « les rôles globaux » et « les passerelles ». La création du MPV-Graph est réalisée en trois étapes :

La transformation de TBox **.** Chaque axiome est décomposé en un ensemble de concepts, de rôles et de points de vue apparaissant dans cet axiome, chaque concept est représenté par un nœud, il existe un arc de type (rôle local) entre deux nœuds si leurs concepts sont dans le même axiome.

La transformation d'ABox. Chaque assertion est décomposée en un ensemble d'individus, de relations, et de points de vue apparaissant dans cette assertion, chaque individu est représenté par un nœud, il existe un arc de type « rôle local » entre deux nœuds si leurs individus sont dans la même assertion.

Par ailleurs, un arc de type « instanciation » est créé entre deux nœuds si l'un des deux nœuds est de type « concept » et l'autre est de type « individu ».

La transformation de LI[1]. Pour chaque entrée de l'ensemble LI, on détermine l'ensemble de concepts, l'ensemble d'individus et l'ensemble de points de vue. en suite, un ou plusieurs arcs de types « rôles globaux ou passerelles » sont créés entre deux ou plusieurs nœuds si leurs concepts (individus) sont dans la même entrée.

Etiquetage du graphe. Dans cette étape nous nous inspirons de l'idée proposée dans [6] pour étiqueter les nœuds et les arcs des deux graphes. Chaque nœud du *MPV-Graph* est étiqueté par un préfixe PV_i qui indique le nom du point de vue de ce nœud plus le nom d'un type (concept local, concept global ou individu), en ajoutant aux nœuds de type concept les étiquettes suivantes selon le cas :

- Un concept primitif est étiqueté par son nom.
- Un concept défini par une conjonction, une disjonction ou bien une négation est étiqueté par son nom et par l'opérateur logique utilisé (and, or, ou not).
- Un concept défini par un rôle est représenté par un arc qui porte le nom du rôle et pointe vers un autre concept.

Chaque arc, qui relie deux nœuds, est étiqueté par le type de la relation (rôle local, rôle global ou une passerelle) et le nom de cette relation.

Dans le cas d'un concept défini, l'arc est étiqueté soit par le quantificateur universel ($\forall$) soit par le quantificateur existentiel ($\exists$) ou bien par les symboles '$\leq, \geq$' et le nombre n et le rôle R dans le cas de la restriction.

[1] La partie LI de l'ontologie mono-point de vue est vide.

5.2 Processus d'Alignement

Le processus d'alignement se déroulera en quatre étapes, il consiste à : calculer la similarité entre tous les éléments de deux graphes, pour chaque couple de nœuds, on calculera d'abord la similarité terminologique, ensuite la similarité basée sur les attributs pertinents, puis calculer la similarité structurelle entre eux et utiliser les valeurs de ces trois similarités pour calculer la valeur de similarité globale entre ces couples de nœuds.

Similarité terminologique. Cette étape prend en entrée les deux graphes et calcule la similarité entre les noms de différents nœuds de MPV-Graph$_s$ (i.e., concepts, individus et attributs).

Cette similarité est calculée soit en utilisant une fonction de similarité syntaxique comme la distance de Levenshtein qui est une distance normalisée, qui a prouvé sa robustesse face aux fautes d'orthographes, soit en utilisant une fonction de similarité lexicale en exploitant l'API de WordNet [7]. L'idée d'utiliser cette API est de résoudre les problèmes de conflit de terme (e.g., les termes humain et personne).

Le calcul de similarité entre les noms d'attributs nécessite une normalisation manuelle. Le but est de maximiser le nombre d'attributs communs entre les différents concepts. De ce fait, nous essayons de simplifier les chaînes de caractères représentant les attributs pour les ramener à des formats équivalents par un ensemble d'opérations (i.e., retirer les diacritiques, normaliser le nombre d'espaces, supprimer les noms de liaisons, extension des abréviations, etc.).

Le résultat de cette étape est une matrice représentant la similarité terminologique entre les nœuds des MPV-Graph$_s$ qui est une somme pondérée des deux types de similarités précédente, dont les lignes représentent les nœuds du MPV-Graph$_1$ et les colonnes représentent les nœuds du MPV-Graph$_2$. Cette matrice est utilisée plus tard dans les autres modules de mise en correspondance.

Mesure basée sur les attributs pertinents. Dans le cas de multi-points de vue, la pondération d'attributs des concepts locaux est une tâche cruciale pour que la mesure de similarité soit pertinente et la plus précise possible. De ce fait, nous proposons d'utiliser une distance entre les attributs pondérés, pour les couples de nœuds de type (concept local, concept local). La mesure est calculée en utilisant une matrice d'attributs des deux concepts en question avec leurs poids et leurs valeurs de similarité terminologique déjà calculée. Dans cette matrice nous définissons un seuil pour classifier l'ensemble d'attributs communs entre les deux concepts. Nous nous inspirons pour cela de la mesure proposée dans [8] pour calculer la similarité entre deux concepts :

$$Sim_{att}(C1, C2) = \frac{\sum_{i,j}^{n} WiW'j * Sim_{ss}(Att_i, Att_j)}{\sum_{i,j}^{m} W_i W'_j} \quad . \qquad (1)$$

Où : n est le nombre d'attributs commun selon le seuil proposé, m=i*j est le nombre total des couples dans la matrice, $Sim_{ss}(Att_i, Att_j)$ est la similarité supérieure au seuil défini, $\sum_{i,j}^{n}$ WiW'j: la somme des poids des attributs commun,

$\sum_{ij}^{m}$ $W_iW'_j$: la somme des poids de tous les attributs (utilisée pour normaliser la mesure).

Similarité structurelle. Elle est calculée en exploitant la matrice de similarité terminologique ainsi que le voisinage et la hiérarchie des concepts en question dans les MPV-Graph$_s$. Dans notre modèle de représentation, nous trouvons trois types de voisinage qui dépendent de la position et de type des nœuds dans les MPV-Graph$_s$:

Racine globale. d'un nœud de type concept local ;
Racine locale. d'un nœud de type individus ;

Pour ces deux types, la similarité est calculée en extrayant le chemin racine possédant le nœud en question et la racine globale (locale), puis nous calculons la distance entre les deux nœuds en utilisant la mesure de Wu et Palmer [9] ;

Voisinage par point de vue. Dans cette section nous exploitons la structure de voisinage par point de vue et nous calculons la mesure de similarité sémantique entre deux ensembles de nœuds voisins, les ensembles sont organisés par point de vue. Pour calculer cette mesure, nous utilisons la mesure proposée dans [10] qui calcule la similarité entre deux ensembles. ENS_{PVi}, ENS_{pvj} représentent deux ensembles de nœuds. Les entités de chaque ensemble sont étiquetées par le même point de vue. La similarité sémantique entre deux nœuds appartenant à deux points de vue différents se calcule alors par la fonction suivante :

$$Sim_{vois}(N1, N2) = \frac{\sum_{(i,i') \in paires(ENSpvi,\ ENSpvj)} SIM_{term}(i,i')}{Max\left(\left|ENS_{pvi}\right|, \left|ENS_{pvj}\right|\right)} \quad . \qquad (2)$$

Où (i, i') un couple de ENS_{PVi}* ENS_{pvj} , $SIM_{term}(i,i')$: la similarité terminologique déjà calculé.

La similarité structurelle Sim_{struc} est calculée en employant la technique de la somme pondérée des trois types précédents suivant le cas de type des nœuds à comparer.

Similarité globale. La valeur de similarité globale est calculée en employant la méthode de la somme pondérée des trois valeurs de similarité partielles (terminologique, basée sur les attributs pondérés et la similarité structurelle) :

$$Sim_{glob}(C1, C2) = \mu_{term}*Sim_{term} + \mu_{att}*Sim_{att} + \mu_{struc}*Sim_{struc} \quad . \qquad (3)$$

La similarité partielle Sim_{att} est égale à zéro si l'un des deux nœuds à comparer ou bien les deux sont de type individus (concept global).

Le résultat de cette phase est une matrice de similarité globale, qui contient tous les couples d'entités comparées avec une valeur de similarité compris entre [0, 1].

5.3Post-Alignement

Filtrage des couples candidats à base de seuil. Cette étape est la première à effectuer dans la phase de post-alignement. Pour définir les nœuds les plus similaires nous déterminons une valeur de seuil située entre 0 et 1 dans ce cas. Après le calcul de similarité entre chaque couple de nœuds provenant de deux ontologies en entrée, on a une matrice contenant ces valeurs de similarité. Le filtrage consiste à éliminer les couples dont la valeur de similarité est inférieure à ce seuil. Nous obtenons donc une matrice des couples les plus similaires. Où nous pouvons trouver qu'un nœud du MPV-Graph$_1$ peut apparier à zéro, un ou plusieurs nœuds du MPV-Graph$_2$ (i.e., appariement multivoque).

Optimisation d'alignement. Afin de trouver les meilleures correspondances entre les deux ontologies, nous formalisons notre problème sous forme d'un problème de sélection de sous-ensemble et nous adaptons l'algorithme d'optimisation par colonies de fourmis (ACO) proposé dans [11] pour ce type de problème.

Bien que l'algorithme de colonie de fourmis est conçu au départ pour le problème du voyageur de commerce, il offre finalement beaucoup de souplesse, et il a été possible de l'adapter à un grand nombre de problèmes combinatoire (appariement de graphes).

*Formalisation du problème***.** Dans notre contexte, le problème d'alignement d'une ontologie classique avec une ontologie multi-points de vue, est un problème *d'appariements multivoques* de deux graphes *MPV-Graph*$_s$, ce problème se ramène au problème de sélection de sous-ensemble [11], et nous pouvons le formaliser par le triplet $(S, S_{consistant}, f)$ tel que :

- S contient l'ensemble des couples appariant un nœud de *MPV-Graph*$_1$ à un nœud de *MPV-Graph*$_2$ « la matrice de similarité globale »;
- $S_{consistant} = P(S)$ contient tous les sous-ensembles de S et aussi les appariements multivoques ;
- f est définie par la fonction score, la fonction score est définie dans [10] par la formule :

$$score(S') = f(MPV\text{-}Graph_1 \cap_S MPV\text{-}Graph_2) - g(splits(S')). \quad (4)$$

Où splits(S') est l'ensemble des nœuds qui sont appariés à plus d'un nœud. Donc, le résultat est un sous-ensemble consistant de couples de nœuds $S' \in S_{consistant}$, tel que la fonction score soit maximal.

Description de l'algorithme. A chaque cycle de l'algorithme, chacune des fourmis construit un sous-ensemble. En partant d'un sous-ensemble S' vide, les fourmis ajoutent à chaque itération un couple de nœuds de la matrice de similarité globale à S' choisi parmi l'ensemble des couples non encore sélectionnés. Le couple de nœuds à ajouter à S' est choisi selon une probabilité qui dépend des traces des phéromones et deux facteurs heuristiques, l'un vise à favoriser les couples qui ont la similarité la plus forte et l'autre vise à favoriser les couples qui font le plus augmenter la fonction score.

Une fois que chaque fourmi a construit son sous-ensemble, une procédure de recherche locale est lancée afin d'essayer d'améliorer la qualité du meilleur sous-

ensemble trouvé lors de ce cycle. Les traces de phéromone sont par la suite mises à jour en fonction de ce sous-ensemble amélioré.

Les fourmis arrêtent leur construction quand tous les couples de nœuds candidats font décroitre le score de sous-ensemble ou quand les trois derniers ajouts n'ont pas permis d'accroître se score.

Construction d'une solution par une fourmi. Le code suivant décrit la procédure suivie par les fourmis pour construire un sous-ensemble.

Procedure construire sous-ensemble
Entrée : un SS-problème (S,$S_{consistant}$, f)et une fonction heuristique associée : S*P(S)→IR$^+$;
Une stratégie phéromonale et un facteur phéromonal τ, et deux facteurs heuristiques $\varphi 1, \varphi 2$.
Deux paramètres à valeurs numériques : α, β_1, β_2
Sortie : un sous-ensemble consistant d'objets S'$\in S_{consistant}$
Initialiser les traces de phéromone à τ_{max}
Répéter
Pour chaque fourmi k dans 1..$nbFourmis$,
construire une solution S_k comme suit :
Choisir aléatoirement le premier nœud $o_i \in S$
$S_k \leftarrow \{o_i\}$, $Candidat \leftarrow \{o_j \in S \mid S_k \cup \{o_j\} \in S_{consistant}\}$
Tant que $Candidats \neq \emptyset$ **faire**
Choisir $o_i \in$ Candidats avec la probabilité

$$P_{o_i} = \frac{\left[\tau_{facteur}(o_i, S_k)\right]^{\alpha} \cdot \left[\varphi 1_{facteur}(o_i, S_k)\right]^{\beta_1} \cdot \left[\varphi 2_{facteur}(o_i, S_k)\right]^{\beta_2}}{\sum_{o_j \in Candidat} \left[\tau_{facteur}(o_j, S_k)\right]^{\alpha} \cdot \left[\varphi 1_{facteur}(o_j, S_k)\right]^{\beta_1} \cdot \left[\varphi 2_{facteur}(o_j, S_k)\right]^{\beta_2}}$$

$S_k \leftarrow S_k \cup \{o_i\}$
Enlever de Candidats chaque couple qui viole les contraintes.
Fin tant que
Fin pour
Mettre à jour les traces de phéromone en fonction de $\{S_1, \ldots, S_{nbfourmis}\}$
Si une trace de phéromone est inférieure à τ_{min} **alors** la mettre à τ_{min}
Si une trace de phéromone est supérieure à τ_{max} **alors** la mettre à τ_{max}
Jusqu'à nombre maximal de cycles atteint ou solution trouvée.

Stratégie phéromonale. Le choix d'une stratégie phéromonale est un point clé lors du développement d'un algorithme à base de colonies de fourmis[2]. Deux stratégies sont distinguées : « Vertex » mémorise l'expérience concernant chaque couple de nœuds individuellement et donc l'expérience de la colonie concernant l'intérêt d'apparier un nœud du premier graphe à un autre nœud du deuxième graphe. Et la stratégie « Edge » qui mémorise l'expérience concernant chaque couples de nœuds et donc l'expérience d'apparier deux arcs entre eux lors de la construction de nouveaux appariements.

Les résultats expérimentaux présentés dans [10] sur les problèmes d'appariement de graphes ont montré que la stratégie « Vertex » donne de meilleurs résultats que la stratégie « Edge ». ainsi la stratégie « Vertex » est plus rapide que la stratégie « Edge ».

La procédure de Recherche locale. Elle est appliquée sur le meilleur sous-ensemble trouvé lors de chaque cycle. Et permet d'offrir un bon compromis entre la quantité des solutions trouvées et son temps d'exécution. La procédure permet à chaque itération de sélectionner le couple de nœuds qui essaye d'améliorer la solution en explorant le voisinage. Les voisins d'un sous-ensemble sont les couples qui peuvent être obtenus en ajoutant ou en supprimant un couple de nœuds à ce sous-ensemble.

Résultat de l'optimisation. Le résultat de l'algorithme d'optimisation est un graphe de construction sur lequel les fourmis déposent de la phéromone. Il est construit à partir de deux MPV-Graph$_s$. Les nœuds de ce graphe sont les combinaisons de deux nœuds, l'un du graphe de l'ontologie mono-point de vue et l'autre du graphe de l'ontologie multi-points de vue. Ces combinaisons représentent la meilleure solution que les fourmis peuvent sélectionner lors de la construction de leurs solutions. Cette solution représente les meilleures correspondances entre les deux ontologies.

Résultat de l'alignement. A partir du graphe de construction, nous déduisons une représentation graphique de l'alignement de deux MPV-Graph$_s$, elle est composée des représentations des deux graphes et d'un ensemble de correspondances entre des nœuds issus respectivement des deux graphes.

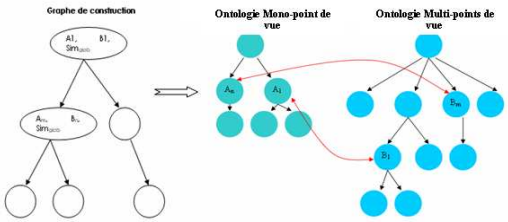


Fig. 2. Résultat de l'alignement

[2] Une stratégie phéromonale consiste à décider où les traces de phéromone doivent être déposées et comment elles doivent être exploitées et mises à jour.

6 Conclusion

Dans ce papier, nous avons adapté l'algorithme d'optimisation par les colonies de fourmis pour optimiser le résultat d'un alignement d'une ontologie multi-points de vue et une ontologie classique.

De ce fait, nous avons tout d'abord proposé une approche d'alignement d'une ontologie classique avec une ontologie multi-points de vue. L'approche proposée se compose de trois phases principales : (i) la phase de pré-alignement qui permet d'identifier les formats des ontologies en entrée de les adapter et de les transformer en des structures de graphes afin de résoudre le problème d'hétérogénéité de représentation. (ii) Le processus d'alignement qui combine les techniques et les méthodes d'appariement syntaxique, linguistiques, basé sur les attributs pertinents selon les différents points de vue et les techniques structurelles. (iii) La phase de post-alignement : dans cette phase, l'optimisation de l'appariement est effectuée par une technique de méta-heuristique de colonies de fourmis.

Dans l'approche proposée l'ordre des ontologies en entrée a une grande importance. Par ailleurs, l'hybridation des ACO/ recherche locale dans notre adaptation, nous permis de trouver un compromis entre le temps et la qualité d'alignement.

Références

1. Zghal, S., Kamoun, K., Ben Yahia S., Mephu-Nguifo, E., Slimani, Y. :EDOLA : Une Nouvelle Méthode d'Alignement d'Ontologies OWL-Lite - Dans CORIA'2007, pp.351-367 (2007)
2. Hemam, M., Boufaida, Z. : Prise en Compte des Points de Vue dans la Construction des Ontologies en Logique de Description. 6ème Colloque International sur l'Optimisation et les Systèmes d'Information, Algérie (2008)
3. Dorigo M., Caro G. D. :The Ant Colony Optimization Meta-Heuristic. : In New Ideas in Optimization, pp. 11–32. McGraw Hill, London, UK, (1999)
4. Hemam, M., Boufaida, Z. : Raisonnement par classification sur une ontologie multi-points de vue. Actes de la 3èmes Journée Francophone sur les Ontologies (JFO'09), Poitiers (France): ACM Edition. (2009)
5. Hemam, M., Boufaida, Z. : Représentation d'Ontologies Multi-points de Vue : une Approche Basée sur la Logique de Description. Conférence en Ingénierie des Connaissances, 25-29 mai, Hammamet, Tunisie, (2009)
6. Falquet, G., Mottaz, J.C.L. : Navigation Hypertexte dans une Ontologie Multi-Points de Vue. NîmesTIC'2001, Nîmes, France, (2001)
7. Miller, A.M. :WordNet : A Lexical Database for English.Communications of the ACM, 38(11) :39-41, (1995)
8. Gilles, B. : La similarité: une notion symbolique/numérique. Apprentissage symbolique-numérique (tome 2). Eds Moulet, Brito. Editions CEPADUES. pp. 169-201. (2000)
9. Wu, Z., Palmer, M.: Verb Semantics and Lexical Selection'', Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, pp 133-138 (1994)
10. Touzani, M.: Alignement des Ontologies OWL-Lite. Master'thesis, University of Montreal, (2005)
11. Solnon, C. : Contributions à la Résolution Pratique de Problèmes Combinatoires – des Fourmis et des Graphes-. Mémoire pour l'Obtention de l'Habilitation à Diriger des Recherches, Université Claude Bernard Lyon1, (2005).

Un Algorithme de Partitionnement d'Ontologies Orienté Alignement

Soumaya kasri[1,1], Fouzia Benchikha[1]
[1,1]Université de Skikda, département d'informatique, Algérie
[1]Université de Skikda, laboratoire LIRE Constantine, Algérie
{Kasri.soumaya, f_benchikha}@yahoo.fr

Résumé. Dans les applications réelles où les ontologies sont volumineuses, les exigences de l'exécution du temps et de l'espace de mémoire sont les deux facteurs significatifs qui influencent directement sur la performance d'un algorithme d'alignement. L'une des solutions du passage à l'échelle suppose la possibilité de partitionner les ontologies en blocs avant de réaliser l'alignement. Dans cet article, nous proposons un algorithme de partitionnement d'ontologie. Cet algorithme consiste à partitionner chaque ontologie en blocs autour des ancres en utilisant les algorithmes de clustering car les blocs qui contiennent les entités similaires (ancrées) devraient l'être aussi. Les résultats obtenus dans l'évaluation de notre algorithme montrent son efficacité.

Mot clé: ontologie, alignement, partitionnement, clustering.

1 Introduction

De nos jours, les ontologies sont devenues l'une des plus importantes orientations de recherche notamment avec l'avènement du Web Sémantique. Elles jouent un rôle primordial pour l'annotation de pages ou de services web puisqu'elles modélisent les concepts, attributs et relations utilisés pour annoter le contenu des ressources.

Dans de nombreux contextes applicatifs, plusieurs ontologies couvrant un même domaine ou des domaines connexes sont développées indépendamment les unes des autres par des communautés différentes. L'hétérogénéité entre les connaissances exprimées au sein de chacune d'entre elles doit être résolue. C'est la problématique de l'interopérabilité qui a pour objectif de permettre à des systèmes hétérogènes, qui s'appuient sur une de ces ontologies, de pouvoir communiquer et coopérer dans le but d'atteindre leurs objectifs. À cette fin, les liens sémantiques entre les entités appartenant à deux ontologies différentes doivent être établis d'où l'alignement d'ontologies.

L'alignement consiste à déterminer l'ensemble de correspondances entre deux ontologies en utilisant ou en mettant en œuvre des solutions aux différents problèmes d'hétérogénéité. L'ensemble de correspondances peut par la suite être utilisé notamment pour des applications telles que l'échange, l'intégration et la transformation des données.

De nombreuses méthodes d'alignement dédiées aux ontologies ont vu le jour cette dernière décennie [2,4,5]. Cependant, ces méthodes sont conçues pour aligner des

ontologies de petite taille. Dans la littérature actuelle, il existe très peu d'algorithmes d'alignement qui visent à traiter le problème du passage à l'échelle des méthodes d'alignement. L'une des solutions du passage à l'échelle suppose la possibilité de partitionner les ontologies en blocs avant de réaliser l'alignement [8,10]. En effet, pour diminuer l'espace de recherche des correspondances, il faut limiter la taille des ensembles de concepts en entrée de l'outil d'alignement.

L'objectif de notre travail est de relever le challenge du passage à l'échelle des méthodes d'alignement. En particulier, nous proposons un algorithme de partitionnement d'ontologie orienté alignement. Cet algorithme consiste à partitionner chaque ontologie en blocs autour des ancres[1] en utilisant les algorithmes de clustering. Les blocs générés peuvent être utilisés dans un algorithme d'alignement car les blocs qui contiennent les entités similaires (ancrées) devraient l'être aussi. Notre algorithme a été testé sur des ontologies de test disponibles pour le partitionnement notamment les paires Russia12 et TourimAB. Des résultats satisfaisants ont été obtenus.

La suite de l'article est organisée comme suit: la section 2 discute des travaux relatifs en présentant quelques méthodes de partitionnement qui sont appliquées dans différents domaines. Toutefois, nous nous sommes intéressés aux méthodes qui ont comme objectif l'alignement des ontologies. Dans la section 3, nous présentons notre algorithme de partitionnement. La section 4 présente une évaluation sur des ontologies de test et la section 5 conclut notre travail.

2 Etat de l'art

Avec l'apparition des nouveaux standards, outils et langages très expressifs, de grandes ontologies ont été développées dans plusieurs domaines (e.g biologie, géographie, médecine …etc.). Ces ontologies comportent plusieurs dizaines de milliers de concepts par exemples, l'ontologie Gene Ontology GO [14] et l'ontologie AGROVOC[2] [1] contiennent 26 057 et 28 439 concepts respectivement. Dans la littérature, on rencontre plusieurs algorithmes [8,10,12,13] qui visent le partitionnement des ontologies de façon à faciliter et à rendre plus performantes les opérations de maintenance, de visualisation, de raisonnement ou d'alignement.

Ainsi, les travaux de [12] proposent deux méthodes de décomposition d'ontologie en plusieurs sous-ontologies exprimées en logique de description. Cette décomposition est appliquée telle qu'elle préserve toujours la sémantique et les services d'inférence de l'ontologie originale. La première méthode est basée sur le séparateur minimal en utilisant l'algorithme récursive de Even [6] et la deuxième est basée sur la segmentation d'images en utilisant les vecteurs et les valeurs propres [11]. Les travaux dans [13] partitionnent une ontologie en blocs indépendants et cohérents à partir d'un graphe de dépendance. Le but de cette méthode est de faciliter

[1] Les ancres sont des entités jugées comme équivalentes en se basant sur une mesure de similarité.

[2] Une ontologie construite par le FAO (Food and Agriculture Organization).

les différentes opérations sur les ontologies (maintenance, validation et raisonnement).

Falcon-AO [10] et TaxoMap [8] sont deux méthodes de découverte de correspondance qui ont été préalablement développées dans le cadre d'alignement linguistique et structurel d'ontologies de petite taille. Mais, dans leurs versions actuelles Falcon-AO et TaxoMap intègrent des algorithmes de partitionnement adaptés au contexte d'alignement des ontologies volumineuses.

Dans le cadre de notre travail, nous nous intéressons plus particulièrement à ces deux méthodes que nous présentons brièvement dans ce que suit.

Falcon-AO. Cette méthode partitionne une ontologie en clusters $g_1, g_2, \ldots\ldots\ldots g_n$. L'algorithme proposé est agglomératif et inspiré de l'algorithme de clustering de ROCK [7]. On part de petits clusters nombreux et on les regroupe progressivement en clusters plus conséquents.

L'algorithme de partitionnement s'appuie sur deux propriétés essentielles: la cohésion au sein d'un cluster et le couplage entre deux clusters distincts. La cohésion mesure la similarité entre les entités appartenant à un même cluster et le couplage mesure la similarité entre les entités appartenant à deux clusters différents. Les deux propriétés sont calculées au sein d'une même fonction cut () qui mesure la distance entre deux clusters en reposant sur un critère d'agrégation. Ce dernier détermine la manière d'agglomérer deux clusters.

L'algorithme prend en entrée l'ensemble g des n clusters à partitionner, où chaque cluster est réduit au départ à une unique entité et ε le nombre des entités dans un cluster que l'on souhaite obtenir. Dans chaque itération, l'algorithme choisit tout d'abord le cluster qui a la cohésion maximale, puis le cluster qui a le couplage minimal avec ce premier cluster, et fusionne ces deux clusters.

Dans l'étape d'alignement, Falcon-AO aligne toutes les paires ayant une proximité supérieure à un seuil. Cette proximité s'effectue en s'appuyant sur des ancres. Soit b et b′ deux blocs, la proximité entre b et b′ est calculée comme suit :

$$\text{prox}(b, b') = \frac{2 \cdot \text{nombre d'ancres partagées par b et b'}}{\text{nombre d'ancres contenues par b} + \text{nombre d'ancres contenues par b'}} . \quad (1)$$

La proximité entre deux blocs est liée au nombre d'ancres partagées. Si le nombre d'ancres partagées est petit, la proximité sera inférieure au seuil et la paire n'est pas alignée c.-à-d. l'alignement de deux ancres partagées n'est pas retrouvé. D'autre part, plusieurs blocs pourront ne contenir aucune ancre. Dans ce cas ces blocs sont isolés[3] et on a risque de perdre plusieurs alignements.

TaxoMap. Deux méthodes ont été proposées. Soit O_S et O_T deux ontologies à aligner. La première méthode comprend trois étapes en plus du calcul des ancres. (1)Partitionner l'ontologie O_T en plusieurs blocs B_{Ti},. Le partitionnement est effectué

[3] Les blocs isolés sont des blocs qui ne contiennent aucune ancre ou dont la similarité avec les autres blocs ne dépasse pas le seuil choisi.

conformément à l'algorithme de Falcon-AO. (2)Identifier les centres CB_{Si} des futurs blocs de l'ontologie O_S. Les centres de O_S sont déterminés en se basant sur deux critères: les couples d'ancres identifiés entre O_S et O_T, et les blocs B_{Ti} construits à partir de l'ontologie O_T. (3)Partitionner l'ontologie source autour des centres CB_{Si} identifiés dans l'étape précédente.

La deuxième méthode comprend deux étapes. (1)Partitionner l'ontologie O_T en plusieurs blocs B_{Ti}. Le partitionnement est effectué conformément à l'algorithme de Falcon-AO mais en prenant en compte lors de la génération des blocs, l'ensemble des ancres. (2)Partitionner l'ontologie O_S de la même manière mais en se basant à chaque fois sur une partie des ancres qui appartiennent à un bloc de l'ontologie O_T .

Dans les méthodes de TaxoMap, le problème de l'alignement des ancres partagées est résolu mais le problème de perdre des alignements provenant des blocs isolés existe toujours.

Pour contribuer à résoudre ce problème, nous proposons un algorithme de partitionnement d'ontologie autour des ancres. Notre algorithme a l'avantage de n'avoir aucun bloc isolé en assurant la non perte des alignements et fournit en sortie des paires de blocs à ligner. Chaque bloc de la première ontologie ne s'aligne qu'avec un seul bloc de la deuxième ontologie.

3 Algorithme de partitionnement proposé

Dans une ontologie, les entités sont décrites en utilisant les noms, les étiquètes, et les commentaires. L'algorithme que nous proposons consiste à partitionner chaque ontologie en blocs autour des ancres, puis l'alignement s'effectue entre les paires des blocs fournies en sortie. Nous rappelons brièvement quelques définitions des mesures utilisées dans notre algorithme.

La similarité lexicale. Pour calculer la similarité lexicale entre les entités nous utilisons la mesure de Jaro-Winkler [15] qui prend simultanément en compte le nombre et la position des sous chaînes communes avec l'utilisation de la taille p du plus grand préfixe commun de deux chaines comparées. En OWL, les entités sont décrites en utilisant les constructeurs (rdf :id, rdfs : label, rdfs : comment) qui traduisent le triplet (nom, étiquète, commentaire) respectivement. Dans notre cas nous utilisons seulement les noms des entités (classe, propriété) pour calculer la similarité lexicale entre les entités, sans prétraitement, dans une mesure à moindre coût.

$$\mathrm{Sim_L}(e_1, e_2) = 1\text{-}\mathrm{DS_{Jaro-Winkler}}. \quad (2)$$

La similarité structurelle. Wu & Palmer [16] ont proposé de calculer la similarité entre deux concepts par la formule suivante :

$$\mathrm{Sim_S}(e_1, e_2) = \frac{2*\mathrm{depth}(e)}{\mathrm{depth}(e_1)+\mathrm{depth}(e_2)}. \quad (3)$$

e : le concept le plus spécifique qui subsume les deux concepts e_1, e_2 dans l'ontologie.

$depth(e)$: le nombre d'arcs qui séparent le concept e de la racine.

$depth(e_i)$: le nombre d'arcs qui séparent le concept e_i de la racine en passant par e .

La mesure de Wu & Palmer est intéressante, simple à implémenter et performante dans les évaluations. Cependant, pour une ontologie de grande taille, le calcul de similarité entre toutes les entités peut prendre beaucoup de temps. Pour cela, nous proposons d'effectuer le calcul de similarité seulement entre les entités qui se situent à un rayon r (r étant le nombre d'arcs entre les deux entités) dans un procédé itératif où r est défini selon la taille et la structure de l'ontologie.

Soient O_t et O_s deux ontologies à aligner, notre algorithme de partitionnement orienté alignement comprend les étapes suivantes :

1. Déterminer les couples d'ancres entre O_t et O_s en utilisant une mesure de similarité lexicale.
2. Classifier ces couples d'ancres en k clusters préliminaires.
3. Identifier les clusters d'ancres de chaque ontologie (l'ensemble des ancres appartenant à chaque ontologie) .
4. Classifier les concepts de chaque ontologie autour des clusters d'ancres.

Dans ce que suit, nous détaillons les différentes étapes de partitionnement.

3.1 Déterminer les ancres

Dans l'étape de classification, l'algorithme tentera de regrouper les entités autour des ancres. Nous disons que deux entités sont des ancres si et seulement si elles sont équivalentes. Pour cela, nous utilisons la similarité lexicale présentée ci-dessus dans la formule (2) pour les déterminer.

3.2 Partitionnement préliminaire

Le but de ce partitionnement est de regrouper les ancres dans K clusters. Après la détermination des ancres par la similarité lexicale, on regroupe chaque couple d'ancres dans un cluster comme montré en Fig. 1. Puis il s'agit d'exécuter un algorithme de clustering inspiré de celui de Falcon-AO[10].

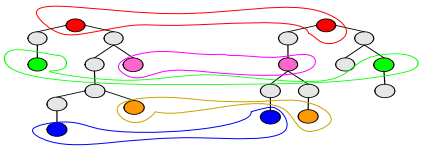


Fig. 1. L'entrée de l'algorithme (les couples d'ancres)

L'algorithme. La classification consiste à agréger progressivement les clusters d'ancres selon leur ressemblance. Cette agrégation nécessite un critère de classification. L'algorithme est réalisé grâce à l'utilisation d'un algorithme de partitionnement agglomératif hiérarchique inspiré de l'algorithme de Falcon-AO. La première différence entre notre algorithme et celui de Falcon-AO est que ce dernier prend en entrée toutes les entités de l'ontologie à aligner et le nombre k des blocs que l'on souhaite obtenir en sortie par contre notre algorithme prend en entrée les clusters initialisés par les paires d'ancres, le nombre de cluster k et le nombre maximum d'entités dans un cluster fusionnable m. Le but du partitionnement est de diminuer le nombre d'entités dans un bloc, car un système d'alignement n'est plus du tout efficace à partir d'un certain nombre d'entités.

La deuxième différence est que la méthode Falcon-AO utilise la fonction cut() comme un critère de classification. Celle-ci est définie comme suit :

$$\mathrm{cut}(g_i, g_j) = \frac{\sum_{d_i \in g_i} \sum_{d_j \in g_j} W(d_i, d_j)}{|g_i| \cdot |g_j|}. \quad (4)$$

$$\text{avec} \quad \begin{cases} \mathrm{cut}(g_i, g_i) = \mathrm{cohésion}(g_i) \\ \mathrm{cut}(g_i, g_j) = \mathrm{couplage}(g_i, g_j) \quad \text{avec } g_i \neq g_j \end{cases}.$$

Où g_i, g_j sont deux clusters et W est la matrice de la proximité (lexicale/structurelle) entre les entités. Pour calculer l'élément (i, j) dans W, Falcon-AO mesure le lien pondéré entre deux entités, $\mathrm{link}(e_i, e_j)$ comme suit :

$$\mathrm{link}(e_i, e_j) = \begin{cases} \mathrm{prox}(e_i, e_j) \text{ si } \mathrm{prox}(e_i, e_j) > \varepsilon \\ 0, \text{sinon} \end{cases}. \quad (5)$$

$$\text{avec } \mathrm{prox}(e_i, e_j) = \alpha\, \mathrm{prox}_s(e_i, e_j) + (1 - \alpha)\, \mathrm{sim}_l(e_i, e_j). \quad (6)$$

Où ε est un seuil donné tel que $\varepsilon \in [0, 1]$ et $\alpha \in [0, 1]$. Il permet à l'utilisateur de faire varier le poids relatif des mesures de similarité lexicale (sim_l) et structurelle (prox_s).

Dans notre cas, le calcul de similarité lexicale ne s'effectue que pour obtenir la cohésion au sein d'un cluster pour les raisons suivantes :

- le rapprochement de deux clusters se fait sur la base de similarité de leurs entités. Généralement, ces entités sont décrites différemment au sein d'une même ontologie. Pour cela, la similarité structurelle sera suffisante ;
- réduire la complexité du calcul ;
- obtenir un meilleur temps d'exécution.

Donc si $cut(g_i, g_j) = couplage(g_i, g_j)$, le calcul de lien pondéré s'effectue comme suit :

$$\mathrm{prox}(e_i, e_j) = \mathrm{Sim}_S(e_i, e_j). \quad (7)$$

Et si $\mathrm{cut}(g_i, g_j) = \mathrm{cohésion}(g_i)$, le calcul de lien pondéré s'effectue comme suit :

$$\mathrm{prox}(e_i, e_j) = \alpha\ \mathrm{Sim}_S(e_1, e_2) + \beta\ \mathrm{Sim}_L(e_1, e_2). \quad (8)$$

L'algorithme prend en entrée l'ensemble S de n clusters à partitionner, ou chaque cluster est réduit au départ à une paire d'ancres, le nombre k de clusters que l'on souhaite obtenir en sortie, et le nombre maximum m d'entités au sein d'un cluster fusionnable.

Comme Falcon-AO, dans chaque itération, l'algorithme choisit tout d'abord le cluster qui a la cohésion maximale (ordonner les clusters par leurs cohésions), puis le cluster qui a la valeur de couplage maximale. Si le nombre d'entités dans un cluster est égale au nombre maximum m, sa cohésion est réinitialisée par la valeur zéro, et les clusters sont réordonnés. Le pseudo-code de cet algorithme est présenté ci-dessous :

```
Algorithme(S,k,m)
Initialisercohesion(S);//initialiser chaque Si dans S
                       //par sa valeur de cohésion.
Ordonner(S);//ordonner les Si selon leurs cohésions .
Initialisercouplage(S);//dans une structure de type
                       //matrice
Tant que le nombre courant de cluster l > k  faire
   Prendre Si ;//le premier cluster dans S
   Choisir Sj ;//le cluster qui a la valeur de couplage
               //maximale avec Si et |Si|+|Sj| <= m*2
   Fusionner les deux cluster Si , Sj dans St;
   Supprimer Si et Sj ;
   Si |St| < m*2 faire// cluster des paires d'ancres
     Mettre à jour la cohésion de St ,
   Sinon
     Mettre la cohésion de St à zéro ,
   Finsi
   Ordonner(S),
   Mettre à jour la matrice de couplage,
   l:=l-1,
 Fin tant que
```

L'algorithme fournit en sortie un ensemble de k clusters d'ancres comme le montre la Fig. 2.

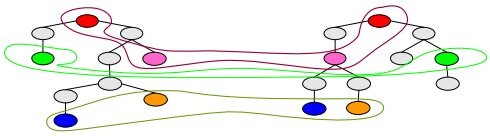


Fig. 2. La sortie de l'algorithme de partitionnement préliminaire (k clusters d'ancres)

3.3 Identifier les clusters d'ancres de chaque ontologie

Après la classification des ancres en K clusters d'ancres, nous partons de ces clusters et nous les divisons (voir Fig. 3) en des clusters qui ne contiennent que les ancres d'une même ontologie.

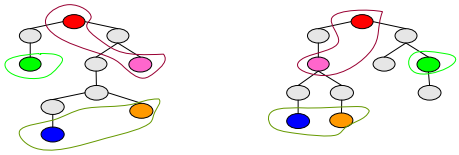


Fig. 3. Les clusters d'ancres de chaque ontologie.

3.4 Partitionnement autour des ancres

Dans cette étape, les concepts sont classifiés progressivement autour des ancres.

L'algorithme. La classification consiste à agréger progressivement dans chaque itération les concepts autour des ancres selon leur ressemblance en deux étapes :

- Etape d'affectation : pour chaque concept, on détermine le cluster d'ancres auquel on doit l'affecter (le cluster le plus proche c.à.d. où le couplage est le plus grand).
- Etape de représentation : pour chaque cluster défini, on recalcule les nouveaux couplages.

L'algorithme fournit en sortie un ensemble de blocs où chaque bloc contient une ou plusieurs ancres comme présenté en Fig. 4.

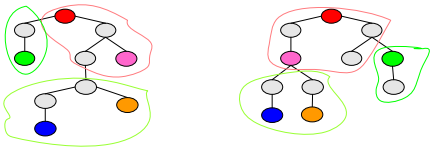


Fig. 4. Les blocs générés après la classification

Dans la phase d'alignement, chaque bloc de la première ontologie ne s'aligne qu'avec un seul bloc de la deuxième ontologie (voir la Fig. 5).

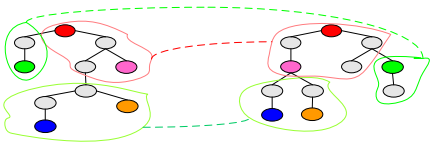


Fig. 5. Les blocs alignés après la classification

4 Evaluation de l'algorithme

Nous avons implémenté l'algorithme présenté ci-dessus sous forme d'un module java dans un système d'alignement. Dans l'évaluation de notre algorithme, nous avons choisi deux paires d'ontologies : Russia12 et TourismAB. Ce choix se justifie par :

- la taille des deux ontologies est modérée ;
- les fichiers de références, qui contiennent les entités classifiées en blocs, sont disponibles ;
- la comparaison des résultats de notre algorithme de partitionnement avec ceux des autres systèmes testés sur le même jeux de test.

Dans ce que suit, nous décrivons brièvement les deux paires de jeu de test.

Russia12. Les deux ontologies sont créées séparément par différentes personnes de deux contenus de sites web de voyages de la Russie. Russia1 contient 151 classes, 76 propriétés et 22 blocs dans son fichier de référence. Russia2 contient 162 classes, 81 propriétés et 22 blocs dans son fichier de référence.

TourismAB. Les deux ontologies sont créées séparément par différentes communautés décrivant le domaine du tourisme de MecKlenburg-Vorpommern (Allemagne). TourismA contient 340 classes, 97 propriétés et 18 blocs dans son fichier de référence. TourismB contient 447 classes, 100 propriétés et 23 blocs dans son fichier de référence.

Notre algorithme est testé sur une machine de technologie Intel Core 2 Duo CPU 2 GHz avec une mémoire de 3 GB DDR2, sur un système d'exploitation de Windows Vista, et un compilateur java 1.6.

4.1 Métrique d'évaluation

Nous utilisons la métrique d'entropie pour comparer les blocs générés automatiquement avec les blocs de référence [10].

Soit B l'ensemble de blocs générés automatiquement (|B|=n) et R l'ensemble de blocs de référence (|R|=m). b_i est un bloc dans B, tandis que r_j est un bloc dans R. $|b_i|$ est le nombre d'entités dans b_i , et $|r_j|$ est le nombre d'entités dans r_j .$b_i \cap r_j$ présente les entités communes dans les deux blocs b_i et r_j. Nous décrivons l'opération de base

appelée "prec" qui mesure la précision des blocs générés par rapport aux blocs de référence.

$$prec(b_i, r_j)=\frac{|b_i \cap r_j|}{b_i} \quad . \tag{9}$$

L'entropie mesure la distribution des entités dans les blocs et reflète la qualité de partitionnement. Un score faible de l'entropie indique une meilleure qualité de partitionnement. Le meilleur score est donc, égale à 0 et le mauvais est égal à 1. L'entropie de l'ensemble B est définie de la manière suivante :

$$\text{entropy (B)}=\frac{1}{\sum_{i=1}^{n}|b_i|}\sum_{i=1}^{n}\text{entropy}(b_i).|b_i|. \tag{10}$$

$$\text{entropy}(b_i) = \frac{1}{\log m}\sum_{j=1}^{n}\text{prec}(b_i, r_j).\log(\text{prec}(b_i, r_j)). \tag{11}$$

L'histogramme de la figure 6 présente les résultats d'évaluation de quelques systèmes (PBM, BMO,COMA) en utilisant cette métrique [10,9, 3]. Le critère d'arrêt du partitionnement est l'obtention du nombre de blocs de référence. L'histogramme présenté ci-dessous indique que les résultats d'expérimentation de PMB est dominant par rapport aux autres systèmes dans tous les tests.

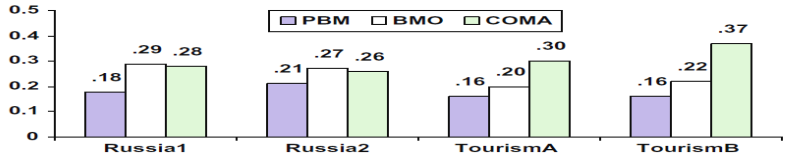


Fig. 6. L'entropie de PBM (Falcon-AO), BMO et COMA [10].

Notre algorithme de partitionnement est orienté alignement. La phase de partitionnement préliminaire des ancres est la plus importante car les blocs sont générés autour des ancres. Notre objectif est de partitionner les ontologies de façon à n'avoir aucun bloc isolé et diminuer le nombre de combinaisons à faire lors du processus d'alignement (un bloc de la première ontologie ne s'aligne qu'avec un seul bloc de la deuxième). En sortie, le nombre de blocs générés est exactement égal au nombre de bloc d'ancres généré lors de la phase de partitionnement préliminaire. Cependant, les blocs de référence disponibles sont construits manuellement sans prendre en compte l'objectif de l'alignement. Pour cela, nous adaptons les fichiers de référence aux fichiers de référence d'ancres pour évaluer notre algorithme de partitionnement préliminaire des ancres.

Dans notre expérimentation le nombre de blocs de référence après l'adaptation est comme suit : 13 blocs pour Russia1, 13 pour Russia2, 11 pour TourismA, et 14 pour TourismB. On a 55 ancres dans Russia12 et 144 dans TourismAB.

4.2 Démarche expérimentale

Les expérimentations que nous avons menées ont eu pour objectifs de calculer l'entropie de notre algorithme de partitionnement. Pour chaque paire d'ontologie, l'algorithme de partitionnement est exécuté plusieurs fois avec plusieurs valeurs présentant le nombre de blocs que l'on souhaite obtenir en sortie.

L'histogramme de la figure 7 présente les résultats d'évaluation de notre algorithme.

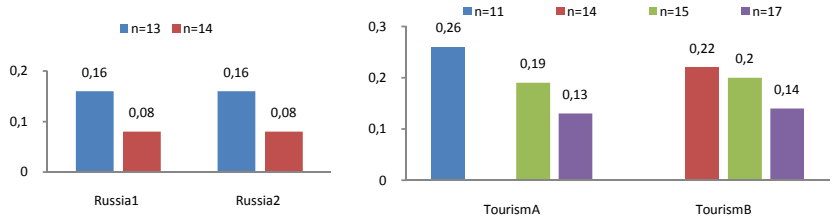


Fig. 7. L'entropie de l'algorithme de partitionnement d'ancres (les deux paires d'ontologies Russia12 et TourismAB)

Les résultats renvoyés par l'algorithme sont bons. Dans Russia12, si le nombre de blocs générés est égal au nombre de blocs de référence (n=m) l'entropie est égale à 0.16. Si nous augmentons le nombre de blocs générés, la valeur d'entropie diminue (0.08). Pour l'ontologie TourismAB, nous avons appliqué plusieurs valeurs du nombre de blocs générés notamment celui de référence (11 et 14) ainsi que les valeurs 15 et 17 (voir Fig. 7) et ceci afin de montrer l'influence de la valeur du nombre de blocs générés sur la qualité de partitionnement. Nous observons que l'entropie est changée en fonction du nombre de blocs générés choisi. Pour cela, le nombre de blocs a été choisi de façon à n'avoir aucun impact négatif sur l'alignement (i.e. la complexité de l'algorithme et le temps d'exécution). Les valeurs de l'entropie (0.26, 0.22) s'expliquent par le fait que le calcul de similarité structurelle se base seulement sur la taxonomie décrite par l'ontologiste et ignore les liens prédéfinis entre les classes et les propriétés OWL. Par exemple, les deux propriétés suivantes sont des ancres dans l'ontologie TourismA.

```
<rdf:Description rdf:about='http://meh/tourism1#DEFAULT_ROOT_RELATION'>
   <rdf:type rdf:resource='http://www.w3.org/2002/07/owl#ObjectProperty'/>
  </rdf:Description>
  <rdf:Description
rdf:about='http://meh/tourism1#DEFAULT_ROOT_DATATYPE_RELATION'>
   <rdf:type rdf:resource='http://www.w3.org/2002/07/owl#DatatypeProperty'/>
  </rdf:Description>
```

Elles sont classifiées dans deux clusters distincts car notre algorithme ne découvre pas le lien structurel à partir de leurs types (les deux propriétés DEFAULT_ROOT_RELATION et DEFAULT_ROOT_DATATYPE_RELATION' sont sous-propriétés de Property).

5 Conclusion

Dans le but de relever le challenge du passage à l'échelle des méthodes d'alignement, nous avons proposé une méthode de partitionnement des ontologies larges.

Notre algorithme de partitionnement est orienté alignement. Les blocs sont générés autour des ancres afin de n'avoir aucun bloc isolé. Nous poursuivons actuellement les expérimentations pour tester l'efficacité de l'algorithme avec notre méthode d'alignement structurelle à grande échelle. Cet algorithme sera appliqué sur les ontologies réelles, complexes, et de forte hétérogénéité.

Références

1. Agricultural Information Management Standards, http://www.fao.org/aims/
2. Bach, T.L.: Construction d'un Web Sémantique Multi-Points de Vue . These de doctorat Informatique, Ecole des Mines de Paris a Sophia Antipolis (2006)
3. Do, H.H. , Rahm, E.: Matching Large Schemas: Approaches and evaluation, Information Systems, vol.32 n.6, pp.857--885. (2007)
4. Ehrig, M., Staab, S.: QOM - Quick Ontology Mapping. In International Semantic Web Conference , pp. 683--697. (2004).
5. Euzenat, J., Valtchev, P.: Similarity-based Ontology Alignment in OWL-lite. In Proc. 15th ECAI, pp. 333--337. Valencia (ES), (2004)
6. Eyal, A., Sheila, M.: Partition-based Logical Reasoning for First-Order and Propositional theories, Artificial Intelligence, Vol. 162, pp. 49--88. (2005)
7. Guha, S., Rastogi, R., Shim, K.: ROCK: a Robust Clustering Algorithm for Categorical Attributes, in: Proceedings of the 15th International Conference on Data Engineering, pp. 512--521. (1999)
8. Hamdi, F., Zargayouna, H.,Safar, B., Reynaud, C.: TaxoMap in the OAEI Alignment Contest In Ontology Alignment Evaluation Initiative (OAEI) 2008 Campaign, 8p, Workshop ISWC'08, Karlsruhe, Germany, (2008)
9. Hu, W., Qu, Y.: Block Matching for Ontologies. In: LNCS, vol. 4273. Springer. pp. 300--313. (2006)
10. Hu, W., Qu, Y., Cheng, G.: Matching Large Ontologies: A Divide-and-Conquer Approach. Journal on Data and Knowledge Engineering, vol. 67, n°1, p.140--160. (2008)
11. Jianbo, S., Jitendra, M.: Normalized Cuts and Image Segmentation. IEEE Transactions on PAMI, Vol. 22, No. 8, pp. 888--905. (2000)
12. Pham, T.A.L., Thanh, N. L., Sander, P.: Some Approaches of Ontology Decomposition in Description Logics. 14th ISPE International Conference on Concurrent Engineering, CE2007, São José dos Campos, SP, Brazil, (2007)
13. Stuckenschmidt, H., Klein, M.: Structured-based Partitioning of Large Concept Hierarchies. In International Semantic Web Conference- ISWC, pp. 289--303. (2004).
14. The Gene Ontology, http://www.geneontology.org
15. Winkler W. E.: The State of Record Linkage and Current Research Problems. Statistics of Income Division, Internal Revenue Service Publication R99/04, (1999)
16. Wu, Z., Palmer, M.: Verb Semantics and Lexical Selection, Proceedings of the 32nd Annual Meetings of the Associations for Computational Linguistics, pp. 133--138. (1994).

Une approche multicritère pour lever l'ambiguïté morphologique dans le texte arabe

Mohamed Amine Chéragui[1], Youssef Hceini[2] et Moncef Abbas[3]

[1]Ecole national Supérieure d'Informatique (E.S.I.), E-mails : m_cheragui@esi.dz
[2]Institut d'informatique, Université de Bechar, B.P. 417 Bechar, E-mails : y_hoceini@yahoo.fr
[3]USTHB, Faculté de Mathématiques, Dépt. R.O, E-mails : moncef_abbas@yahoo.com

Résumé : Nous essayons à travers cet article, de décrire le phénomène de l'ambiguïté morphologique dans le traitement automatique du texte arabe et de présenter une nouvelle approche de désambiguïsation morphologique différente de celles qui existent actuellement, comme l'approche par contraintes ou l'approche stochastique (probabiliste/statistique), où l'originalité de notre travail réside dans le fait de s'appuyer sur un formalisme mathématique solide qui est l'aide multicritère à la décision.

Mots clés : TALN[1]; TALA[2] ; Ambiguïté morphologique; désambiguïsation; étiquetage morphosyntaxique; AMD[3]; critère; agrégation; pondération.

1 Introduction

Depuis l'apparition de l'intelligence artificielle au milieu des années cinquante, plusieurs chercheurs ont eux l'idée de modéliser et reproduire, à l'aide d'ordinateur la capacité humaine à produire et à comprendre des énoncés linguistiques dans le but de communication. Donnant ainsi naissance à une nouvelle discipline connue sous l'acronyme TALN qui veut dire Traitement Automatique du Langage Naturel faisant intervenir plusieurs sciences comme : la linguistique, la logique, l'informatique théorique, les statistiques, la neuroscience…etc.

Aujourd'hui, avec le développement qu'a subit l'informatique que se soit en terme de vitesse de traitement ou de support de stockage, le traitement automatique du langage naturel est devenu un domaine à la fois technologique due à l'émergence d'un nombre important d'applications, tels que : les traducteurs automatiques, générateurs automatiques de résumé, correcteurs orthographiques d'erreurs, …etc. Mais aussi un domaine scientifique traitant des problématiques de plus en plus complexes comme celle de l'ambiguïté.

Dans la littérature l'ambiguïté est comparée à un état de confusion, cet embrouillement se manifeste sous différentes formes et selon les différents niveaux de traitements que se soit lexical, morphologique, syntaxique et même sémantique. L'une

[1] TALN : Traitement Automatique de la Langue Naturel.
[2] T.A.L.A : Traitement Automatique de la Langue Arabe.
[3] A.M.D : Aide multicritère à la décision.

des formes d'ambiguïté la plus persistant en traitement automatique de la langue arabe est l'ambiguïté morphologique.

Dans cet article, nous donnons une brève et complète description du phénomène de l'ambiguïté morphologique ainsi que les différentes approches de désambiguïsation existantes. Nous présenterons ensuite notre approche pour lever cette forme d'ambiguïté qui se base sur les techniques de l'aide multicritère à la décision que nous appliquerons à la langue arabe.

2 La morphologie : Principe et Objective

2.1 Principe

La morphologie est un domaine de la langue qui permet la description des règles régissant la structure interne des mots (unités lexicales), chez les grammairiens la morphologie est l'étude des formes des mots (flexion et dérivation), en d'autres termes, la morphologie est l'étude des mots considéré isolément (hors contexte) sous le double aspect de la nature et des variations qu'ils peuvent subir [1]. En langue arabe, l'analyse morphologique est d'autant plus importante que les mots sont fortement agglutinés[4], c'est-à-dire qu'ils sont formés dans leur majorité par assemblage d'unités lexicales et grammaticales élémentaires [2].

Ainsi Le traitement morphologique est considéré comme une introduction principale à la compréhension globale d'une langue naturelle ; il joue un rôle très important aussi bien du côté linguistique que du côté technique.

2.2 Objective

La plupart des études faites sur la morphologie arabe dans le passé ou bien aujourd'hui visent généralement à satisfaire les points suivants:

- La formation de nouveaux mots à partir des éléments lexicaux disponibles ;
- L'analyse des mots réellement existant ;
- Fournir les données nécessaires aux travaux des différents niveaux de traitement (syntaxe, sémantique et pragmatique) ;

3 L'ambiguïté morphologique dans le T.A.L.A.

Le traitement morphologique porte sur l'unité élémentaire identifiable, en l'occurrence le morphème, à ce niveau on s'intéresse à deux concepts essentiels qui sont :

- *la synthèse* qui permet de générer des mots ou des phrases par le biais d'un ensemble de règles de dérivation, de flexions et d'adaptations.

[4] Processus d'ajout d'affixes à un mot qui exprime ses différentes relations grammaticales.

- *l'analyse* qui a pour rôle d'associer à un mot graphique un ensemble d'informations décrivant les unités morphologiques et grammaticales entrant dans sa composition (proclitiques, préfixes, base, suffixes, enclitique) [3]. C'est à ce stade, que l'ambiguïté morphologique se manifeste ; lorsque l'analyse assigne à une unité lexicale plusieurs informations (ou l'inverse) ce qui génère la notion de combinatoire [4].

  Exemple :

  Prenons comme exemple, le mot «أحْـمَـدُ », ce dernier peut prendre plusieurs interprétations (voir figure 1) et celà est dû au contexte.

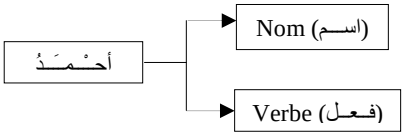


Figure 1. Exemple d'une ambiguïté morphologique.

4 Etiquetage Morphosyntaxique pour lever l'ambiguïté morphologique

4.1 Qu'est ce que le Tagging?

Le Tagging (étiquetage ou marquage) est le fait d'assigner une étiquette ou un tag à un mot. Le tag contient des informations morphosyntaxiques sur le mot, c'est-à-dire des informations concernant la forme et la fonction du mot. Elles comprennent notamment la catégorie grammaticale, le genre, le nombre, le temps et le mode [5].

4.2 Principe de Base

Pour mettre en place un étiqueteur morphosyntaxique de la langue arabe, on doit construire trois (03) modules qui seront complémentaires, ces modules sont :

- *Module de segmentation :* lorsqu'on évoque la segmentation dans le traitement automatique des langues dites naturelles, on parle le plus souvent de trois (03) niveaux de segmentations :
  - Segmentation au niveau du texte ;
  - Segmentation au niveau de la phrase ;
  - Segmentation au niveau du mot.
- *Module d'analyse morphologique :* Le but principal de ce type d'analyse est de vérifier l'appartenance d'un mot donné au domaine linguistique choisi et de pouvoir disposer ainsi de tous les renseignements le concernant pouvant servir à l'analyse syntaxique.
- *Module de désambiguïsation :* La désambiguïsation est une étape cruciale dans le processus d'étiquetage morphosyntaxique, à ce niveau du traitement si un mot est

mal étiqueté, les règles de la grammaire s'appliqueront mal ou pas du tout. Cependant la phase de désambiguïsation n'est pas toujours nécessaire ou obligatoire au bon déroulement du processus d'étiquetage. Il faut dire que le module de désambiguïsation rentre en jeu dans un seul cas de figure, celui où l'unité lexicale (mot) reçoit plus d'une étiquette (plus d'une information morphosyntaxique), ce qui va générer une situation de confusion ou ambiguïté. C'est à ce stade que notre contribution va apparaître en présentant une nouvelle démarche de désambiguïsation différente de celle qui existe actuellement basée sur l'aide multicritère à la décision.

5 Méthodes déjà existantes pour lever l'ambiguïté morphologique arabe

Dans la littérature, les approches de désambiguïsation se répartissent en deux (02) catégories et chaque catégorie englobe une ou plusieurs techniques pour lever l'ambiguïté morphologique. La figure 2 ci-après donne une indication sur les différentes approches et les techniques qui vont avec :

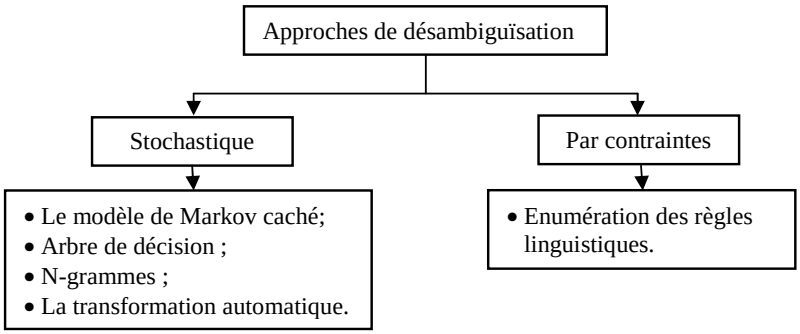


Figure 2: Approches et techniques de désambiguïsation.

5.1 Approche par contraintes

L'approche de désambiguïsation par contraintes (ou par règles) est la méthode la plus ancienne qui a été mis en place pour remédier au problème de l'ambiguïté morphologique. Cette approche se base principalement sur l'intervention d'un linguiste (ou un grammairien) afin d'établir une liste de règles permettant de lever l'ambiguïté. Ces règles sont généralement classées en trois (03) catégories [6], qui sont :

- Les règles contextuelles ou de principes ;
- Les heuristiques ;
- Les règles non contextuelles.

Parmi les étiqueteurs qui adoptent l'approche par contraintes on peut citer : Freeman's Arabic Tagger [7].

5.2 Approche stochastique (probabiliste / statistique)

Lever l'ambiguïté morphologique en adoptant une approche stochastique, cela consiste essentiellement à définir deux (02) sortes d'informations [8] :

- La première est sur le mot a étiqueté (*i.e. l'association entre le mot et l'étiquette*) : Soit « W » une unité lexicale et « T » l'étiquette qui va avec.

$$P(T/W) = \frac{P(T, W)}{P(W)} . \quad \textbf{(1)}$$

  Où :
  - P(T, W) : le nombre de fois que « W » est étiqueté par « T » ;
  - P(W) : le nombre d'occurrence de « W ».
- La seconde information est contextuelle syntaxique (*i.e. la possibilité de déterminer la probabilité d'avoir une étiquette « T^k » quand elle est précédée de l'étiquette « T^j » dans le texte*).

$$P(T^k/T^j) = \frac{P(T^k, T^j)}{P(T^j)} . \quad \textbf{(2)}$$

En plus de ces deux (02) hypothèses qui sont considérées comme étant les formules de calcul, une phase d'apprentissage est obligatoire et celà, en entraînant le module de désambiguïsation sur un corpus généralement annoté (*i.e. étiqueter à la main*) au préalable. Parmi les étiqueteurs qui adoptent l'approche stochastique on peut citer : TnT[5] qui utilise le modèle de Markov caché [9].

6 Une approche de désambiguïsation basée sur l'aide multicritère à la décision

Le but de notre étude est de proposer une nouvelle approche de désambiguïsation différente de celles qui existent actuellement, comme l'approche stochastique et l'approche par contrainte. Cette nouvelle approche dédiée à la levée de l'ambiguïté morphologique s'appuie essentiellement sur les principes de la théorie décisionnelle et les méthodes issues de l'aide multicritère à la décision.

6.1 Justification du choix d'une démarche multicritère

L'intérêt d'adopter une approche d'aide multicritère à la décision pour lever l'ambiguïté morphologique dans le T.A.L.A., peut être résumé en deux (02) points, qui sont :

- Réduction de l'ensemble des étiquettes candidates: elle permet de réduire d'amblée le nombre d'étiquettes de correction, en éliminant ceux dominées générant ainsi l'ensemble des étiquette efficaces ;

[5] TnT : Trigrams 'n' Tags.

- Classification des étiquettes efficaces, selon un score global obtenu après traitement suivant un ordre décroissant.

6.2 Différentes étapes d'une démarche d'aide multicritère à la décision

La mise en place d'une démarche basée sur l'aide multicritères à la décision exige le plus souvent de suivre une succession d'étapes afin de générer un résultat. Ces étapes sont [10], [11]:

- Etape 1 : Dresser la liste des scénarios potentiels, cette liste contient les scénarios (*i.e. solution, actions, …etc.*), qui sont provisoirement considérées comme étant candidates ou bien réaliste du point de vue du décideur.
- Etape 2 : Dresser la liste des critères, une bonne application d'une démarche multicritère passe impérativement par un bon choix concernant les critères, sur lesquelles le calcul sera posé. Ces critères seront définis en se basant sur les notions d'indifférence, de préférence stricte ou faible ou de non comparabilité.
- Etape 3 : Définir une fonction d'évaluation pour chaque critère sélectionné une fonction d'évaluation est définie. Cette dernière doit être maximisée ou bien minimisée selon le type de critère utilisé.
- Etape 4 : Etablir un tableau de performance, ce tableau (*i.e. matrice d'évaluation*) contient tous les résultats des évaluations de chaque scénario potentielle suivant tous les critères ; de telle sorte que les lignes de ce tableau correspondent aux scénarios potentielles et les colonnes correspondent aux critères, et les intersections correspondent aux évaluations.
- Etape 5 : Pondération et agrégation des performances, afin de s'assurer à la fois d'une bonne analyse des résultats et d'appliquer le concept de dominance (relation de dominance) une phase de normalisation est obligatoire. Cette phase est appelée la pondération des critères qui consiste à attribuer à chaque critère un poids justifiant ainsi son importance ; ainsi les critères les plus importants suivant les préférences du décideur auront les poids les plus élevés. Après la pondération une phase d'agrégation s'enchaine afin de générer une évaluation en appliquant l'une des trois approches d'agrégation (globale, partielle ou interactive).

6.3 La méthode d'agrégation « TOPSIS[6] »

Créée par Hwang et Yoon en 1981 [12], cette méthode se base sur la relation de dominance qui résulte de la distance par rapport à la solution idéale. Son fondement consiste à choisir une solution qui se rapproche le plus de la solution idéale (*i.e. la meilleure sur tous les critères*) et de s'éloigner le plus possible de la pire solution qui dégrade tous les critères. Cette méthode est composée de six (06) phases:

- Phase 1: Calcul de la matrice de décision normalisée, Les valeurs normalisées « e_{ij} » sont calculées comme suit :

[6] TOPSIS: Technique for Order by Similarity to Ideal Solution.

$$e'_{ij} = \frac{g_j(a_i)}{\sqrt{\sum_{i}^{m}[g_j(a_i)]^2}} \qquad \text{Avec:} \quad \begin{matrix} i= 1, \ldots,m. \\ j= 1, \ldots, n. \end{matrix} \qquad \textbf{(3)}$$

Où : les « $g_j(a_i)$ » correspondent aux valeurs déterministes des scénarios « i » pour le critère « j ».

- Phase 2 : Calcul de la matrice de décision normalisée pondérée (*i.e.* Calculer le produit des performances normalisées par les coefficients d'importance relative des attributs). Les éléments de la matrice sont calculés comme suit :

$$e''_{ij} = \pi_j \, . \, e'_{ij} \qquad \text{Avec:} \quad \begin{matrix} i= 1, \ldots,m. \\ j= 1, \ldots, n. \end{matrix} \qquad \textbf{(4)}$$

Où : «πj» est le poids du $j^{ième}$ critère et: $\sum_{j}^{n} \pi_j = 1.$

- Phase 3 : Détermination des solutions (profils) idéale (a^+) et des solutions anti-idéale (a_-) par rapport à chaque critère.

$$\begin{aligned} &a^+ = \{\max_i e''_{ij},\ i= 1, \ldots, m \text{ ; et } j=1, \ldots, n\} \text{ ;} \\ &a^+ = \{e_j^+,\ j=1, \ldots, n\} = \{e_1^+, e_2^+, \ldots, e_n^+\} \text{ ;} \qquad e_j^+ = \max_i \{e''_{ij}\}. \\ &a_- = \{\min_i e''_{ij},\ i= 1, \ldots, m \text{ ; et } j=1, \ldots, n\} \text{ ;} \\ &a_- = \{e_{j-},\ j=1, \ldots, n\} = \{e_{1-}, e_{2-}, \ldots, e_{n-}\} \text{ ;} \qquad e_{j-} = \min_i \{e''_{ij}\}. \end{aligned} \qquad \textbf{(5)}$$

- Phase 4 : Calcul des mesures d'éloignements (*i.e. Calculer la distance euclidienne par rapport aux profils a^+ et a_-*) ; L'éloignement entre les alternatives est mesuré par une distance euclidienne de dimension « n ». L'éloignement de l'alternative « i » par rapport à la solution idéale (a^+) et anti idéale (a_-), qui peut être assimilé à la mesure d'exposition aux risque et donné par :

$$D_i^+ = \sqrt{\sum_{i}^{n}(e''_{ij} - e_j^+)^2}. \qquad \text{Avec :} \quad i= 1,2, \ldots, m. \qquad \textbf{(6.1)}$$

$$D_{i\_} = \sqrt{\sum_{i}^{n}(e''_{ij} - e_{j-})^2}. \qquad \text{Avec :} \quad i= 1,2, \ldots, m. \qquad \textbf{(6.2)}$$

- Phase 5 : Calculer un coefficient de mesure du rapprochement au profil idéal :

$$C_i^+ = \frac{D_{i-}}{D_i^+ + D_{i-}}. \qquad \text{Avec:} \quad \begin{matrix} i= 1, \ldots, m \text{ ;} \\ 0 \le C_i^+ \le 1. \end{matrix} \qquad \textbf{(7)}$$

- Phase 6 : Rangement des scénarios suivant leur ordre de préférences (*i.e.* en fonction des valeurs décroissantes de C_i^+ ; « *i » est meilleur que «j » si $C_i^+ > C_j^+$*).

6.4 La méthode de pondération « Entropie »

Comme la méthode TOPSIS ne permet pas de générer de manière automatique les poids des critères, on n'était obligé d'intégrer une autre méthode pour pondérer les critères à l'intérieur de TOPSIS. Cette méthode de pondération est l'Entropie, dont le fondement et l'algorithme sont donnés comme suit :

La méthode Entropie [11] est une technique objective de pondération des critères, l'idée est qu'un critère « j » est d'autant plus important que la dispersion des évaluations des scénarios est importante. Ainsi les critères les plus importants sont ceux qui discriminent le plus entre les scénarios (dans notre cas ce sont les étiquettes). Les étapes de cette méthode sont données comme suite :

- Etape 1 : L'entropie d'un critère « j » est calculée par la formule :

$$E_j = -K \sum_{j}^{n} X_{ij} \log(X_{ij}). \quad \textbf{(8)}$$

Où :

- K : est constante choisie de telle sorte que, pour tous « j », on a $0 \leq E_j \leq 1$, pour notre cas « K » est calculé comme suit :

$$K = \frac{1}{\text{Log (n)}}. \qquad \text{Avec: n : le nombre de scénarios de désambiguïsation.} \quad \textbf{(9)}$$

- X_{ij} : l'évaluation du scénario « i » suivant le critère « j ».

- Etape 2 : L'entropie E_j est d'autant plus grande que les valeurs de « e_j » sont proches. Ainsi, les poids seront calculés en fonction de la mesure de dispersion (opposée de l'entropie) :

$$D_j = 1 - E_j. \qquad \text{Avec:} \quad j= 1, \ldots, n. \quad \textbf{(10)}$$

- Etape 3 : Les poids seront ensuite normalisés par :

$$W_j = \frac{D_j}{\sum_{j}^{n} D_j} \qquad \text{Avec:} \quad j= 1, \ldots, n. \quad \textbf{(11)}$$

7 Exemple illustrative

Afin de mieux cerner la solution proposée, nous allons garder la même démarche multicritère citée auparavant, tout en incluant les modifications et explications nécessaires :

Soit la phrase P= « **رَجَعَ المُغْتَرِبُ إلى الوَطَن.** », qui se trouve à l'entrée de notre analyseur.
Après segmentation de la phrase en mots, l'analyse se fait sans problème pour les unités 2, 3 et 4, mais l'unité 1 « **رَجَـعَ** » présente un cas typique d'ambiguïté morphologique.
Pour lever cette ambiguïté nous allons appliquer notre approche de désambiguïsation multicritère, selon la démarche suivante :

- ***Etape 1 :*** **Construire la liste des scénarios d'analyse :** Cette liste est obtenue directement après le processus d'analyse morphologique. Ce qui va générer l'ensemble « A ».

Verbe	Scénario (Schème)	longueur
رجـــع	فَـعِـلَ	6
	فَـعِـلَ	6
	فَـعُـلَ	6
	فُـعِـلَ	6

Tableau 1 : Exemple d'ambiguïté générée lors de l'analyse du verbe « رجَـعَ ».

- ***Etape 2 :*** **Application des critères**

  Afin de construire une famille cohérente de critères F, nous proposons deux critères de base pour discriminer entre les scénarios d'analyse, à savoir :

  a) Critère 1 : Concordance des voyelles

  Ce critère va vérifier la concordance entre les voyelles de l'unité lexicale et les voyelles de chaque scénario candidat, de telle sorte que chaque concordance vaut : un (1).

  b) Critère 2 : La Fréquence d'apparition

  Ce critère s'appuie sur un calcul statistique fait sur la base d'un corpus annoté, de telle manière que le scénario qui se manifeste le plus souvent (une plus grande fréquence d'apparition) aura systématiquement le score le plus élevé. Chaque apparition vaut : un (1).

  Remarque :

  Le corpus utilisé est composé de plus de 300 unités réparties sur 10 paragraphes choisis arbitrairement à partir des livres scolaire de l'école algérienne.

- ***Etape 3 :*** **Application de la fonction d'évaluation**

  Pour les deux (02) critères, la fonction d'évaluation est l'addition (+), ainsi il s'agit de deux (02) critères à maximiser.

- ***Etape 4 :*** Générer le tableau (matrice d'évaluation) d'évaluation

➔ Scénario ⬇ Critère	S1«فَـعَـلَ »	S2«فَـعِـلَ»	S3«فَـعُـلَ»	S4«فُـعِـلَ»
Concordance des voyelles	3	2	2	1
Fréquence d'apparition	16	5	2	1

Tableau 2 : Tableau (Matrice) d'évaluation.

- ***Etape 5 :*** Agrégation des performances et pondération des critères.

  a) Normalisation du tableau d'évaluation

  En appliquant la formule (3) de la méthode TOPSIS.

➔ Scénario ⬇ Critère	S1«فَـعَـلَ»	S2«فَـعِـلَ»	S3«فَـعُـلَ»	S4«فُـعِـلَ»
Concordance des voyelles	0.71	0.47	0.47	0.24
Fréquence d'apparition	0.95	0.30	0.12	0.06

Tableau 3: Tableau d'évaluation normalisé

b) Pondération des critères.

Pour pondérer les critères nous utilisons la méthode Entropie. Le tableau suivant montre les valeurs de calcul de l'entropie (E_j), l'opposé de l'entropie (D_j) et normalisation des poids (W_j) des deux critères.

➔ Scénario ⬇ Critère	E	D	W
Concordance des voyelles	0.24	0.76	0.47
Fréquence d'apparition	0.15	0.85	0.53

Tableau 4: Pondération des critères

c) Pondération du tableau d'évaluation (normalisé)

Cette pondération est faite en appliquant la formule (4) de la méthode TOPSIS.

➔ Scénario ⬇ Critère	S1«فَـعَـلَ»	S2«فَـعِـلَ»	S3«فَـعُـلَ»	S4«فُـعِـلَ»
Concordance des voyelles	0.33	0.22	0.22	0.11
Fréquence d'apparition	0.50	0.16	0.06	0.03

Tableau 5: Tableau d'évaluation normalisé et pondéré.

d) Détermination de la solution idéale (D^+) et la solution anti idéale (D^-)

La solution idéale correspond à la valeur maximale et la solution anti idéale correspond à la valeur minimale.

	Solution idéale	Solution anti idéale
Concordance des voyelles	0.33	0.11
Fréquence d'apparition	0.50	0.03

Tableau 6: Détermination de la Solution idéale et anti idéale.

e) Calcul des mesures d'éloignements

Après application des formules (6.1) et (6.2) de la méthode TOPSIS on aura les différentes mesures d'éloignement de chaque scénario comme il est illustré dans le tableau suivant :

	S1«فَـعَـلَ»	S2«فَـعِـلَ»	S3«فَـعُـلَ»	S4«فُـعِـلَ»
D^+	0	0.36	0.45	0.52
D^-	0.52	0.17	0.14	0

Tableau 7: Mesures d'éloignements.

f) Calcul des coefficients de mesure du rapprochement au profil idéal : Pour calculer ces coefficients C_i^+, nous utilisons la formule (7) de la méthode TOPSIS, nous allons établir un classement de ces coefficients, selon un ordre décroissant et le scénario ayant obtenu le score le plus élevé sera élu. On aura les valeurs suivantes :

$$C_1^+=1 > \quad C_2^+=0,32 > \quad C_3^+=0,24 \; > \; C_4^+=0.$$

Selon notre méthode le scénario 1 « فَـعَـلَ » sera sélectionné par le système, donc les informations morphologiques suivantes seront générées.

	Information
Verbe	رَجَـعَ
Schème	فَـعَـلَ
Etiquette	VAA3PMSIA
Désignation en français	Verbe Accompli Actif 3e Personne Masculin Singulier Invariable Accusatif.
Désignation en arabe	فعل ماضي مبني للمعلوم للمفرد المذكر الغائب، مبني على الفتح.
Racine	رجع

Tableau 8: Informations générées de l'étiquetage du verbe « رجـع ».

8 Conclusion

L'étiquetage morphosyntaxique est considéré aujourd'hui, comme étant la partie vitale dans n'importe quelle application de traitement automatique du langage naturel. De ce fait, la performance d'une application linguiciel[7] dépend directement de l'efficacité de

[7] Logiciel traitant la langue naturel sur les différents niveaux : morphologique, syntaxique, sémantique et pragmatique, utilisant des bases de connaissances linguistique.

son étiqueteur morphosyntaxique et pour obtenir une fiabilité irréprochable on doit s'intéresser à trois (03) points qui constituent la structure d'un étiqueteur, à savoir : une bonne phase de segmentation, une bonne organisation des unités lexicales de la langue et surtout un module de désambiguïsation fiable. C'est ce dernier point qui est le centre d'intérêt de notre étude. Nous avons essayé à travers cet article de mettre l'accent sur l'impact que l'ambiguïté sur le processus d'automatisation de la langue arabe et plus particulièrement l'ambiguïté morphologique, ainsi que les méthodes dédiées à la lever de cette forme d'ambiguïté, comme l'approche par contraintes et l'approche stochastique.

Mais notre principal objectif reste de présenter notre nouvelle approche de désambiguïsation fondée sur les techniques issues de l'aide multicritère à la décision.

Comme conclusion, qu'on peut tirer de cette étude est que l'approche multicritère peut être considérée comme une alternative de choix pour remplacer les deux (02) approches citées auparavant, afin de remédier au problème de l'ambiguïté morphologique qui constitue un point de passage qui condamne la réussite ou l'échec d'une application en T.A.L.A.

Références

1. Hoceini Y.: Un système d'analyse morphologique de la langue arabe, mémoire magister, école nationale supérieure d'informatique, (2002).
2. Baloul S. : Développement d'un système automatique de synthèse de la parole à partir du texte arabe standard voyellé, thèse de doctorat, université du Mans, (2003).
3. Tuerlinckx L.: La lemmatization de l'arabe non classique, U.C.L. Institut orientaliste, Belgique.
4. Vergne J., Giguet E.: Regards théorique sur le tagging, TALN, pp 22-31, (1998).
5. Lechel M. : Analyse et conception d'un correcteur grammatical libre pour le français, thèse de magister, université de Gronoble 3, (2005).
6. Chanod J. P., Tapanainen P. : Les étiqueteurs statistiques et les étiqueteurs par contraintes, (1995).
7. Al-Sulaiti L.: Deigning and Developing a Corpus of Contemporary Arabic, School of Computing, University of Leeds (UK), March (2004).
8. Merialdo B. : Modèle Probabiliste et Etiquetage Automatique, Institut EUROCOM, Sophia Antipolis.
9. Constant M. : Etiquetage morphosyntaxique probabiliste, Cours pour Master en informatique, Université Paris-Est Marne la Vallée, (2007).
10. Belguith L. H., Hamadou A. B. : Traitement des erreurs d'accord, RSTI-RIA, Volume 18, pp 679-707, (2004).
11. Vincke P. : L'aide multicritère à la décision, SMA, Université de Bruxelles, (1989).
12. Hwang C. R., Yoon K.: Lecture Notes in economics and Mathématical System, Springer Verlag Berlin Heidelberg, New York, (1981).

Graphes et optimisation

Plongement et placement de certaines classes d'arbres dans l'hypercube

Kamel KABYL[1] and Abdelhafid BERRACHEDI[2]

[1] Université A/Mira de Béjaia, Département des sciences Commerciales,
k_kabyle2000@yahoo.fr,
[2] Faculté de mathématiques, USTHB,BP32 El Alia 16111 Bab Ezzouar,
abdelhafid_berrachedi@yahoo.fr

Résumé De nombreux problèmes (boolèens, de théorie des graphes, de codes...) sont formalisables comme problèmes combinatoires sur l'hypercube. Lhypercube est un graphe intéressant dont la topologie est utilisé en informatique (parallélisme, réseaux), il est fondamentale de déterminer quels sont les graphes et particulièrement les arbres qui sont plongeable dans l'hypercube et de déterminer aussi combien de copies d'un arbre donné qu'on peut placer dans un hypercube de dimension donnée. Nous avons introduit certaines classes d'arbres pour lesquelles nous avons déterminé la dimension cubique et nous avons donné aussi pour certaines classes le nombre de copies qu'on peut placer dans l'hypercube de dimension donnée.
Mots-clés : Hypercube, Plongement, Graphes, Arbres, Isomorphisme

1 Introduction

Un plongement de $G(V, E)$ dans l'hypercube est défini par la donnée d'une application injective φ de l'ensemble des sommets de G dans l'ensemble des sommets de Q_n, et d'une application P_φ de l'ensemble des arêtes de G dans l'ensemble des arêtes de Q_n, qui associe à chaque arête uv de G une arête $\varphi(u)\ \varphi(v)$ dans Q_n. D'une manière générale, l'étude d'un plongement de graphe G dans l'hypercube revient à voir si G est isomorphe à un sous graphe de Q_n. Ce problème est très étudié en théorie des graphes. En effet de nombreux efforts ont été consacrés pour déterminer les conditions (nécessaires et suffisantes) selon lesquelles un graphe G est un sous-graphe de l'hypercube Q_n. Une classe importante à étudier est celles des arbres dans l'hypercube. Cette importance résulte de l'utilisation de ces arbres dans plusieurs domaines, à savoir : informatique, sciences sociales, recherche opérationnelle, optimisation combinatoire, théorie des réseaux électriques...et l'utilisation pratique de l'hypercube en théorie des codes, transfért de l'information, architecture parallèle, décision

mulicrére, réseaux d'intérconnéxion etc... Un graphe $G = (V, E)$ est dit cubique s'il est plongeable dans Q_n pour un certain n. Firsov[8] a remarqué que les arbres sont des graphes cubiques et a montré que tout graphe cubique est nécessairement biparti, mais la réciproque n'est pas toujours vraie. Le problème consiste à trouver la plus petite dimension de l'hypercube dans lequel un arbre donné G est plongeable (on parle alors d'hypercube optimal). Arfati, Papadimitriou et Papageorgiou [1] ont montré le résultat suivant : Le problème de décider si un graphe G est plongeable dans Q_n est $N.P$-complet.Wagner et Corneil [4] ont montré que ce problème reste N.P complet même dans le cas où G est un arbre. Plusieurs auteurs se sont intéréssés á l'étude de plongement d'arbres dans l'hypercube : On peut citer : A. Berrachedi [2], S. Bezrukov [3], I. Havel [5], F. Harary [9], M. Kobeissi [11], M.Laborde [12].... Dans le même contexte, on définit dans ce papier des nouvelles classes pour lesquelles la dimension cubique est déterminée. Comme on a donné aussi le nombre maximum de copies de certaines topologies qu'on peut placer dans un hypercube de dimension donnée. Pour un graphe $G(V(G),\ E(G))$ $V(G)$ et $E(G)$,désignent respectivement l'ensemble des sommets et l'ensemble des arêtes.
Un hypercube de dimension n, noté Q_n, est le graphe dont l'ensemble de sommets sont les n-uplets binaires et deux sommets sont adjacents si et seulement s'ils différent en une seule coordonnée.
Un graphe biparti $G(X \cup Y; E)$ est dit équilibré si $Card(X) = Card(Y)$. L'hypercube Q_n est un graphe biparti équilibré, n-régulier ayant 2^n sommets et $n.2^{n-1}$ arêtes. Tout graphe plongeable dans un hypercube est dit cubique. Comme conditions nécessaires de plongement de graphe dans Q_n on a : pour qu'un graphe G soit plongeable dans Q_n il faut que : $|V(G)| \leq 2^n$, G est biparti et le degré maximum de (G), $\Delta(G) \leq n$. Si de plus $|V(G)| = 2^n$ alors G doit être équilibré. Toutes ces conditions sont nécessaires pour un graphe G plongeable dans Q_n, mais ne sont pas suffisantes.

La Cn-valuation aux cas des arbres est donnée comme suit : Un arbre T est Cn- valué si les arêtes de T sont marquées par les entiers de l'ensemble $\{1, 2, 3, \ldots, n\}$ de sorte que pour toute chaîne P de T, il existe un entier $K \in \{1, 2, 3, \ldots, n\}$ pour lequel un nombre impair d'arêtes de P sont marquées par K. I. Havel et moravek [7] ont montré qu'un graphe G est plongeable dans Q_n si et seulement s'il existe une Cn-valuation de G.

2 Quelques classes d'arbres plongeables dans Q_n

On présente quelques résultats connus sur les plongements d'arbres dans l'hypercube.

2.1 Arbres binaires

Un arbre T est dit binaire si son degré maximum $\Delta(T) \leq 3$. Un résultat concernant les arbres binaires a été donné par I-Havel [5].

Proposition 1. *[5] Soit T un arbre binaire d'ordre 2^n avec $n \geq 3$. Si T est équilibré et possède deux sommets de degré 3 alors T est plongeable dans Q_n.*

2.2 Arbres binaires complets

L'arbre binaire complet D_n est le graphe défini inductivement comme suit : Pour $n = 1$, $D_1 = K_{1,2}$ est un graphe biparti complet. Pour $n \geq 2$, D_n est obtenu à partir de deux copies disjointes $T_1,\ T_2$ de D_{n-1} et d'un nouveau sommet u, tel que u est relié par une arête à un sommet de degré 2 de T_1 et par une autre arête à un sommet de degré 2 de T_2. D_n Possède 2^n sommets pendants, $2^n - 2$ sommets de degré 3 et un seul sommet de degré 2. Le sommet de degré 2 sera appelé la racine de D_n, donc D_n possède $2^{n+1} - 1$ sommets.

Proposition 2. *[5] Pour tout $n \geq 2$, l'arbre D_n est plongeable dans $Q_n + 2$ $dim(D_1) = 2$ et $dim(D_n) = n + 2$.*

A partir de l'arbre binaire complet D_n, on définira d'autres arbres plongeables dans l'hypercube.

1. Pour $n \geq 1$ on désigne par $\widehat{\widehat{D}}_n$ l'arbre formé à partir de deux copies disjointes de D_n, tel que leurs racines sont reliées par une arête appelée arête axiale. $\widehat{\widehat{D}}_n$ a $2^{n+2} - 2$ sommets.

   Proposition 3. *[5]*
   Pour tout $n \geq 1$, l'arbre $\widehat{\widehat{D}}$ est plongeable dans $Q_n + 2$; $dim(\widehat{\widehat{D}}_n) = n + 2$.

2. Soit $n \geq 1$, on désigne par $\widehat{D}_n$ l'arbre formé à partir de $\widehat{\widehat{D}}_n$ en insérant deux nouveaux sommets au niveau de l'arête axiale, et la chaîne obtenue partir de l'arête axiale sera appelée chaîne axiale de $\hat{D}_n$. L'arbre $\check{D}_n$ peut être défini à partir de $\widehat{\widehat{D}}_n$ en insérant deux nouveaux sommets de degré 2 au niveau d'une arête pendante de $\widehat{\widehat{D}}_n$. Il est clair que $\check{D}_n$ et $\widehat{D}_n$. possèdent le même nombre de sommets. $\widehat{D}_n$. possède deux sommets de degré 2, 2^{n+1} sommets pendants et $2^{n+1} - 2$ sommets de degré 3.

   Proposition 4. *[13]*
   Pour tout $n \geq 1$, $dim(\check{D}_n) = dim(\hat{D}_n) = n + 2$.

3 Plongement et placement de certaines classes d'arbres

3.1 La classe AD_n

Pour $n \geq 1$ l'arbre AD_n est obtenu à partir de l'arbre binaire D_n en reliant un seul sommet de degré 1 à nouveau sommet. AD_n possède donc 2^{n+1} sommets. AD_3 est montré dans la figure suivante :

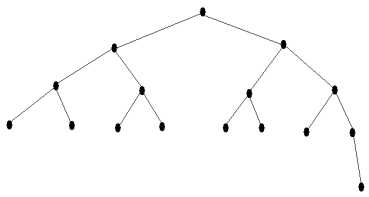


Fig. 1.

Théorème 1. *Pour tout* $n \geq 3$, $dim(AD_n) = n + 1$.

Démonstration. Il est clair que $D - n$ est plongeable dans AD_n, et que AD_n est plongeable dans $\widehat{\widehat{D}}_n$, donc $dim(D_n) \leq dim(AD_n) \leq dim(\widehat{\widehat{D}}_n)$, mais $dim(D_n) = dim(\widehat{\widehat{D}}_n) = n + 2$, alors $dim(AD_n) = n + 2$.

3.2 La classe $A\widehat{D}_n$

Pour $n \geq 1$ l'arbre $A\widehat{D}_n$ est obtenu à partir de deux copies disjointes de AD_n, en reliant les racines par une arête. AD_2 est donné par le graphe suivant :

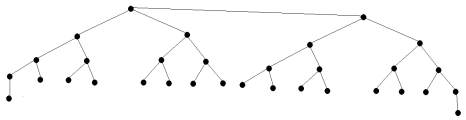


Fig. 2.

Théorème 2. *L'arbre $A\widehat{D}_n$ est plongeable dans Q_{n+2} et $dim(A\widehat{D}_n) = n + 2$*

Démonstration. dans cette dmonstration on va utiliser les chaînes ainsi que la notion de la $C_n$valuation, il est clair que toute chaîne de l'arbre de $A\widehat{D}_n$ est aussi dans l'arbre $\widehat{\widehat{D}}_n$, donc toute chaîne de $A\widehat{D}_n$ est C_{n+2}valuation, car $dim(\widehat{\widehat{D}}_n) = n + 2$. donc $A\widehat{D}_n$ est plongeable dans Q_{n+2} est $dim(A\widehat{D}_n) = n + 2$ car $A\widehat{D}_n$ possède 2^{n+2} sommets et ne peut pas être plongeable dans Q_{n+1}.

On peut parler d'un autre plongement concernant ce type d'arbre qui nécessite de trouver combien d'arbre de même topologie qu'on peut plonger dans un hypercube de dimension donnée.

On peut faire une généralisation comme suit :
pour n=1, l'arbre $A\widehat{D}_1$ est donn dans la figure suivante :

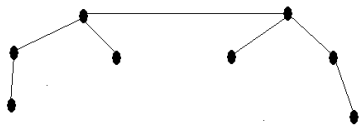


Fig. 3.

Pour $k \geq 2$, l'arbre $A^k\widehat{D}_1$, il obtenu en reliant deux copies de $A^{k-1}\widehat{D}_1$ en reliant un sommet de degré 3 de $A^{k-1}\widehat{D}_1$ à un sommet de degré 3 de $A^{k-1}\widehat{D}_1$ par une arête.
Il est clair que l'arbre $A\widehat{D}_1$ est plongeable dans Q_3, comme Q_4 est obtenu à partir de deux copies de Q_3, donc là si on va prendre seulement l'arête reliant les deux copies de $A\widehat{D}_1$, et les arêtes formant dans chaque Q_3 l'arbre $A\widehat{D}_1$, on obtient l'arbre $A^2\widehat{D}_1$ plogeable dans Q_4, et que qu'on va supprimer l'arête reliant les deux copies de $A\widehat{D}_1$. On fait de la même manière pour $A^3\widehat{D}_1$ est plongeable dans Q_5, car dans chaque Q_4, il y a une copie de $A^2\widehat{D}_1$, et comme Q_5 est obtenu à partir de deux copies de Q_4, alors si on va prendre seulement l'arête reliant seulement les deux copies de $A^2\widehat{D}_1$, on prouve facilement que $A^3\widehat{D}_1$ est plongeable dans Q_5. Faisant de la même manière, on montrera facilement que $A^{k-1}\widehat{D}_1$ est plongeable dans Q_{k+2} avec $K \geq 2$. Par la suite on va supprimer les arêtes formant les arbres $A^k\widehat{D}_1$, $k \geq 2$ on aura donc :

- Q_4 peut contenir au maximum$2 = 2^{(4-3)}$ copies de $A\widehat{D}_1$
- Q_5 peut contenir au maximum $4 = 2^{(5-3)}$ copies de $A\widehat{D}_1$
- Q_6 peut contenir au maximum $8 = 2^{(6-3)}$ copies de $A\widehat{D}_1$
- Donc Q_n peut contenir au maximum 2^{n-3} copies de $A\widehat{D}_1$.

Cette méthode nous permet de donner le résultat suivant :

Proposition 5. *Le nombre de maximum de copie de l'arbre $A\widehat{D}_1$, qu'on peut placer dans un hypercube de dimension n est 2^{n-3}.*

Pour la démonstration il suffit d'utiliser la récurrence sur n.

3.3 La classe AB_n

Pour $n \geq 1$ l'arbre AB_n est obtenu de la manière suivante :
AB_1 est le graphe de la figure suivante :

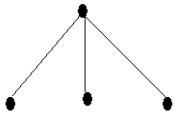


Fig. 4.

Pour $n \geq 2$, AB_n est obtenu en reliant deux copies disjointes de AB_{n-1}, tel que un sommet de degré $n+1$ de la première copie est relie par une arête à un sommet de degré $n+1$ de la deuxième copie.
AB_2 est montré dans la figure suivante :

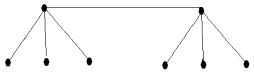


Fig. 5.

AB_3 est montré dans la figure suivante :

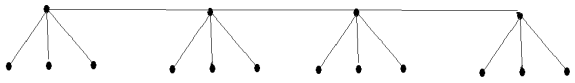


Fig. 6.

AB_4 est montré dans la figure suivante :

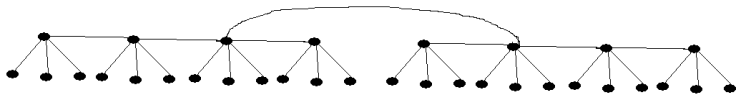


Fig. 7.

Théorème 3. *Pour tout* $n \geq 2$, $dim(AB_n) = n + 3$.

Démonstration. Cette démonstration se fait par construction, il est clair que AB_1 est plongeable dans Q_3, alors si on prend les arêtes de AB_1 dans chaque copie de Q_3 est parmi les arêtes qui forment Q_4 seulement l'arête reliant les deux copies de AB_1 pour obtenir AB_2, donc il est clair que AB_2 est plongeable dans Q_4, on fait la même chose pour AB_3, si on va prendre parmi les arêtes qui forment Q_5 seulement L'arête reliant les deux copies de AB_2, on montrera facilement que AB_3 est plongeable dans Q_5, de la même manière donc on va montrer que AB_n est plongeable dans Q_{n+2}, maintenant il reste à montrer que $dim(AB_n) = n + 2$. Comme le degré maximum de AB_n est n+2, donc AB_n ne peut pas être plongeable dans Q_{n+1} d'où $dim(AB_n) = n + 2$.

On peut parler d'un autre plongement concernant ce type d'arbre qui nécessite de trouver combien d'arbre de même topologie qu'on peut plonger dans un hypercube de dimension donnée.

il est clair que Q_3 peut contenir deux copies de AB_1 de telle sorte que les sommets de la première copie ne sont pas reliés aux sommets de la deuxième copie, comme le montre le graphe de la figure suivante :

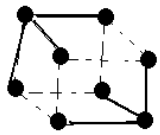


Fig. 8.

Même chose Q_4 peut contenir deux copies de AB_2, comme le montre le

graphe de la figure suivante :

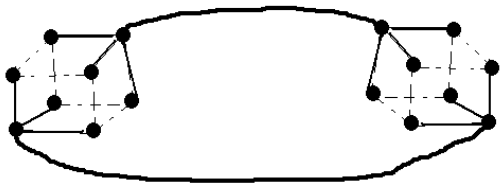


Fig. 9.

On peut aussi le montrer pour Q_5 par le graphe suivant :

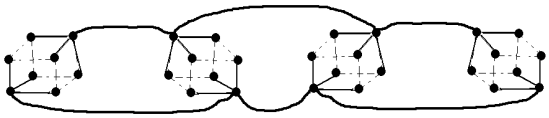


Fig. 10.

Pour Q_4 le fait que Q_3 contient deux copies disjointes de AB_1, et que Q_4 est obtenu à partir de deux copies disjointes de Q_3, alors si on va prendre parmi les arêtes formant Q_4 seulement les arêtes reliant les copies de AB_1, on montre que Q_4 va contenir deux Copies disjointes de AB_2, même chose pour Q_5 va contenir deux copies de AB_3 et d'une manière générale Q_n va contenir deux copies de AB_{n-2} pour tout $n \geq 3$.
D'aprés les 3 figures ci-dessus on a :

- Q_3 peut contenir au maximum $2 = 2^{(3-2)}$ copies de AB_1,
- Q_4 peut contenir au maximum 2 copies de AB_2, donc au maximum $4 = 2^{(4-2)}$ copies de AB_1,
- Q_5 peut contenir au maximum 2 copies de AB_3, donc au maximum $8 = 2^{(5-2)}$ copies de AB_1
- D'où donc Q_n peut contenir au maximum 2^{n-2} copies de AB_1. Cette méthode nous permet de donner le résultat suivant :

Proposition 6. *Le nombre maximum de copies de AB_1, qu'on peut placer dans un hypercube de dimension n, est 2^{n-2}.*

Pour la démonstration il suffit d'utiliser la récurrence sur n.

Références

1. Arfati, J. Papadimitriou, C.H. and Papageorgiou, P. : The complexity of cubical graphs. proceedings of 11 th international Kolloquium on automata , languages and programming. (1984) 51-57.
2. Berrachedi, A. : Sur la dimension cubique de quelques classes d'arbres. Actes du Colloque Cosi'04, Colloque sur L'optimisation et les Systèmes d'Information. Université de Tizi-Ouzou.
3. Bezrukov, S. and Monien, B. Unger, W. and Wechsung, G. : Embedding ladders and caterpillars into hypercube. discrete applied mathematics , **83** (1992) 21-29.
4. Corneil, D.G. and Wagner, A. : Embeding trees in a hypercube is NP- complet. siam j. comput **19** (1990),570-590
5. Havel, I. : On hamiltonian circuits and spanning trees of hypercubes. Cas prest. Mat **109** (1984) 135-152.
6. Havel, I. and Liebel, P. : One legged caterpillars spans hypercubes. Journal of graph theory. **10** (1986) 69-77
7. Havel, I. and Moravek, J. : B-valuation of graphs . Czech- Math .jour ., **22** (1972),338-351.
8. Firsov, V. : On isometric embeddings of graph into a boolean cube. cyber - netics 1, (1965) 112-113.
9. Harary, F. Lewinter, M. and Widolski, W. : On two legged caterpillars which span a hypercube. Congr. Numer. **66** (1988) 103-108.
10. Kabyl, k. : Dimension cubique de deux nouvelles classes d'arbres. Actes du Colloque Cosi'05, Colloque sur L'optimisation et les Systèmes d'Information. Université de Béjaia.
11. kobeissi, M. and Mollard, M . : Spanning graphs of hypercubes starlike and double starlike trees. Accept discrete Math.
12. Labord, J.M. and Rao hebbar, S.P. : Another characterisation of hypercube . discrete Math., **39** , (1982) 161-166.
13. Nebesky, L. : Embedding m-quasistars into n-cubes. C zechoslovak mathematical, journal, praha,38 (113),1988.
14. Nekri, M. and Berrachedi, A. : Two new classes of Trees Embeddable into hypercubes. RAIRO Oper. Res., **38**, (2004) 295303.

Flow shop Problem with transportation considerations

Nacira Chikhi and Mourad Boudhar

Faculty of Mathematics, USTHB University,
BP 32 Bab-Ezzouar, El-Alia 16111, Algiers , Algeria
nacira_chikhi@hotmail.fr
mboudhar@yahoo.fr
http://www.usthb.dz

Abstract. Motivated by applications in manufacturing systems. This paper deals with a scheduling problem of independent tasks with additional constraints (transportation times), where the objective is to minimize the total completion time. This problem arises in automated cells and is a complex flow shop problem with a transportation robot or conveyor. Since the problem is NP-hard, heuristics are developed to give near optimal solutions. Two new programming algorithms are also proposed for solving some special cases of this problem. Finally, we evaluate the proposed heuristics, giving experimental results on randomly generated test problems.

Key words: Scheduling, flow shop, tranport, makespan, heuristics.

1 Introduction

In most manufacturing systems, semi-finished jobs are transferred from one facility to another for further processing through material handling systems such as automated guided vehicles (AGVs) and conveyors. In the last four decades, many books and numerous papers have been published in the area of machine scheduling. However, most of the published literature explicitly or implicitly assumes that either there is an infinite number of transporters or that jobs are transported instantaneously from one location to another without transportation time involved.

These displacements were not therefore taken in account at the time of the construction of the scheduling. However this assumption is often not justified in practice, there are many situations in which it must not be abandoned as being unrealistic. For example, in computer systems the output of a job on one processor may require a communication time so as to become the input to a succeeding job on another processor and in manufacturing systems , there may be a transportation time from one production facility to another. This model can be illustrated in the case of robotic cells that are found in manufacturing systems of semiconductors or textiles, in which an automated guided vehicle is charged to displace jobs. It can also be illustrated by the example of a workshop for

electroplating whose process consists of coating a part by a thin layer of metal on pieces. The displacement of pieces is done mainly by a transporter(hoist) moving horizontally on a rail as it is shown in Figure 1.

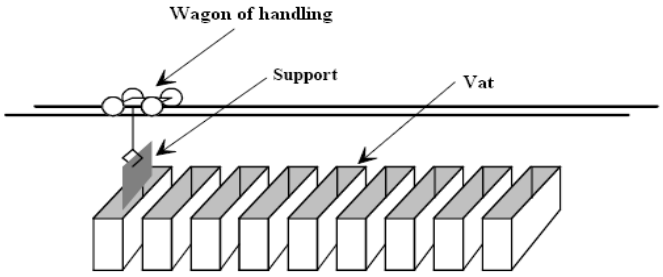


Fig. 1. Industrial application

In this paper, our problem can be defined as follows. We are given a set J of n independent jobs $J = \{J_1, ..., J_n\}$ to be processed on 2 machines M_1 and M_2 in a flow shop with unlimited buffer spaces on both machines. Each job must first be processed on machine M_1, then machine M_2.

The processing time of a job J_i on machine M_k is p_{ik}. We assume that all of the jobs start at machine M_1. Once a job is processed on machine M_1, it is transported to machine M_2 by a transporter. The transporter is initially located at machine M_1. It has a capacity of c, i.e. it can carry up to c jobs in one shipment.

The transportation time from machine M_1 to machine M_2 is denoted by t (the round-trip requires 2t). We assume that the loading and the unloading times are included in the processing times of jobs and are not considered separately. Our goal is to schedule the jobs so as to minimize the makespan .

We follow the commonly used three-field notation $\alpha/\beta/\gamma$ for machine scheduling problems. In the α field, we use notation "TF" to denote a flow-shop problem with transportation between machines. Hence, $TF2/\upsilon = x, c = y/C_{max}$, represents the 2-machine flow shop problem with x transporters, each with capacity y. So our problem thus defined is denoted $TF2/\upsilon = 1, c \geq 1/C_{max}$

Chen and Lee [1] studied a two-machine flow-shop problem with several conveyors of any capacity noted $TF2/\upsilon \geq 1, c \geq 1/C_{max}$. They gave a dynamic algorithm that solves in polynomial time the problem $TF2/p_{i1} = p, \upsilon \geq 1, c \geq 1/C_{max}$. In our work, we limit to only one conveyor and we give two polynomial algorithms for the solution of two particular cases of the general problem $TF2/\upsilon = 1, c \geq 1/C_{max}$. We also propose heuristics to solve the general problem which is NP-hard[1].

This article is organized as follows: the second section is devoted to the mathematical formulation, a mixed integer linear programming model is proposed to determine the schedule with minimum makespan. This model has been tested using CPLEX solver. Section 3 is dedicated to the calculation of lower bounds and the presentation of some subproblems that can be solved polynomially. As for the fourth section some heuristics are presented for the solution of the general problem and some numerical tests are carried out to show the performance and the efficiency of the different heuristics in the last section. Finally, we provide a conclusion at the end of this article.

2 Problem transformation

The following notation is used for the mathematical representation of the general problem.
d_{i1}: is the starting time of the execution of the first operation of the job i on the machine M_1. d_{is}: is the starting time of the transport of the job i and d_{i2}: is the starting time of the execution of the second operation of the job i on the machine M_2
For every couple (i, j) of jobs, we introduce the following binary variable: a_{ij} equal to 1 if $d_{i1} < d_{j1}$, and 0 otherwise. b_{ij} equal to 1 if $d_{i2} < d_{j2}$, and 0 otherwise. α_{ij} equal to 1 if $d_{is} < d_{js}$, and 0 otherwise.
The objective function is to minimize C_{max};
subject to:

$$
\begin{array}{lll}
a_{ij} + a_{ji} = 1 & \forall\ i,j = \overline{1,n}\ ;\ i<j\ \ and\ \ i \neq j & (1)\\
d_{i1} + p_{i1} - d_{j1} \leq (1 - a_{ij}).M & \forall\ i,j = \overline{1,n}\ \ and\ \ i \neq j & (2)\\
d_{is} \geq d_{i1} + p_{i1} & \forall\ i = \overline{1,n} & (3)\\
\alpha_{ij} + \alpha_{ji} \leq 1 & \forall\ i,j = \overline{1,n}\ ;\ i<j\ \ and\ \ i \neq j & (4)\\
d_{js} - d_{is} \geq 2t * \alpha_{ij} - \alpha_{ji} * M & \forall\ i,j = \overline{1,n}\ \ and\ \ i \neq j & (5)\\
d_{is} - d_{js} \geq 2t * \alpha_{ji} - \alpha_{ij} * M & \forall\ i,j = \overline{1,n}\ \ and\ \ i \neq j & (6)\\
\sum_{j=1,i\neq j}^{n} (1 - \alpha_{ij} - \alpha_{ji}) \leq c - 1 & \forall\ i = \overline{1,n} & (7)\\
d_{i2} \geq d_{is} + t & \forall\ i = \overline{1,n} & (8)\\
b_{ij} + b_{ji} = 1 & \forall\ i,j = \overline{1,n}\ ;\ i<j\ \ and\ \ i \neq j & (9)\\
d_{i2} + p_{i2} - d_{j2} \leq (1 - b_{ij}).M & \forall\ i,j = \overline{1,n}\ \ and\ \ i \neq j & (10)\\
d_{i2} + p_{i2} \ \leq \ C_{max} & \forall\ i = \overline{1,n} & (11)\\
a_{ij},\ \ b_{ij},\ \ \alpha_{ij} \in \{0,1\} & \forall\ i,j = \overline{1,n} & (12)\\
d_{i1},\ \ d_{is},\ \ d_{i2} \in \mathbf{N} & \forall\ i = \overline{1,n} & (13)
\end{array}
$$

Where M is a very large value

Constraints (1), (2) and (3) concern the first machine: Constraints (1) mean that for any two jobs J_i and J_j, either J_i precedes J_j on the first machine, or J_j precedes J_i. Constraints (2) require that the first machine executes only one job at a time and (3) assure that a job can not be transported from the first machine to the second machine, once the first operation of this job is finished.

Constraints (4), (5), (6) and (7) are constraints on the conveyor (vehicle) and on the jobs to transport: Constraints (4) express that all jobs must be

transported between the two machines. Constraints (5) and (6) indicate that any job J_i is transported from the first machine to the second machine either before or after another job J_j, or at the same time and show that the transport time of a round-trip of the vehicle requires 2t. The constraints (7) express that the number of transported jobs at any time must be smaller than the vehicle capacity.

Constraints (8), (9) and (10) concern the second machine. Constraints (8) induce that the execution of the second operation of a job can only begin once the job has arrived to the second machine. Constraints (9) express that all jobs must be executed by the second machine. Constraints (10) assure that the second machine executes only one job at a time. The constraints (11) imply that the end of execution of any job is lower or equal to the makespan. Constraints (12) and (13) indicate the type of variables.

The number of variables and the number of constraints of a mathematical model are indications that measure its dimension and the efficiency of the modeling. The number of variables of our model is $3n^2$ and the number of constraints is $n(17n-1)/2$.

From this formulation, we can derive a lower bound by relaxing the constraints (12) and (13). The relaxed problem can be solved using a linear programming solver (CPLEX for example). The inconvenience of this technique consists in a large number of constraints ($n(23n-1)/2$ constraints).

3 Testing of the model with CPLEX

Table 1. Results obtained by the Cplex solver.

n	c	t	Pbms	avr-time
5	2	1	20	0.183
5	2	5	20	0.4725
5	3	1	20	0.715
5	3	5	20	0.6665
6	2	1	20	0.185
6	2	5	20	103.02
6	3	1	20	0.1815
6	3	5	20	0.1807
7	2	1	20	1.03
7	2	5	20	106.755
7	3	1	20	2.064
7	3	5	20	5.459
8	2	1	20	18.489
8	2	5	20	105.65
8	3	1	20	15.363
8	3	5	20	170.12
9	2	1	13	189.32
9	2	5	8	200.56
9	3	1	10	1030.245
9	3	5	6	186.342
10	2	1	5	1045.32
10	2	5	4	1230.458
10	3	1	4	7456.23
10	3	5	3	5131.47
50	20	1	1	6597,8
50	20	5	1	12588,78
70	30	1	1	14265.236
70	30	5	1	24698.24

The linear models with integer and binary variables can be solved by efficient solvers such as LINGO, CPLEX, etc. Our mathematical model has been tested

on a Pentium IV 3.06 GHz Personal Computer with 512 Mo RAM using Cplex Solver. The processing times p_{i1} and p_{i2} are generated by a uniform law in $[1, 100]$. We fixed the number of jobs and let the vehicle capacity and the transportation time vary. For every case, 20 problems are solved and the average execution time (for which the optimal solution is obtained) is computed in seconds. The results are given in the Table 1.

Computational experiments show that the largest problem that can be solved within at least 18 minutes is a two-machine and nine-jobs problem and the largest problem that can be solved within at least 7 hours is a two-machine and seventy-jobs problem.

4 Study of bounds and some subproblems

We proposed two lower bounds LB_1 and LB_2 for the objective function :

- $LB_1 = \max\{(\lceil \frac{n}{c} \rceil - 1) * 2t + t + \min_{1 \leq i \leq n} \{p_{i1}\} + \min_{1 \leq i \leq n} \{p_{i2}\}, \max_{1 \leq i \leq n} \{p_{i1} + p_{i2}\} + t\}$.
- $LB_2 = \max\{\sum_{1 \leq i \leq n} p_{i,1} + \min_{1 \leq i \leq n} \{p_{i2}\} + t, \sum_{1 \leq i \leq n} p_{i2} + \min_{1 \leq i \leq n} \{p_{i1}\} + t\}$.

We studied some subproblems of the general problem $TF2/\upsilon = 1, c \geq 1/C_{max}$. We mention especially the following cases:

Case 1: $p_{i1} \geq 2t, p_{i2} \geq \max_{1 \leq i \leq n} \{p_{i1}\}$

Algorithm 1

Begin

1: Arrange and process jobs in the increasing order (SPT rule) in relation to p_{i1}
2: (In every batch, we have only one job).

End

Theorem 1. *The algorithm 1 resolve the two problems TF2/*$p_{i1} \geq 2t, p_{i2} \geq \max_{1 \leq i \leq n} \{p_{i1}\}, \upsilon = 1, c \geq 1/C_{max}$ *and* $TF2/t \leq p_{i1} \leq \frac{3}{2}t, p_{i2} \geq 2t, \upsilon = 1, c \geq 1/C_{max}$ *in* $O(n \log n)$.

Case 2: $p_{i1} \leq \frac{2t}{c}, p_{i2} \geq 2t$

Algorithm 2

Begin

1: Find a job J_j of J having the minimum execution time on the first machine M_1.
2: Process the job J_j in first position and transport it alone in a first batch.
3: $J := J \setminus \{J_j\}$.
4: Arrange the jobs of J in the decreasing order (LPT rule) in relation to p_{i2} and process them in this order after the first job .

End

Theorem 2. *The algorithm 2 gives an optimal solution for the two problems* $TF2/p_{i1} \leq \frac{2t}{c}, p_{i2} \geq 2t, \upsilon = 1, c \geq 1/C_{max}$ *and* $TF2/p_{i1} \leq \frac{2t}{c}, p_{i2} \geq \frac{2t}{c}, p_{min2} \geq 2t, \upsilon = 1, c \geq 1/C_{max}$ *in* $O(n \log n)$.

Case 3: The problem $TF2/p_{i1} = p, \upsilon = 1, c \geq 1/C_{max}$:
When execution times on the first machine are identical and execution times on the second machine are any, the problem $TF2/p_{i1} = p, \upsilon = 1, c \geq 1/C_{max}$ is polynomial and can be solved by the dynamic algorithm of Chen and Lee [1]. The problem $TF2/p_{i2} = p, \upsilon = 1, c \geq 1/C_{max}$ is also polynomially solvable by the dynamic algorithm of Chen and Lee

5 Heuristics

Recall that in general the problem $TF2/\upsilon = 1, c \geq 1/C_{max}$ is NP-hard, so we propose some heuristics for its solution. We have used several rules of priority, based on the notion of priority between jobs to process. Their main advantages are, in general, their simplicity and especially their speed. Five rules have been applied therefore for the scheduling of jobs on the two machines. The two rules R_1 and R_2 are based on the coupling of jobs. The third rule of investment, is based on the algorithm of Jonhson [**?**]. We improved it in order to take into account the transportation times. Finally, we use the two rules SPT (Shortest Processing Time) and LPT (Longest Processing Time) that are based on the arranging of jobs.

These heuristics are also based on the following procedure that allows the construction of different batches. The principle of this procedure is to choose a maximal set of jobs that follows the job J_i (according to the initial order) so as the sum of the these execution times in this set, on the first machine is lower or equal to the time of the round-trip of the vehicle 2t plus a small amount of time (τ). Once this set of jobs is found, these jobs will be transported with the job J_i in one batch. On the other hand, if such a set doesn't exist, the job J_i will be transported alone in a batch.

Procedure of contruction of batches
Begin

1: $i := 1$, $\ell := 1$;
2: **while** $i < n$ **do**
3: **if** $p_{i+1,1} \geq 2t$ **then**
4: $B_\ell := \{J_i\}$, $\ell := \ell + 1$, $i := i + 1$.
5: (B_l represents the batch Number l);
6: **if** i=n **then**
7: $B_\ell := \{J_n\}$, $\ell := \ell + 1$, $i := i + 1$;
8: **end if**
9: **else**
10: Find $J_{i+1}, ..., J_k$ in J as:
11: $\sum_{j=i+1}^{k} p_{j,1} \leq 2t + \tau$ and $k - i + 1 \leq c$
12: $B_\ell := \{J_i, J_{i+1}, ..., J_k\}$, $\ell \leftarrow \ell + 1$, $i := k + 1$;
13: **if** i=n **then**
14: $B_\ell := \{J_n\}$, $\ell := \ell + 1$, $i := i + 1$;
15: **end if**
16: **end if**

17: **end while**
18: $L := \ell - 1$, $d_1 := \sum_{j \in Bch_1} p_{j,1}$, $r_1 := \sum_{j \in Bch_1} p_{j,1}$
19: $s := \sum_{j \in Bch_1} p_{j,1}$, $c_1 := \sum_{jinBch_1} p_{j,1} + \sum_{j \in Bch_1} p_{j,2} + t$
20: **for** $k = 2$to L **do**
21: $r_k := s + \sum_{j \in Bch_k} p_{j,1}$
22: $d_k := max\{r_k, d_{k-1} + 2t\}$
23: $s := r_k$
24: $c_k := max\{d_k + t, c_{k-1}\} + \sum_{j \in Bch_k} p_{j,2}$,
25: **end for**
26: $C_{max} := c_L$.

End

The first heuristic named H_1 is based on a new rule R_1. This rule forms pairs (J_k, J_j) such that the job J_k have the shortest execution times on M_1 and the job J_j have the longest execution times on M_2. We build a sequence of jobs reassembling all the couples. Finally, we construct the set of batches.

Algorithm H_1

Begin

1: **while** The list of jobs J is not empty **do**
2: Find a job J_k having the shortest execution time on the first machine.
3: $J := J \setminus \{J_k\}$.
4: Find a job J_j having the longest execution time on the second machine
5: $J := J \setminus \{J_j\}$
6: **end while**
7: Apply the previous procedure of batch construction according to the order of jobs determined by the previous loop.

End

Another version of the heuristic H_1 is denoted H_2. It has the same principle as H_1 except that it is based on another new rule named R_2 which forms couples (J_k, J_j) in which the jobs J_k have the longest time of execution on the second machine M_2 and the jobs J_j have the shortest execution times on first machine M_1. Once the pairs are created, we arrange them in the same order and we form a sequence of jobs. Finally, we apply the procedure of the construction of batches on the sequence of jobs obtained.

Algorithm H_2

Begin

1: **while** The list of jobs J is no empty **do**
2: Find a job J_k having the longest execution time on the second machine;
3: $J := J \setminus \{J_k\}$;
4: Find a job J_j having the shortest execution time on the first machine;
5: $J := J \setminus \{J_j\}$;
6: **end while**
7: Apply the procedure of construction of batches according to the order of jobs determined by the previous loop.

End

We propose another heuristic H_3 based on the LPT rule.

Algorithm H_3
Begin

1: Find a job J_k having the shortest execution time on the first machine;
2: Process the job J_k in first position and transport it alone in a first batch.
3: $J := J \setminus \{J_k\}$;
4: Arrange the remaining jobs of T in the decreasing order relative to the execution times on the second machine and process them after the first job.
5: Apply the procedure of construction of batches on the jobs in the order as in J.

End

The fourth heuristic that we propose named H_4 is based on the LPT rule, that consists to arrange jobs in the decreasing order relative to the execution times. In our case, we apply this rule for the job execution times on the machine M_2.

Algorithm H_4
Begin

1: Arrange jobs according to the decreasing order relative to the execution times on the second machine.
2: Apply the procedure of construction of batches.

End

Algorithm H_5
Begin

1: Build a two machines pseudo problem with execution times on the first machine $p'_{i1} = \max(p_{i1}, 2t)$ and $p'_{i2} = p_{i2}$ on the second machine (p_{i1} and p_{i2} are the execution times of the initial flow-shop).
2: Apply the algorithm of Johnson to this pseudo problem to get an ordering of jobs
3: Apply the procedure of construction of batches according to this order.

End

6 Tests according to the uniform law

Until now, there is no method in the literature which treat precisely the problem $TF2/\upsilon = 1, c \geq 1/C_{max}$. So we can not make a comparison with the heuristics that we propose.

However, we have tested the developed methods by using several instances generated randomly according to the uniform law. We have coded our algorithms in Delphi 7 and have run them on a Pentium IV 3.06 GHz Personal Computer with 512 Mo RAM.

We generated 100 instances for each number of jobs and we applied the heuristics cited above on these instances. Some results obtained for the uniform law are summarized in table 2 which follows, where we give the percentage with the best completion time where the solution found by the heuristic is better as compared to the other solutions, the percentage where the makespan is equal to

the lower bound and the average of performance ratio of the heuristics and the average execution time of each heuristic (in milliseconds).

For the first case (a), we suppose that the job execution times on the two machines as well as the vehicle capacity follow a uniform distribution in $[1, 10]$ and the transportation times follow a uniform law in the interval $[1, 100]$.

In the second case (b), we suppose that the job execution times on the two machines follow a uniform law in the interval $[1, 50]$ and the transportation time as well as the vehicle capacity have a uniform distribution in $[1, 10]$. The obtained results are represented in the Table 2.

Table 2. Summary of the tests.

		(a) $pi1, pi2, c \in \overline{1,10}$, $t \in \overline{1,100}$					**(b)** $pi1, pi2 \in \overline{1,50}$, $t, c \in \overline{1,10}$				
	$\tau = 0$	H_1	H_2	H_3	H_4	H_5	H_1	H_2	H_3	H_4	H_5
n=10:	pC_{max}	28%	24%	59%	15%	5%	36%	19%	76%	39%	15%
	Opt	7%	1%	18%	3%	2%	30%	16%	62%	36%	7%
	av-tim	4,71	3,44	6,37	3,71	5.28	6,46	4,35	6,12	3,74	8.01
	av-Rat	1,27	1,25	1,39	1,35	1,29	1,021	1,064	1,016	1,041	1,058
	Mx-Rat	1,89	1,96	2,32	2,15	2.44	1,133	1,215	1,155	1,206	1,188
n=50:	pC_{max}	21%	9%	74%	4%	0%	35%	17%	74%	39%	7%
	Opt	7%	1%	10%	4%	0%	30%	10%	66%	37%	4%
	av-tim	10,02	9,39	12,51	8,43	12.72	11,24	8,6	14,01	8,6	13,45
	av-Rat	1,070	1,077	1,074	1,078	1.099	1,006	1,009	1,004	1,012	1,014
	Mx-Rat	1,208	1,227	1,189	1,309	1.309	1,037	1,047	1,032	1,037	1,034
n=100:	pC_{max}	37%	13%	44%	2%	1%	31%	19%	72%	37%	1%
	Opt	9%	0%	10%	0%	0%	20%	12%	59%	33%	1%
	av-tim	19,04	15,79	21,2	15,81	21.09	18.09	15.8	21.55	15.58	21.78
	av-Rat	1,026	1,023	1,035	1,026	1.033	1.002	1.006	1.002	1.006	1.007
	Mx-Rat	1,094	1,107	1,086	1,099	1.158	1,017	1,019	1,014	1,021	1.024
n=1000:	pC_{max}	54%	30%	32%	9%	0%	36%	31%	62%	40%	5%
	Opt	6%	1%	5%	0%	0%	32%	27%	55%	39%	1%
	av-tim	152,73	138,91	174,2	140,17	169.03	158,9	146,2	180,3	146,1	179.9
	av-Rat	1,003	1,004	1,004	1,002	1.004	1,000	1,000	1,000	1,000	1.001
	Mx-Rat	1,052	1,076	1,026	1,006	1.033	1,001	1,001	1,001	1,001	1.001

We define the calculated parameters:

pCmax: is the percentage for which the heuristic H provides a better solution than the other heuristics.

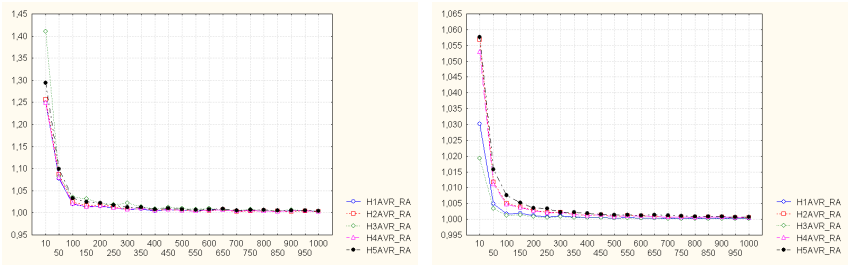
opt: is the percentage for which the solution obtained by the heuristic H coincides with the lower bound LB.

Ratio(H): is the performance ratio of the heuristic H, $Ratio(H) = \frac{Sol(H)}{LB}$.

av-Rat: is the average performance ratio, $average - Ratio(H) = \frac{\sum_{k=1}^{100} Ratio_k(H)}{100}$, k is the number of the instance.

mx-Rat: the maximum of the performance ratio.
av-tim: the average of the execution time.
Dev(H): is the deviation of the heuristic H, Dev(H)=$\frac{Sol(H)-LB}{LB}$.
avr-Dev(H): is the average deviation of the heuristic H, $avr - Dev(H) = \frac{\sum_{k=1}^{100} Dev_k(H)}{100}$.

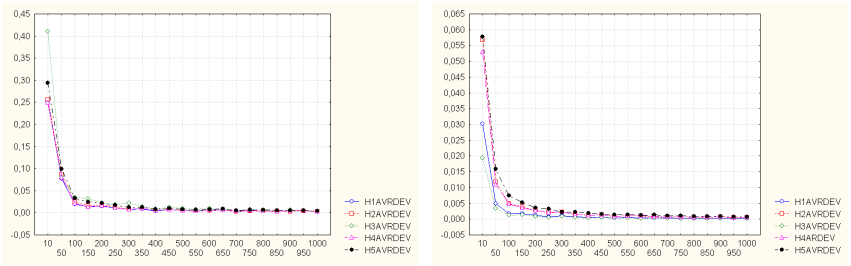
For the different heuristics, we established the average of the performance ratio by applying the heuristic on instances randomly built according to a uniform law. The results are illustrated in Figures 2 and 3.



(a) $pi1, pi2, c \in \overline{1,10}$, $t \in \overline{1,100}$ **(b)** $pi1, pi2 \in \overline{1,50}$, $t, c \in \overline{1,10}$

Fig. 2. Average performance ratio of the heuristics according to "n"

With regard to the average deviations, the obtained graphs are shown in Figure 9.



(a) $pi1, pi2, c \in \overline{1,10}$, $t \in \overline{1,100}$ **(b)** $pi1, pi2 \in \overline{1,50}$, $t, c \in \overline{1,10}$

Fig. 3. Average deviations of the heuristics according to "n"

We generated 100 instances for each number of jobs and we applied the heuristics on these instances ($pi1, pi2 \in [1,50]$, $t, c \in [1,10]$) in the cases $\tau = 0$ and $\tau = 1$. The results are given in Table 3.

Table 3. Comparison between the cases $\tau = 0$ and $\tau = 0$.

(a) $\tau = 0$

	$\tau = 0$	H_1	H_2	H_3	H_4	H_5
n=10:	av-Rat	1,021	1,056	1,016	1,041	1,055
	av-tim	1,09	0,47	1.26	0.92	1.72
n=1000:	av-Rat	1,000	1,000	1,000	1,000	1,001
	av-tim	61,04	60,89	87,47	43,33	86,23

(b) $\tau = 1$

$\tau = 1$	H_1	H_2	H_3	H_4	H_5
n=10	1.030	1,059	1,019	1,055	1,059
	1.25	1,1	1,53	0,16	1.39
n=1000	1,000	1,001	1,000	1,000	1,001
	64,71	59,91	85,16	42,85	83,94

We note that when $\tau = 0$, the results are better than in the case $\tau > 0$.

Table 4. Comparison with optimal solutions

n	c	t	Opt	time(s)	H_3C_{max}	time(m.s)	bC_{max}	$R(H_3)$
5	3	5	29	2,14	33	16	32	0.137
			38	0,59	44	0	39	0.157
			31	0,41	39	0	31	0.258
			31	0,03	32	16	32	0.032
			42	0,05	44	16	44	0.047
			38	0,53	39	15	39	0.026
			30	2,19	34	16	30	0.133
			32	0,11	42	16	33	0.312
			34	0,05	40	0	35	0.176
			39	1,94	46	15	41	.0179
10	3	1	55	0,05	57	0	55	0.036
			58	0,03	58	0	58	0
			53	0,02	55	0	53	0.037
			64	0,03	64	0	64	0
			52	0,03	57	16	53	0.09
			59	0,05	59	0	59	0
			49	0,03	49	0	49	0
			59	0,05	59	15	59	0
			65	0,05	65	31	65	0
			54	0,03	54	62	54	0
			64	0,06	64	16	64	0

For the different methods developed, there are not any precise conditions so that a method is better than another one. It depends on the number of jobs and the transportation time. However, we have compared the best solutions generated by all the heuristics denoted bC_{max} and the solutions generated by the heuristic H_3 denoted H_3C_{max}, which we claim to be better than the others according to the results of the preceding tests, with the exact solutions found by the Cplex software. For this, we have randomly generated some instances of reduced sizes $n \in \{5, 10\}$. Indeed, the processing times p_{i1} and p_{i2} are generated

by a uniform law in $[1, 10]$. To measure the efficiency of the heuristic H_3, we calculate the relative distance between the solution given by the heuristic H_3 and the optimal solution as follows: $R(H) = \frac{Cmax(H)-Opt}{opt}$. Some results of this experimentation are given in the Table 4 with the CPU time in seconds (Time) of the optimal solution. The average CPU time for the heuristic H_3 is smaller than 0.002 seconds. Table 4 clearly shows the efficiency of the heuristic H_3.

In general, results obtained for the different tests reveal that the heuristics based on the LPT rule generally give very good solutions that are optimal in most cases.

7 Conclusion

We studied the flow-shop problem with two machines connected by a conveyor. The performance criteria chosen is the total execution time (makespan). We introduced a transport system of jobs: robot or vehicle. We presented and modeled our problem as a linear program in integer and binary variables. We also proposed lower bounds that are going to serve like reference to appraise the quality of solutions obtained by the developed methods. Some subproblems of the general problem are analyzed and solved in polynomial time. Having shown that the general problem is NP-Hard, we developed the heuristics. Tests have been carried on several instances randomly generated in order to study the performance of the different proposed heuristics.

Research in this field remains open. The introduction of conveyors to the flow-shop problem brings us to conceive other models that are related to the characteristic of storage spaces and the number of conveyors. We may also consider to develop the meta-heuristic or exact methods.

References

1. ZL. Chen and CY. Lee. Machine scheduling with transportation considerations. Journal of scheduling, 4 :3-24, (2001).
2. N. Chikhi. Two machine Flow-shop with transportation time. Thesis of magister. Faculty of Mathematics, USTHB University,Algiers, (2008).
3. J. Hurink and S. Kunst. Flow-shop problems with transportation times and a single robot. Universit Osnabrack, (1998).
4. S. S. Panwalker. Scheduling of a two machine flow shop with travel time between machines. J.Opl.Res.Soc, 42, No 7: 609-613, (1991).
5. H. Kise. On an automated two-machine flowshop scheduling problem with infinite buffer. Journal of the Operations Research Society of Japan, 34 : 354-361, (1991).
6. M. Pinedo. Scheduling: theory, Algorithms, and Systems. Prentice-Hall: Englewoods Cliffs, NJ, (1995).
7. JW. Stevens and DD. Gemmill. Scheduling a two-machine flowshop with travel times to minimize maximum lateness. International Journal of Production Research; 35:1-15,(1997).
8. W. Yu. The two-machine flow shop problem with delays and the one-machine total tardiness problem. Ph.D. Dissertation, Eindhoven University of Technology, (1996).

Extremal trees for new lower bounds on the k-independence number.

Nacera Meddah.
LAMDA-RO, Dep.Mathematics, University of Blida,
B.P. 270, Blida, Algeria. e-mail: meddahnacera@yahoo.fr.

Abstract

Let k be a positive integer and $G = (V, E)$ a graph. A subset S of V is a k-independent set of G if the maximum degree of the subgraph induced by the vertices of S is less or equal to $k-1$. The maximum cardinality of a k-independent set of G is the k-independence number $\beta_k(G)$. We give lower bounds on $\beta_k(G)$ in terms of the order, the chromatic number and the number of supports vertices. Moreover we characterize extremal trees attaining these bounds.

Keywords: Domination, independence, k-independence.

1 Introduction

Let G be a simple graph with vertex set V and edge set E. The number of vertices of G is called the *order,* and is denoted by $n = n(G)$. The *open neighborhood* $N(v) = N_G(v)$ of a vertex v consists of all vertices adjacent to v and $d(v) = d_G(v) = |N(v)|$ is the *degree* of v. The *closed neighborhood* of a vertex v is defined by $N[v] = N_G[v] = N_G(v) \cup \{v\}$. By $\delta = \delta(G)$ and $\Delta = \Delta(G)$, we denote the minimum and the maximum degree of the graph G, respectively. A vertex of *degree* one is called a *leaf* and its neighbor is called a *support vertex*. If v is a *support vertex* then L_v will denote the set of the leaves adjacent to v. We denote by $S(G)$ and $L(G)$ the set of *support* vertices and the set of leaves, respectively, and we let $s(G) = |S(G)|$ and $\ell(G) = |L(G)|$. For a subset $A \subseteq V(G)$, we denote by $\langle A \rangle$ the *subgraph induced* by the vertices of A. We denote by $K_{p,q}$ the *complete bipartite* graph with partite sets X and Y such that $|X| = p$ and $|Y| = q$. We denote by $S_{p,q}$ the double star, obtained by attaching p leaves at an endvertex of a path P_2 and q leaves at the second one.

Let k be a positive integer. A subset S of V is k-independent G if the maximum degree of the subgraph induced by the vertices of S is less or equal to $k-1$. A k-independent set of G is maximal if for every vertex $v \in V \backslash S$, $S \cup \{v\}$ is not k-independent. Clearly every set of a k vertices is a *k-independent* set, and so $\beta_k(G) \geq k$. Also if $k > \Delta$, then the entire vertex set $V(G)$ is k-independent, and so $\beta_k(G) = n$. Therefore in the whole of the paper, we will assume that k is an integer with $1 \leq k \leq \Delta$. The *k-independence number* $\beta_k(G)$ is the maximum cardinality of a k-independent set of G. Notice that 1-independent sets are the classical independent sets, and so $\beta_1(G) = \beta(G)$. If S is a k-independent set of G of size $\beta_k(G)$, then we call S a $\beta_k(G)$-set.

k-independence was introduced by Fink and Jacobson [6, 7] and is studied for examlple in [3, 4, 5, 8, 9] and elsewhere.

A *p-coloring* of a graph G is a function defined on V into a set of colors $\{1, 2, ..., p\}$ such that any two adjacent vertices have different colors. Each set of vertices colored with one color is an *independent* set of vertices of G, so a *p-coloring* is a partition of V into p *independent* sets. The minimum cardinality p for which G has *p-coloring* is the *chromatic number* $\chi(G)$ of G. The parameter $\chi(G)$ has been extensively studied by many authors. One of the classical results concerning the *chromatic number* of a graph is due to Brooks [2].

Theorem 1 (brooks [2]) *For any graph G, $\chi(G) \leq \Delta + 1$, with equality if and only if either $\Delta \neq 2$ and G has a subgraph $K_{\Delta+1}$ as a connected component or $\Delta = 2$ and G has a cycle C_{2k+1} as a connected component.*

2 Lower bounds

We begin by giving the following two results that can be found in [1].

Lemma 2 (Blidia et al.[1]) *For $k \geq 1$, let w be a vertex of a graph G'' such that every neighbor of w has degree at most k, at least w or one of its neighbors has degree k or more, and every vertex in $V(G'')\backslash N[w]$, if any, has degree less than k in G''. Let G' be any graph and G the graph constructed from G' and G'' by adding an edge between w and any vertex of G'. Then $\beta_k(G) = \beta_k(G') + |V(G'')| - 1$.*

Theorem 3 (Blidia et al.[1]) *Let G be a connected bipartite graph of order $n \geq 2$ with $s(G)$ support vertices. Then $\beta_2(G) \geq \dfrac{n + s(G)}{2}$.*

Next we provide a generalization of Theorem 3. Let $\delta_s(G) = \underset{v \in S(G)}{Min} |L_v|$.

Theorem 4 *Let G be a graph of order n with a chromatic number $\chi(G)$. Then*

a) *If* $\delta_s(G) \geq k-1$, *then* $\beta_k(G) \geq \dfrac{n + (\chi(G)-1)(k-1)s(G)}{\chi(G)}$.

b) *If* $\delta_s(G) \leq k-2$, *then* $\beta_k(G) \geq \dfrac{n + i + (\delta_s(G)\chi(G) - (k-1))s(G)}{\chi(G)}$

with $i = \sum_{v \in S(G)} (k - 1 - Min(|L_v|, k-1)) \geq 1$.

Proof. The result can be easily checked if G is a complete graph. Thus assume that G is not complete and let C be a set of leaves defined as follows: for each support vertex v of G we put in C exactly $Min(|L_v|, k-1)$ of its leaves. Clearly $|C| \leq (k-1)s(G)$. Let $A_1, A_2, ..., A_{\chi(G)}$ be a $\chi(G)$-coloration of the subgraph induced by the vertices of $V(G) - C$. Without loss of generality, we can assume that $|A_1| \leq |A_2| \leq ... \leq \left|A_{\chi(G)}\right|$. Note that $\chi(G) = \chi\langle V(G) - C\rangle$. We consider the following two cases.

Case **a.** $\delta_s(G) \geq k-1$. Then $|C| = (k-1)s(G)$ and therefore

$$n - (k-1)s(G) = |A_1| + |A_2| + ... + \left|A_{\chi(G)}\right| \leq \chi(G)\left|A_{\chi(G)}\right|,$$

implying that $\left|A_{\chi(G)}\right| \geq \dfrac{n-(k-1)s(G)}{\chi(G)}$. Since $A_{\chi(G)} \cup C$ is k-independent, $\beta_k(G) \geq \left|A_{\chi(G)} \cup C\right| \geq \dfrac{n-(k-1)s(G)}{\chi(G)} + (k-1)s(G)$. It follows that $\beta_k(G) \geq \dfrac{n + (\chi(G)-1)(k-1)s(G)}{\chi(G)}$.

Case **b.** $\delta_s(G) \leq k-2$. Then $\delta_s(G)s(G) \leq |C| < (k-1)s(G)$ and therefore $|C| = (k-1)s(G) - i$, where $i = \sum_{v \in S(G)} (k - 1 - Min(|L_v|, k-1)) \geq 1$. Hence $n - ((k-1)s(G) - i) = |A_1| + |A_2| ... + \left|A_{\chi(G)}\right| \leq \chi(G)\left|A_{\chi(G)}\right|$, implying that $\left|A_{\chi(G)}\right| \geq \dfrac{n + i - (k-1)s(G)}{\chi(G)}$. Since $A_{\chi(G)} \cup C$ is k-independent, $\beta_k(G) \geq \left|A_{\chi(G)} \cup C\right| \geq \dfrac{n + i - (k-1)s(G)}{\chi(G)} + \delta_s(G)s(G)$. It follows that $\beta_k(G) \geq \dfrac{n + i + (\chi(G)\delta_s(G) - (k-1))s(G)}{\chi(G)}$ with $i \geq 1$. This completes the proof of Theorem 4. ■

as immediate consequences to Theorem 1 and 4, we obtain the following corollaries.

Corollary 5 *Let G be a graph of order n and maximum degree $\Delta(G)$. Then*

a) *If* $\delta_s(G) \geq k-1$, *then* $\beta_k(G) \geq \dfrac{n+\Delta(G)(k-1)s(G)}{\Delta(G)+1}$.

b) *If* $\delta_s(G) \leq k-2$ *and* $i = \sum_{v \in S(G)} (k-1-Min(|L_v|, k-1)) \geq 1$, *then*

$$\beta_k(G) \geq \frac{n+i+(\delta_s(G)(\Delta(G)+1)-(k-1))s(G)}{\Delta(G)+1}.$$

Observe that if $G = C_{2m+1}$, then $\beta_k(G) > \dfrac{n+\Delta(G)(k-1)s(G)}{\Delta(G)+1}$. Thus if connected with $\beta_k(G) = \dfrac{n+\Delta(G)(k-1)s(G)}{\Delta(G)+1}$, then $\chi(G) = \Delta(G)+1$ and by Theorem 1, $G = K_n$.

On the other hand, $\chi(G) = 2$ for all bipartite graphs G having at least one edge. Using this fact we have:

Corollary 6 *Let G be a bipartite graph of order n. Then*

a) *If* $\delta_s(G) \geq k-1$, *then* $\beta_k(G) \geq \dfrac{n+(k-1)s(G)}{2}$ (1).

b) *If* $\delta_s(G) \leq k-2$ *and* $i = \sum_{v \in S(G)} (k-1-Min(|L_v|, k-1)) \geq 1$, *then*

$$\beta_k(G) \geq \frac{n+i+(2\delta_s(G)-(k-1))s(G)}{2} \quad (2).$$

To see that the bound of Corollary 6 -(a) is sharp, we consider the graph G_q ($q \geq 1$) obtained from a path P_q and q cycles C_4 by identifying one vertex of each cycle with a vertex of the path. Then $n = 4q$, $s(G_q) = 0$, $\delta_s(G_q) = 0$, $k = 1$ and $\beta_1 = 2q = \dfrac{n+(k-1)s(G)}{2} = \dfrac{4q+0}{2}$.

For the particular case $k = 2$, we have:

Corollary 7 *Let G be a graph with chromatic number $\chi(G)$. Then*

$$\beta_2(G) \geq \frac{n+(\chi(G)-1)s(G)}{\chi(G)}.$$

3 Trees with equality in (1)

For the purpose of characterizing trees that attain the bound in Corollary 6-(a), we define the family $\mathcal{G}$ of all non trivial trees T that can be obtained from a sequence $T_0, T_1, ..., T_i$ $(i \geq 1)$ of trees, where $T_0 = K_{1,k}$ $(k \geq 1)$, $T_1 = S_{k-1,k-1}$:$(k \geq 2)$, $T = T_i$, and if $i \geq 2$, T_{i+1} can be obtained recursively from T_i by one of the following operations.

- *Operation* $\mathcal{G}_1$: Add a copy of a star $K_{1,k}$ attached by an edge between any vertex of the star $K_{1,k}$ and a vertex r of T_i, with the condition that if r is a leaf of T_i, then its support vertex z satisfy $|L_z| \geq k-1$.
- *Operation* $\mathcal{G}_2$: Add a copy of a double star $S_{k-1,k-1}$ of supports vertices u, v attached by an edge uz at a vertex z of T_i, with the condition that if z is a leaf of T_i, then its support vertex z' in T_i satisfy $|L_{z'}| \geq k-1$.

Observe that if T is a tree of $\mathcal{G}$, then $\delta_s(T) \geq k-1$. We let $s(P_2) = 2$.

Lemma 8 *If $T = P_2$ or $T \in \mathcal{G}$. Then $\beta_k(T) = \dfrac{n + (k-1)s(T)}{2}$.*

Proof. Clearly if $T = P_2$, then $\beta(T) = n/2 = 1$ and $\beta_2(T) = \dfrac{n + s(T)}{2} = 2$. Now let T be any tree of $\mathcal{G}$. We proceed by induction on the number of operations $\mathcal{G}_i$ performed to construct T. The property is true for $T_0 = K_{1,k}$ and $T_1 = S_{k-1,k-1}$. Suppose the property true for all trees of $\mathcal{G}$ constructed with $j-1 \geq 0$ operations and let T be a tree of $\mathcal{G}$ constructed with j operations. Consider the following two cases depending on whether if T is obtained by performing operation $\mathcal{G}_1$ or $\mathcal{G}_2$.

If the last operation performed on a tree T' obtained by $j-1$ operations is $\mathcal{G}_1$, then $n(T) = n(T') + k + 1$ and $s(T) = s(T') + 1$. By Lemma 2 and the inductive hypothesis applied on T',

$$\begin{aligned}\beta_k(T) = \beta_k(T') + k &= \frac{(n(T') + (k-1)s(T'))}{2} + k \\ &= \frac{(n(T) - k - 1 + (k-1)(s(T) - 1))}{2} + k = \frac{(n(T) + (k-1)s(T))}{2}.\end{aligned}$$

So, $\beta_k(T) = \dfrac{(n(T) + (k-1)s(T))}{2}$.

If the last operation performed on a tree T' obtained by $j-1$ operations, is $\mathcal{G}_2$, then $n(T) = n(T') + 2k$ and $s(T) = s(T') + 2$. By Lemma 2 and the inductive hypothesis applied on T',

$$\beta_k(T) = \beta_k(T') + 2k - 1 = \frac{(n(T') + (k-1)s(T')}{2} + 2k - 1$$
$$= \frac{(n(T) - 2k + (k-1)(s(T) - 2))}{2} + 2k - 1 = \frac{(n(T) + (k-1)s(T))}{2}.$$

So, $\beta_k(T) = \dfrac{(n(T) + (k-1)s(T))}{2}$. ■

We now are ready to give extremal trees achieving equality in (1).

Theorem 9 *Let T be a non-trivial tree with $\delta_s(T) \geq k-1$. Then $\beta_k(T) = \dfrac{(n(T) + (k-1)s(T))}{2}$ if and only if $T = P_2$ or $T \in \mathcal{G}$.*

Proof. The sufficient condition follows from Lemma 8.

Conversely, let T be a tree with $\beta_k(T) = \dfrac{(n(T) + (k-1)s(T))}{2}$. If $n = 2$, then $T = P_2$. Suppose that $n \geq 3$. We proceed by induction on the order n of T. If $diam(T) = 2$, then $T = K_{1,p}$ with $p \geq k-1$. If $p = k-1$, then $\beta_k(T) = p+1 = \dfrac{p+1+k-1}{2}$, which implies that $p = k-2$, impossible. And if $p \geq k$, then $\beta_k(T) = p = \dfrac{p+1+k-1}{2}$, which implies that $p = k$ and so $T = K_{1,k}$ establishing the base case T_0 and so $T \in \mathcal{G}$. If $diam(T) = 3$, then $T = S_{p,q}$ with $p \geq q \geq k-1$. If $q \geq k$, then $\beta_k(T) = p + q = \dfrac{p+q+2+(k-1)2}{2}$, which implies that $p + q = 2k \leq 2q$, so $p \leq q$. Since $p \geq q$ it results that $p = q = k$ and $T = S_{k,k}$. Thus T is obtained from T_0 by performing $\mathcal{G}_1$ and $T \in \mathcal{G}$. If $q = k-1$, then $\beta_k(T) = p + q + 1 = \dfrac{p+q+2+(k-1)2}{2}$, which implies that $p + q = 2(k-1) = 2q$, which holds $p = q = k-1$ and $T = S_{k-1,k-1}$, establishing the base case T_1 and so $T \in \mathcal{G}$. Now assume that $diam(T) \geq 4$ and root T at a vertex r of maximum eccentricity. Let v be a support vertex at maximum distance from r, and u its parent. We distinguish between two cases:

Case 1. $|L_v| \geq k$.

Let $T' = T - T_v$. Then $n' = n - |L_v| - 1 \geq 3$ and $s(T) \geq s(T') \geq s(T) - 1$. Moreover, $s(T') = s(T)$ if and only if u is the unique leaf of a support vertex of T'. By Lemma 2, $\beta_k(T) = \beta_k(T') + |L_v|$, and by Corollary 6, we have:

$$\frac{n(T) + (k-1)s(T)}{2} = \beta_k(T) = \beta_k(T') + |L_v| \geq \frac{n(T') + (k-1)s(T')}{2} + |L_v|$$

$$\text{So } \beta_k(T) \geq \frac{(n(T) - |L_\upsilon| - 1 + (k-1)(s(T)-1))}{2} + |L_\upsilon|$$
$$= \frac{(n(T) + (k-1)s(T) + |L_\upsilon| - k}{2}$$
$$\geq \frac{(n(T) + (k-1)s(T)}{2} = \beta_k(T).$$

The equality between the extremal two members implies that $\beta_k(T') = \frac{n(T') + (k-1)s(T')}{2}$, $|L_\upsilon| = k$ and $s(T') = s(T) - 1$. Thus u is either a leaf of a strong support vertex in T' with $\delta_s(T') \geq k-1$, or different from a leaf in T'. Now by induction on T', $T' \in \mathcal{G}$, and so $T \in \mathcal{G}$ because it is obtained from T' by performing $\mathcal{G}_1$.

Case 2. $|L_\upsilon| = k-1$. Let $T' = T - T_u$.

From the above case, we may assume that every descendent of u has degree at most k, then $n(T') \geq 3$. Assume that u is adjacent to $q \geq k-1$ or $q = 0$ leaves and has $p \geq 1$ children as support vertices. By Lemma 2 :

$$\frac{n(T) + (k-1)s(T)}{2} = \beta_k(T) = \beta_k(T') + pk + q.$$

Since $\delta_s(T') \geq \ \delta_s(T) \geq k-1$, then $\beta_k(T') \geq \frac{n(T') + (k-1)s(T')}{2}$ and thus

$$\beta_k(T) = \beta_k(T') + pk + q \geq \frac{n(T') + (k-1)s(T')}{2} + pk + q.$$

We have $n(T') = n(T) - pk - q - 1$ and $p \geq 1$. By the different situations related to the value of q and the position of the parent w of u in T', one can check that $s(T) - p - 1 \leq s(T') \leq s(T) - p + 1$. Then we can write $s(T') \geq s(T) - p - i$ with $i = 1$ if $q \geq k-1$, $i = 0$ if $q = 0$, Thus $s(T') = s(T) - p - i$ if and only if w either is not a leaf of T' or w is a leaf of a strong support vertex of T'. Therefore

$$\beta_k(T) = \beta_k(T') + pk + q \geq \frac{(n(T') + (k-1)s(T'))}{2} + pk + q$$
$$= \frac{n(T) - pk - q - 1 + (k-1)(s(T) - p - i) + 2pk + 2q}{2},$$

which implies that $\beta_k(T) \geq \beta_k(T) + \frac{q - i(k-1) + p - 1}{2}$. If $q = 0$, then $i = 0$, so, $\beta_k(T) \geq \beta_k(T) + \frac{p-1}{2} \geq \beta_k(T)$ and if $q \geq k-1$, then $i = 1$, so,

$$\beta_k(T) \geq \beta_k(T) + \frac{q - i(k-1) + p - 1}{2} \geq \beta_k(T).$$

The equality between the extremal two members implies that $\beta_k(T') = \frac{n(T') + (k-1)s(T')}{2}$, and thus $T' \in \mathcal{G}$ by the inductive hypothesis, $q - i(k-1) + p - 1 = 0$ and $s(T') = s(T) - p - i$. It follows from $q - i(k-1) + p - 1 = 0$ that $p = 1$ and $q = i(k-1)$, that is either $p = 1$ and $q = (k-1)$ or $p = 1$ and $q = 0$.

In both cases, T can be obtained from T' by performing operation $\mathcal{G}_1$ if $p = 1$ and $q = 0$, or operation $\mathcal{G}_2$ if $p = 1$ and $q = k - 1$. Therefore $T \in \mathcal{G}$ which completes the proof. ■

In order to characterize trees T that attain the bound in Corollary 6-(b), we give the following proposition.

Proposition 10 *Let G be a bipartite graph with $\delta_s(G) \leq (k-2)$ such that $\beta_k(G) = \frac{n + i + (2\delta_s(G) - (k-1))s(G)}{2}$ with $i = \sum_{v \in S(G)} (k - 1 - Min(|L_v|, k-1)) \geq 1$. Then $\beta_k(G) = \beta_{k-1}(G) = ... = \beta_{k-j}(G) = \frac{n + (k-1-j)s(G)}{2}$ with $1 \leq j \leq k - 1 - \delta_s(G)$ and $d(v) \geq k$ for every vertex $v \in V(G) - L(G)$.*

Proof. Similarly to the proof of Theorem 4; let C be a set of leaves defined as follows: for each support vertex v of G we put in C exactly $Min(|L_v|, k-1)$ of its leaves. Since $\delta_s(G) \leq k-2$, $\delta_s(G)s(G) \leq |C| < (k-1)s(G)$. Therefore $|C| = (k-1)s(G) - i$ with $i = \sum_{v \in S(G)} (k - 1 - Min(|L_v|, k-1)) \geq 1$. Let A_1, A_2 be the 2-coloration of the subgraph induced by the vertices of $V(G) - C$. Without loss of generality, we have $\frac{n + i - (k-1)s(G)}{2} \leq |A_1| \leq |A_2| \leq \frac{n - \delta_s(G)s(G)}{2}$. Since $A_2 \cup C$ is a k-independent set,

$$\beta_k(G) \geq |A_2 \cup C| = |A_2| + |C| \geq \frac{n + i - (k-1)s(G)}{2} + \delta_s(G)s(G).$$

It follows that $\beta_k(G) \geq \frac{n + i + (2\delta_s(G) - (k-1))s(G)}{2}$. If

$$\beta_k(G) = \frac{n + i + (2\delta_s(G) - (k-1))s(G)}{2},$$

then $|A_2| = \frac{n + i - (k-1)s(G)}{2}$ and $|C| = \delta_s(G)s(G)$. Since

$$\frac{n + i - (k-1)s(G)}{2} \leq |A_1| \leq |A_2|,$$

$|A_2| = \frac{n+i-(k-1)s(G)}{2} = |A_1|$. On the other hand there exists a stable S in the subgraph induced by $V(G) - C$ such that $|S| \geq \frac{|V(G)-C|}{2} = \frac{n-\delta_s(G)s(G)}{2}$. Since $S \cup C$ is a k-independent set, $\beta_k(G) \geq |S \cup C| = |S|+|C| \geq \frac{n-\delta_s(G)s(G)}{2}+\delta_s(G)s(G)$, implying that $\frac{n+i-(k-1)s(G)}{2} = |A_1| = |A_2| = \frac{n-\delta_s(G)s(G)}{2}$. However

$$i-(k-1)s(G) = \sum_{v \in S(G)} (k-1-Min(|L_v|, k-1))-(k-1)s(G) = -\delta_s(G)s(G),$$

it follows that $Min(|L_v|, k-1) = \delta_s(G) \leq k-2$. So $|L_v| = \delta_s(G)\ \forall v \in S(G)$, and therefore $C = L(G)$. Since $\delta_s s(G) = (k-1)s(G) - i$ and $\delta_s(G) \leq k-2$, $i = js(G)$ with $j \in N$, and so $k = \delta_s + 1 + j$. Hence $\beta_k(G) = \frac{n+(k-1-j)s(G)}{2}$ with $1 \leq j \leq k-1-\delta_s(G)$. If $j = 1$, then from Case a of Corollary 6, we have $\beta_k(G) \geq \beta_{k-1}(G) \geq \frac{n+(k-2)s(G)}{2}$. Clearly if $\beta_k(G) = \frac{n+(k-2)s(G)}{2}$, then $\beta_k(G) = \beta_{k-1}(G) = \frac{n+(k-2)s(G)}{2}$. So, we have $\beta_k(G) \geq \beta_{k-1}(G) \geq ... \geq \beta_{k-j}(G) \geq \frac{n+(k-1-j)s(G)}{2}$with $1 \leq j \leq k-1-\delta_s$. Equality between the extremal two members implies that $\beta_k(G) = \beta_{k-1}(G) = ... = \beta_{k-j}(G) = \frac{n+(k-1-j)s(G)}{2}$ and $d(v) \geq k$ for every vertex $v \in V(G) - L(G)$. ■

4 Trees with equality in (2)

In view of Proposition 10, all trees satisfy $\beta_k(T) > \frac{n+i+(2\delta_s(T)-(k-1))s(T)}{2}$, where $i = \sum_{v \in S(T)} (k-1-Min(|L_v|, k-1)) \geq 1$. We will now prove that there is no extremal tree of the bound in Corollary 6-(b).

Theorem 11 *Let T be a non-trivial tree with $\delta_s(T) \leq k-2$. Then there is no tree with $\beta_k(T) = \frac{n+i+(2\delta_s(T)-(k-1))s(T)}{2}$ where*

$$i = \sum_{v \in S(T)} (k-1-Min(|L_v|, k-1)) \geq 1.$$

Proof. From proof of Proposition 10, and since in every tree T, there exists in $\langle V(T) - C \rangle$ a pendent vertex x with degree equal to $(k-1-j)+1$ in T, so $d_T(x) = k - j$ with $j \geq 1$. However the family of extremal trees with $\beta_k(T) = \dfrac{n + i + (2\delta_s(T) - (k-1))s(T)}{2}$ where $i = \sum_{v \in S(T)} (k-1-Min(|L_v|, k-1)) \geq 1$ is empty. ■

5 Conclusion

We have studied the parameter β_k by giving some lower bounds in graphs in section 2. Also we have characterized trees achieving this bounds. For the next research, we propose to characterize extremal bipartite graphs achieving the bounds in Corollary 6-(a, b).

References

[1] M. Blidia, M. Chellali, O. Favaron and N. Meddah. "On k-independence in graphs with emphasis on trees", Discrete Math. 307 (2007) 2209−2216.

[2] R. L. Brooks, On coloring the nodes of a network. Proc. Cambridge Philos. Soc. 37 194-197.

[3] Y. Caro and Z. Tuza, Improved lower bounds on k-independence, J. Graph Theory 15 (1991) $99 - 107$.

[4] O. Favaron, k-domination and k-independence in graphs, Ars Combin. 25 (1988) C $159 - 167$.

[5] O. Favaron, On a conjecture of Fink and Jacobson concerning k-domination and k-dependene, J. Combin. Theory Series B 39 n°1 (1985) $101 - 102$.

[6] J. F. Fink and M. S. Jacobson, n-domination in graphs, in : *Graph Theory with Applications to Algorithms and Computer.* John Wiley and sons, New York (1985) $283 - 300$.

[7] J. F. Fink and M. S. Jacobson, n-domination, n-dependence and forbidden subgraphs, in : *Graph Theory with Applications to Algorithms and Computer.* John Wiley and sons, New York (1985) $301 - 311$.

[8] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, Fundamentals of Domination in Graphs, Marcel Dekker, New York, 1998.

[9] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater, Domination in Graphs: Advanced Topics, Marcel Dekker, New York, 1998.

Benders Decomposition Approach to Set Covering Problems Satisfying Almost the Consecutive Ones Property*

Salim Haddadi and Nacira Hamidane
University of the 8th of May, 1945
Department of Computer Science
Guelma, Algeria
salim.haddadi@yahoo.com

March 22, 2010

Abstract

Although in the actual phase, we are evaluating it, the algorithm presented in this paper is designed for solving real world set covering instances arising from railway public transportation in Germany (Deutsche Bahn). These instances, with large sizes, are characterized by a special pattern. The binary matrices they exhibit, have "almost" consecutive ones property. Taking advantage of this nice structure, we present a decomposition of the SCP into a mixed linear-integer program, and we present a Benders-like algorithm. We end the paper with the computational experience where twenty randomly generated instances are run.
Key Words Set Covering, Benders Decomposition, Consecutive Ones Property, Consecutive Block Minimisation..

1 Introduction

Let A be a binary $m \times n$-matrix and, without loss of generality, let $c_j > 0, j = 1, \cdots, n$, be a cost associated with column j. The set covering problem (SCP) is to cover the rows of A by a subset of the columns at minimum cost. The mathematical model is:

$$\max \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad i = 1, \cdots, m \tag{1}$$

$$x_j \in \{0, 1\} \qquad j = 1, \cdots, n, \tag{2}$$

where the n variables are defined by

*This paper is part of a doctoral dissertation prepared by the second author.

$$x_j = \begin{cases} 1 & \text{if column } j \text{ is in the cover} \\ 0 & \text{otherwise.} \end{cases}$$

The SCP is well known to be hard from theoretical, as well as practical, point of view. It has applications, which are surveyed in [3], in a huge number of domains such as crew scheduling, location of emergency facilities, assembly line balancing, information retrieval, political districting, simplification of boolean expressions, vehicle routing, steel production, traffic assignment in satellite communication systems, and so on, see also the recent papers [1,4 and 9]. Most of the significative algorithms and heuristics are surveyed in [2]. Since then, we noticed that no exact algorithm has been published, and we have found two recent evolutionary heuristics [7,8].

Our goal in this paper, is to present an exact algorithm capable of solving the real world set covering instances arising from railway public transportation in Germany (Deutsche Bahn). These instances, though having a large size, are characterized by a special pattern. The binary matrices they exhibit, have, as was observed in [8], "almost" consecutive ones property (COP). In section 2, we enlighten this fact, and show how can we decrease the number of blocks of consecutive ones. We present, in section 3, the decomposition of the SCP into a mixed linear-integer program, and we present the Benders algorithm. In section 4, we end the paper with the computational experience.

2 Binary matrices having almost the COP

Consider a binary $m \times n$-matrix A. Without loss of generality we may assume that A has no zero rows, nor does it have a zero column. A block of consecutive ones in row i (bco for short) of the binary matrix A is a maximal sequence of ones occurring consecutively. Formally, it is any sequence of entries $a_{ip}, a_{i,p+1}, \cdots, a_{iq}$ in row i satisfying the following:

(i) $a_{ij} = 1, p \leq j \leq q$

(ii) either $p = 1$ or $a_{i,p-1} = 0$

(iii) either $q = n$ or $a_{i,q+1} = 0$.

The matrix A is said to have the consecutive ones property (COP for short) if there exists a permutation of the columns of A so that the ones occur consecutively in every row (in other words, the matrix A has m bco's). Using PQ-trees we can easily (in time linear in the density of A) recognize binary matrices having the COP. Furthermore, the recognition algorithm provides the permutation which leaves the ones appear consecutively in every row. Therefore, without loss of generality, we may assume that a binary matrix having the COP has the property that the ones already occur consecutively in every row.

Most of the binary matrices do not have the COP, and we are interested in the problem of finding a permutation of the n columns of A which minimizes the number of bco's, a well known problem called 'Consecutive Block Minimization (CBM)' which is NP-hard. In [6] we polynomially transformed CBM to the traveling salesman problem satisfying the triangle inequality. Since this

polynomial transformation preserves the ratios of approximation, it constitutes a 1.5-approximation for CBM.

The concept of binary matrices having almost the COP appears for the fist time in [8] in the context of SCP's arising from railway transportation problems. Binary matrices obtained in this way exhibit a special shape. Most of the rows have exactly one bco, and each of the remaining rows has a few number of bco's, far less than n. Binary matrices of this kind are said to have almost the COP. Not surprisingly, the SCP with instances having almost the COP is NP-hard (to see this, think of the vertex cover which is a SCP on matrices having two 1's per row, i.e. every row has either one or two bco's).

3 Benders Decomposition

Without loss of generality, suppose that each of the first p rows of A has one bco, and each of the remaining $m-p$ rows has more than one bco ($m-p \ll m$). In every one of the last $m-q$ rows where there are more than one bco, we define a hole to be any maximal sequence of consecutive 0's between two bco's (a hole must exist since there are more than one bco). Our idea is to fill every hole by inserting 1's, so that the resulting matrix has the COP, and attach to each of the block of ones inserted a new binary coupling variable.

Suppose there are h_i holes in row i of the matrix $A, i = p+1, \cdots, m$. Counting the number of bco's, we find (since there are no zero rows)

$$p + \left(m - p + \sum_{i=p+1}^{m} h_i \right) = m + \sum_{i=p+1}^{m} h_i.$$

The total number of holes in A is $h = \sum_{i=p+1}^{m} h_i$. We shall identify every hole by a subset of the set of the columns $J = \{1, \cdots, n\}$. So, let the k^{th} hole in row i be $H_i^k \subset J, (k = 1, \cdots, h_i), (i = p+1, \cdots, m)$. We transform the SCP into the following mixed binary integer program (call it P):

$$\begin{aligned}
\min \sum_{j=1}^{n} c_j x_j & & & \\
\sum_{j=1}^{n} a_{ij} x_j & \geq 1 & & i = 1, \cdots, p \\
\sum_{j=1}^{n} a_{ij} x_j + \sum_{k=1}^{h_i} \sum_{j \in H_i^k} x_j - \sum_{k=1}^{h_i} y_i^k & \geq 1 & & i = p+1, \cdots, m \\
\sum_{j \in H_i^k} x_j - y_i^k & \leq 0 & & k = 1, \cdots, h_i, i = p+1, \cdots, m \\
x_j & \in \{0,1\} & & j = 1, \cdots, n \\
y_i^k & \in \mathbb{Z}_+ & & k = 1, \cdots, h_i, i = p+1, \cdots, m
\end{aligned}$$

There are as many binary variables y_i^k as holes in the binary matrix of the SCP. In condensed form, problem P reads

$$
\begin{aligned}
\min\, & cx & & \\
A_0 x & \geq & e_1 & \\
A_1 x - B_1 y & \geq & e_2 & (3)\\
A_2 x - I_h y & \leq & 0 & (4)\\
x & \in & \{0,1\}^n & \\
y & \in & \mathbb{Z}^h_+ &
\end{aligned}
$$

where e_1, e_2 are respectively the p(resp. $m-p$)-column vector of 1's, and I_h is the $h \times h$ identity matrix. The sizes of the matrices A_0, A_1, A_2, B_1 are respectively $p \times n, (m-p) \times n, h \times n, (m-p) \times h$. Constraints (3) and (4) together replace the original $m-p$ last constraints of the SCP. To see this subtract (4) from (3). From this observation, it is easy to see that problems SCP and P are equivalent. Since the block

$$
\begin{pmatrix} A_0 \\ A_1 \\ A_2 \end{pmatrix}
$$

has the COP, it is totally unimodular and thus we can rewrite problem P, which is a mixed integer linear program as follows

$$
\begin{aligned}
\min\, & cx & & \\
A_0 x & \geq & e_1 & \\
A_1 x - B_1 y & \geq & e_2 & \\
A_2 x - I_h y & \leq & 0 & \\
x & \geq & 0 & \\
y & \in & \mathbb{Z}^h_+. &
\end{aligned}
$$

Before going any further, let us present an example. Consider the following instance of the SCP

$$
\begin{array}{lccccccccc}
\min & 5x_1 & +3x_2 & +2x_3 & +x_4 & +x_5 & +2x_6 & +3x_7 & & \\
 & & x_2 & +x_3 & +x_4 & +x_5 & & & \geq 1 & \\
 & x_1 & +x_2 & +x_3 & +x_4 & & & & \geq 1 & \\
 & & & x_3 & +x_4 & +x_5 & +x_6 & +x_7 & \geq 1 & \\
 & & x_2 & +x_3 & +x_4 & +x_5 & +x_6 & & \geq 1 & \\
 & x_1 & & +x_3 & +x_4 & & & +x_7 & \geq 1 & \\
 & & x_2 & +x_3 & & & +x_6 & & \geq 1 & \\
 & x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7 & \in \{0,1\}. &
\end{array}
$$

Here $m = 6, n = 7$. The binary constraints matrix has $p = 4$ first rows containing one bco each. The remaining $m - p = 2$ rows have $h = 3$ holes, two in the fifth row and one in the sixth, and we have $H^1_5 = \{2\}, H^2_5 = \{5,6\}, H^1_6 = \{4,5\}$. The proposed decomposition (filled holes are boldfaced) results in the problem

P shown below

$$
\begin{array}{lccccccccccc}
\min & 5x_1 & +3x_2 & +2x_3 & +x_4 & +x_5 & +2x_6 & +3x_7 & & & & \\
 & & x_2 & +x_3 & +x_4 & +x_5 & & & & & & \geq 1 \\
 & x_1 & +x_2 & +x_3 & +x_4 & & & & & & & \geq 1 \\
 & & & x_3 & +x_4 & +x_5 & +x_6 & +x_7 & & & & \geq 1 \\
 & & x_2 & +x_3 & +x_4 & +x_5 & +x_6 & & & & & \geq 1 \\
 & x_1 & +\mathbf{x}_2 & +x_3 & +x_4 & +\mathbf{x}_5 & +\mathbf{x}_6 & +x_7 & -y_5^1 & -y_5^2 & & \geq 1 \\
 & & x_2 & +x_3 & +\mathbf{x}_4 & +\mathbf{x}_5 & +x_6 & & & & -y_6^1 & \geq 1 \\
 & & x_2 & & & & & & -y_5^1 & & & \leq 0 \\
 & & & & & x_5 & +x_6 & & & -y_5^2 & & \leq 0 \\
 & & & & x_4 & +x_5 & & & & & -y_6^1 & \leq 0 \\
 & x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7 & & & & \geq 0 \\
 & & & & & & & & y_5^1, & y_5^2, & y_6^1 & \in \mathbb{Z}_+
\end{array}
$$

where

$$
A_0 = \begin{pmatrix}
 & 1 & 1 & 1 & 1 & & \\
1 & 1 & 1 & 1 & & & \\
 & & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 &
\end{pmatrix}
$$

$$
A_1 = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 & 1 & 1 & 1 & 1 & 1 &
\end{pmatrix}
\quad
B_1 = \begin{pmatrix}
1 & 1 & \\
 & & 1
\end{pmatrix}
$$

$$
A_2 = \begin{pmatrix}
 & 1 & & & & & \\
 & & & & 1 & 1 & \\
 & & & 1 & 1 & &
\end{pmatrix}
\quad
I_3 = \begin{pmatrix}
1 & & \\
 & 1 & \\
 & & 1
\end{pmatrix}
$$

For the convenience of presenting the algorithm, we shall rewrite problem P as follows

$$
\begin{array}{rcl}
\min cx & & \\
Ax - By & \geq & e \\
A_2x - I_hy & \leq & 0 \\
x & \geq & 0 \\
y & \in & \mathbb{Z}_+^h.
\end{array}
$$

where

$$
A = \begin{pmatrix} A_0 \\ A_1 \end{pmatrix} \text{ and } B = \begin{pmatrix} \mathbf{0} \\ B_1 \end{pmatrix}
$$

where $\mathbf{0}$ is the $p \times h$-matrix whose elements are 0's. Benders decomposition paradigm is well known, so we merely present the algorithm steps.

Observe that problem (X) has always a feasible solution $u_1 = 0, u_2 = 0$ (since $c > 0$). Therefore, either does it have an optimal solution, or is it unbounded. On the other hand, assuming no zero rows in the binary constraints matrix, the SCP has an optimal cover. Consequently, problem (Y) must have an optimal solution at any iteration, and the only criteria to finish is to attain the same objective function value for the two problems.

4 Computational experience

We coded our algorithm in C on HP Compaq Core Duo 1.6GHz and run it on a set of twenty randomly generated instances. The numbers m, n, p were fixed

Input: positive integers m, n, matrix constraint of the SCP and cost vector c
Output: optimal cover $\bar{x}$ and its cost $\bar{c}$

- try to reduce the size of the set covering problem by using logical tests;
- try to decrease the number of holes by approximating the underlying CBM problem;
- Use the greedy heuristic to find an approximate cover $\widetilde{x}$;
- $\bar{y}^{(0)} \leftarrow A_2\widetilde{x}$ and $s \leftarrow 0$;
- repeat

  solve problem

$$(X)\begin{cases} \max\left(e + B\bar{y}^{(s)}\right)u_1 - A_2u_2 \\ A^T u_1 - A_2^T u_2 \leq c \\ u_1, u_2 \geq 0 \end{cases}$$

  if (X) has an optimal solution $\left(\bar{u}_1^{(s+1)}, \bar{u}_2^{(s+1)}\right)$ with an objective function value $\bar{c}$ then add to the master problem

$$(Y)\begin{cases} \min z \\ (\bar{u}_1^{(k)}B - \bar{u}_1^{(k)}A_2)y - z \leq -e\bar{u}_1^{(k)} k = 1, \cdots, s \\ z \geq 0 \\ y \in \mathbb{Z}_+^h \end{cases}$$

  the cut $(\bar{u}_1^{(s+1)}B - \bar{u}_2^{(s+1)}A_2)y - z \leq -e\bar{u}_1^{(s+1)}$;

  if (X) is unbounded, let $\left(\bar{u}_1^{(s+1)}, \bar{u}_2^{(s+1)}\right)$ be the actual extreme point and $\left(\bar{u}_1^{(s+2)}, \bar{u}_2^{(s+2)}\right)$ be the extreme ray, then add to (Y) the two cuts $(\bar{u}_1^{(s+1)}B - \bar{u}_2^{(s+1)}A_2)y - z \leq -e\bar{u}_1^{(s+1)}$and $(\bar{u}_1^{(s+2)}B - \bar{u}_2^{(s+2)}A_2)y \leq -e\bar{u}_1^{(s+2)}$;

  Solve problem (Y) and let $\bar{y}^{(s+1)}$be an optimal solution with objective function value $\bar{z}$;

  $s \leftarrow s + 1$;
- until $\left(\bar{c} = \bar{z}\right)$:

Dens.	Number of holes before heuristic	Number of holes after heuristic	Number of cuts	Time
5%	41	26	13	51.23
	35	25	10	42.36
	41	29	28	80.03
	34	28	10	35.23
	38	27	14	55.76
Mean values			15.0	52.9
10%	36	28	10	52.30
	41	32	17	71.26
	50	33	23	87.33
	68	44	22	75.16
	40	30	17	70.33
Mean values			17.8	71.2
15%	69	44	17	84.66
	49	38	18	82.30
	52	36	22	90.63
	62	45	28	96.56
	80	55	21	90.33
Mean values			21.2	88.9
20%	128	84	19	151.83
	87	69	24	122.26
	115	80	32	163.26
	96	63	26	118.80
	101	65	29	158.70
Mean values			26	143.0

Table 1: Computational results on twenty randomly generated instances

once for all to 50, 200, 45 respectively. First, the binary matrices generated have the COP, then $\partial\%$ of the ones in the five last rows are replaced by 0's ($\partial = 5, 10, 15, 20$). The binary matrices are preprocessed in order to decrease the crucial number of holes, by transforming the underlying CBM problem to the traveling salesman problem, and applying a simple greedy heuristic to find an approximate tour. During the execution of our algorithm, the subprograms (related to vector x) are computed using a rudimentary simplex algorithm (method of tableaus) and the master programs are resolved with a rudimentary implicit enumeration method. This explains why executing times are rather large, the instances being of small or medium size. What is interesting is the small number of cuts needed to confirm the optimality.

Our next task is to test this algorithm (after fine-tuning the preprocessing phase using logical tests to reduce the size of the problem, and using Cplex instead of the rudimentary methods used above) on the real-world instances obtained from Ruf and Schöbel (see [8]).

References

[1] Blazsik, Z. and B. Imreh (1996) A Note on Connection Between Process Network Synthesis and Set Covering Problems, Acta Cybernetica 12, 309-312.

[2] Caprara, A., M. Fischetti and P. Toth (2000) Algorithms for the Set Covering Problem, Annals of Operations Research 98, 353-371.

[3] Ceria, S., P. Nobili and A. Sassano (1997) Set Covering Problem. In M. Dell'Amico, F. Mafioli and P. Toth (eds.) Annotated Bibliographies in Combinatorial Optimization, John Wiley and Sons, New York, 415-428.

[4] Chen, L. and J. Crampton (2009) Set Covering Problems in Role-Based Access Control, In Proceedings of the 14th European Symposium on Research in Computer Security (ESORICS09), 689-704.

[5] Gouwanda, D. and S.G. Ponnambalam (2008) Evolutionnary Search Techniques to Solve Set Covering Problems, World Academy of Science, Engineering and Technology 39, 20-25.

[6] Haddadi, S. (2008) Consecutive Block Minimization is 1.5-approximable, Information Processing Letters 108, 132-135.

[7] Lan, G., G.W. DePuy and G.E. Whitehouse (2007) An Effective and Simple Heuristic for the Set Covering Problem, European Journal of Operational Research 176, 1387-1403.

[8] Ruf, N. and A. Schöbel (2004) Set Covering with almost Consecutive Ones Property, Discrete Optimization 15, 215-228.

[9] Yang, J. and J.Y-T. Leung (2003) A Generalization of the Weighted Set Covering Problem, Wiley Periodicals (Draft).

Programmation par contraintes et ses applications

Control With Constraints of a Class of Hybrid Systems Based on Adaptive Method of Linear Programming

Nait Abdesselam Aldjia[1], Aidene Mohamed[2] and Djennoune Said[3]

Université Mouloud Mammeri ,
Faculté de Génie Electrique et d'Informatique,
Laboratoire de Conception et Conduites de
Systèmes de Production (L2CSP),
Tizi Ouzou, Algerie.

Abstract. This paper presents an extended version of adaptive method of linear programming to be used for constructing optimal open loop controls of hybrid dynamic systems. We particularly consider a class of hybrid systems described by a finite set of linear subsystems and a commutation law. The active subsystem and the commutations between the subsystems can be defined by the autonomous transitions (autonomous model switchings). In general, such problems are solved in two steps to find both optimal continuous inputs and optimal switching times.The results are illustrated by an example.

1 Introduction

Traditionally, most of research work in process control has been concerned with the control of continuous dynamic processes described by ordinary differential equations [1] , or discrete time systems described by finite automaton [2]. The increasing role of the control of physical processes and the need to design effective control systems that can explicitly take into account the continuous and discrete dynamics, are the reasons for the increased interest in hybrid systems[3]. Hybrid system is a dynamical system whose evolution depends on a coupling between variables that take values in a continuum and variables that take values in a finite or countable set. The development of specific methods of representation, analysis and control is necessary to take into account the complexity of these systems.

In recent years, there has been an increasing interest in the study of autonomous-switching systems because of its significance in both academic research and practical applications [4, 11]. This systems are an important class of hybrid dynamical systems which consist of a family of subsystems and a switching law specifying the active subsystem at each time instant. Examples of autonomous-switching systems can be found in chemical processes, air traffic management, telecommunication and computer networks, electrical circuit systems, etc.

Recently, optimal control problems of switched systems have been attracting researchers from various fields in science and engineering, since this system type represents a powerful tool for approximating non linear systems. Various efforts have been made to extend the classical optimal control methods to hybrid systems [5, 6, 9, 14]. hybrid versions of the maximum principle have been presented in [6], more complicated versions of maximum principle are proved by [12] and by [16], Capuzzo Dolcetta and Kratz [10, 16] study systems with switchings using the dynamic programming approach to drive the Hamilton-Jacobi-Bellman (HJB) equations and prove the existence and uniqueness of viscosity solutions. Branicky in [5] formulates optimal control problems for hybrid systems modeled by his unified model approach; he also proposes some theoretically algorithmic approaches related to some inequalities of the value functions.

The main purpose of this paper is to extend the principle of adaptive method for linear programming to solve optimal control problem of autonomous-switching systems. This method originated from an approach to the solution of linear programming problems given in [8, 7] which is based on the concept of the support matrix for the problem. The paper is organized as follows. In section 2, the optimization problem for a switched system is formulated in the class of discrete controls. In section 3, an algorithmic resolution of the hybrid optimal control problem is suggested. In section 4 we calculate optimal time instants of transition. As an illustration, an example considered in section 5, demonstrate that the algorithm is efficient in constructing optimal open loop controls and can therefore be implemented.

1.1 Autonomous switching Systems

The autonomous systems are characterized by a finite number of linear dynamical models together with a set of rules for switching among these models. Here the vector field changes discontinuously when the state $x(t)$ hits certain boundaries. An example of autonomous switching systems is the following:

Consider a thermostat that is used to control the temperature of a room. The thermostat consists of a heater and a thermometer. Its lower and upper thresholds are set at θ_m and θ_M. Such that $\theta_m \prec \theta_M$. The heater is maintained **on** as long as the room temperature is below θ_M, and it is turned **off** whenever the thermometer detects that the temperature reaches θ_M. Similarly, the heater remains **off** if the temperature is above θ_m and is switched **on** whenever the temperature falls to θ_m. The evolution of the temperature is described as follows: If the heater is **off** the temperature dynamics is given by

$$\dot{T}(t) = -T + 15,$$

and if it is **on** the temperature dynamics is given by

$$\dot{T}(t) = -T + 25.$$

The hybrid system describing the heating of the room can be modeled as the graph shown in figure 1. The two vertices of the graph represent the two discrete modes '**on**' and '**off**'. We associate with the edges the conditions for switching from one mode to another.

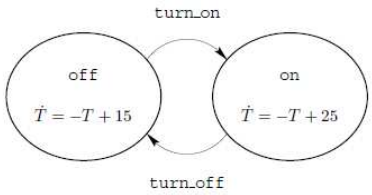


Fig. 1. The model for the thermostat

The trajectory of the temperature alternates between two phases corresponding to the two operation modes of the thermostat (see figure 2).

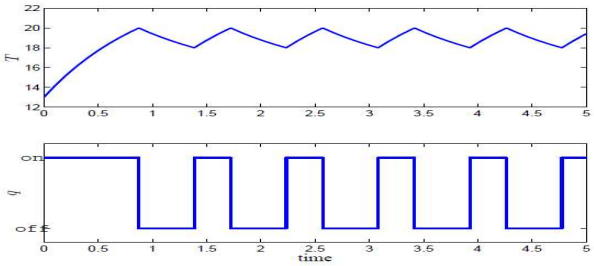


Fig. 2. The trajectory of the thermostat

2 Problem formulation

Given a fixed time interval $T = [t_0, t_f]$ and a sequence of switching times $\tau = \{\tau_1, \tau_2, ..., \tau_r\}$ at which the trajectory $x(t), \ \ t \in [t_0, t_f]$ hits certain boundaries. We note that $\tau_0 = t_0$, $\tau_{r+1} = t_f$ and $\tau_0 \prec \tau_1 \prec ... \prec \tau_{r+1}$.
For all $t \in [\tau_{i-1}, \tau_i], \ \ i = \overline{1, r+1}$ and for every $q \in Q = \{q_1, q_2, ..., q_{r+1}\}$, The dynamical system takes the form:

$$\dot{x}(t) = A_{q_i} x(t) + B_{q_i} u(t), \tag{1}$$

Where $x \in \mathbf{R^n}$ is the continuous state, $q \in Q$ is the discrete state (it is sometimes called the mode), $Q = \{q_1, q_2, ..., q_{r+1}\}$ is the finite set of the value of the discrete

state, $r+1$ is the number of the discrete state, $u(t),\ \ t \in [t_0, t_f]$ is the control input, A_q and B_q the $n \times n$ and $n \times 1$ constant matrices for mode Q.
Let $h = (t_f - t_0)/N$, where N is a positive integer. $T_h = [t_0, t_0+h, t_0+2h, ..., t_f - h]$. $T_{h_i} = [\tau_{i-1}, \tau_{i-1}+h, \tau_{i-1}+2h, ..., \tau_i - h], i = \overline{1, r+1}$ and $T_{h_1} \cup T_{h_2} \cup ... \cup T_{h_{r+1}} = T_h$.
The optimal control problem is to maximize the cost function:

$$L(u, \tau) = c' x(t_f) \to \max, \tag{2}$$

Subject to:

$$\begin{cases} Hx(\tau_i) = g_i, \\ d_* \preceq u(t) \preceq d^*, \end{cases} \tag{3}$$

While bringing the system from an initial state x_0 at time t_0, to a final state x_f at time t_f where the end time is fixed.
Here, $g_i,\ \ i = \overline{1, r+1}$ is a $m-$vector, H is a $m \times n-$matrix, $d_*,\ d^*$ the scalars, c an $n-$cost vector. $u(t),\ \ t \in T$ is said to be a discrete control with the quantization period h, if $u(t) = u(t_0 + kh)$, $t \in [t_0 + kh, t_0 + (k+1)h],\ \ k = \overline{0, N-1}$.
In This paper, we consider a class of hybrid system that it has no discontinuities of the state x at the switching instants. Then, we have:

$$x(\tau_i^+) = x(\tau_i^-) = x(\tau_i),\ \ i = \overline{1, r},$$

The notation τ_i^-, τ_i^+ is used for the left (resp. right) hand limit of x at τ_i

Definition 1. For autonomous switched system, the control input of the system consists of both a control input $u(t),\ \ t \in [t_0 + kh, t_0 + (k+1)h],\ \ k = \overline{0, N-1}$ and a switching instants $\tau = \{\tau_1, \tau_2, ..., \tau_r\}$.

Definition 2. The discrete control $u(t),\ \ t \in T$ and the vector τ are called the feasible control for problem (1-3) if they satisfy constraints (2-3).

Definition 3. The admissible control $(u^0(t),\ \tau^0)$ and the corresponding trajectory $x^0(t),\ \ t \in T$ are said to be optimal open loop control and trajectory if the control criterion reaches its maximal value:

$$c' x^0(t_f) = \max_{(u,\tau)} c' x(t_f).$$

Definition 4. For given $\epsilon \succeq 0$, an $\epsilon-$optimal control $(u^\epsilon(t), \tau^\epsilon)\ \ t \in T$ are defined with inequality:

$$c' x^0(t_f) - c' x^\epsilon(t_f) \preceq \epsilon.$$

The purpose of this study is to realize the adaptive method of linear programming for constructing the optimal open loop control of a class of hybrid system. In general, we need to find an optimal or $\epsilon-$optimal control solution $(u^0(t),\ \tau^0)$ (resp. $(u^\epsilon(t), \tau^\epsilon)$) for problem (1-3). This problem is solved in two steps.

1 The first step consists in fixing the vector τ corresponding to a feasible trajectory. Then, problem (1-3) reduces to an optimal control input $u(t),\ \ t \in T$ that maximizes $L_\tau(u) = L(u, \tau)$, the problem is solved by applying the adaptive method presented in [7].

2 In second step, the switching instants are corrected by choosing optimal instants of transition from one mode to another.

Step 01: In order to use the concepts and to adapt the methods of linear programming, we reduce the problem (1-3) to a linear programming problem.
Let $\psi_c(t),\ \ t \in T$, be a solution to the adjoint equation:

$$\dot{\psi_c}(t) = -A'_{q_i}\psi_c(t),\ \ i = 1, r+1,$$

with the initial condition $\psi_c(t_f) = c$.
$G(t),\ \ t \in T$, be an $m \times n$ matrix function satisfying the equation:

$$\dot{G}_i = -G_i(t)A_{q_i},$$

with the initial condition $G(t_f) = H$.
We assume that:

$$p_{q_i}(t) = \int_t^{t+h} \psi'_c(\upsilon)B_{q_i}d\upsilon,$$

and

$$\varphi_{q_i}(t) = \int_t^{t+h} G_i(\upsilon)B_{q_i}d\upsilon.$$

Thus, we obtain a linear optimal control problem:

$$L(u) = \sum_{i=1}^{r+1}\sum_{t\in T_{h_i}} p_{q_i}(t)u(t) \to \max, \tag{4}$$

$$\sum_{t\in T_{h_i}} \varphi_{q_i}(t)u(t) = \bar{g}_i,\ \ q \in Q = \{q_1, q_2, ..., q_{r+1}\},\ \ i = \overline{1, r+1}, \tag{5}$$

$$d_* \preceq u(t) \preceq d^*,\ \ t \in T_h, \tag{6}$$

where $\bar{g}_i = g_i - Hx_0(\tau_i),\ x_0(\tau_i),\ \ t \in T_{h_i}$, is the trajectory of system (1) with $u(t) = 0,\ \ t \in [\tau_{i-1}, \tau_i],\ \ i = \overline{1, r+1}$.

2.1 Support control and the accompanying elements

We choose from T_h an arbitrary subset $T_{sup} = \{t_l,\ \ l = \overline{1,m}\}$ and from $\varphi_{q_i}(t)$ an $m \times m-$matrix $\varphi_{sup} = \{\varphi_{q_i}(t),\ \ t \in T_{sup},\ \ q_i \in Q = \{q_1, q_2, ..., q_{r+1}\}\}$. A set T_{sup} is said to be a support of problem (4-6) if $det(\varphi_{sup}) \neq 0$.
A pair $\{u(t), T_{sup}\}$ made up of an admissible control and a support is called a support control.

Define the support accompanying elements:
On the base of the support T_{sup} we find the Lagrange $m-$vector y as a solution to the equation $y'\varphi_{sup} = p'_{sup}$, where $p_{sup} = \{p_{q_i},\ \ t \in T_{sup}\}$.

With the knowledge of the Lagrange vector y, we construct a co-control which is an analogue of an estimate vector: $\Delta_{q_i}(t) = p_{q_i}(t) - y'\varphi_{q_i}(t),\ \ t \in [\tau_{i-1}, \tau_i]$,

Using a solution of the adjoint equation, it is not difficult to show that

$$\Delta_{q_i}(t) = \int_t^{t+h} \psi'(\upsilon) B_{q_i} d\upsilon,\ \ t \in T_{h_i}, \tag{7}$$

where $\psi(t),\ \ t \in T$, is a solution to the adjoint equation with the initial condition $\psi(t_f) = c - H'y$. (Transversally condition).

To construct a pseudo-control $w(t),\ \ t \in T$, based on T_{sup}, we first define the $w(t),\ \ t \in T_n$, $T_n = T \backslash T_{sup}$:

$$\begin{cases} w(t) = -1, & \text{if } \Delta_{q_i}(t) \prec 0, \\ w(t) = 1, & \text{if } \Delta_{q_i}(t) \succ 0, \\ w(t) \in [-1, 1], & \text{if } \Delta_{q_i}(t) = 0. \end{cases} \tag{8}$$

and $w(t),\ \ t \in T_{sup}$ is constructed with the use of the equation (5):

$$\sum_{t \in T_{sup}} \varphi_{q_i}(t) w(t) + \sum_{t \in T_n} \varphi_{q_i}(t) w(t) = \bar{g}_i, \tag{9}$$

If $d_* \preceq w(t) \preceq d^*,\ \ t \in T_{sup}$, then, $u^0(t) = w(t), t \in T_h$ is an optimal control.

A solution $æ(t),\ \ t \in T_h$ to equation (1) with the discrete control $u(t) = w(t),\ \ t \in T_h$ and the initial condition $x(t_0) = x_0$ will be called a pseudo-trajectory.

A suboptimality estimate of the support control $u(t), T_{sup}$, can be defined by:

$$\beta(u(t), T_{sup}) = c' æ(t_f) - c' x(t_f). \tag{10}$$

3 Method of calculation of the optimal control with a fixed τ

The adaptive method is based on an iteration in which a current support control is replaced by a new one:

$$\{u(t), T_{sup}\} \to \{\bar{u}(t), \overline{T}_{sup}\},$$

so that $\beta(\bar{u}(t), \overline{T}_{sup}) \preceq \beta(u(t), T_{sup})$.

Suppose that for a given $\epsilon \succeq 0$ at an initial support control $\{u(t), T_{sup}\}$, the suboptimality estimate $\beta(u(t), T_{sup}) \succ \epsilon$ and inequalities $d_* \preceq w(t) \preceq d^*, \ \ t \in T_{sup}$ do not hold. An iteration consists of two procedures:

1. Change of an admissible control $u(t) \to \bar{u}(t)$.
2. Change of a support $T_{sup} \to \overline{T}_{sup}$.

3.1 Change of an admissible control

A new feasible control is constructed according to the formula:

$$\bar{u}(t) = u(t + \partial u(t) = u(t) + \theta^0 l(t), \ \ t \in T_h), \tag{11}$$

where the direction $l(t)$ is defined by:

$$l(t) = w(t) - u(t).$$

A step θ^0 is computed as:

$$\theta^0 = \min\{1, \theta(t)\}, \ \ t \in T_{sup},$$

where

$$\theta(t) = \begin{cases} (-1 - u(t))/l(t), & \text{if } l(t) \prec 0, \\ (1 - u(t))/l(t), & \text{if } l(t) \succ 0, \\ +\infty, & \text{if } l(t) = 0, \ \ t \in T_{sup}. \end{cases}$$

The new admissible control $\bar{u}(t)$ satisfies the relation:

$$\beta(\bar{u}(t), T_{sup}) = (1 - \theta^0)\beta(u(t), T_{sup}).$$

If $\beta(\bar{u}(t), T_{sup}) \preceq \epsilon$ then $\bar{u}(t), \ \ t \in T_h$, is an $\epsilon-$optimal control of problem (4-6). Otherwise we go on to the change of support.

3.2 Change of a support

In this procedure, the support of problem (4-6) is transformed into the optimal support T^0_{sup}.

The transformation of the current support T_{sup} to the new support $\overline{T}_{sup}$ is done as follows, first, an instant $t^0 \in T_{sup}$ corresponding to θ^0 is eliminated from the support T_{sup}. In order to determine the instant of time or the index to be added to the support, we calculate a step σ^0 along the variation ∂y.

A construction of the new support starts with the calculation of the variation (direction of changing) ∂y of the Lagrange vector y. ∂y is obtained from the equation:

$$-\varphi'_{sup}\partial y = \partial\delta(t), \quad t \in T_{sup}.$$

where $\partial\delta(t^0) = sign(\bar{u}(t^0))$, $\partial\delta(t) = 0, \quad t \in T_{sup}\backslash t^0$.

Define

$$\partial\delta_{q_i}(t) = -\partial y\varphi_{q_i} = -\partial y \int_t^{t+h} G_i(\upsilon)B_{q_i}d\upsilon.$$

A new estimate vector is given by:

$$\overline{\Delta}_{q_i}(t) = \Delta_{q_i}(t,\sigma) = \Delta_{q_i}(t) + \sigma^*\partial\delta_{q_i}(t), \quad t \in T_{h_i}, \quad \sigma \succeq 0. \tag{12}$$

Let $T_{n_0} = \{t \in T_n, \quad if \quad \Delta_{q_i}(t) = 0\}$ be a subset of nonsupport zeroes. For every point $t \in T_{n_0}$ we calculate a value $\sigma(\tilde{t})$ for which a new zero of function (12) arises at one of the nodes:

1. If $\Delta_{q_i}(t)\partial\delta_{q_i}(t) \prec 0$, then $\tilde{t} = t, \quad t \in T_{n_0}$.
2. If $\Delta_{q_i}(t)\partial\delta_{q_i}(t) \succ 0$, then $\tilde{t} = t - h, \quad t \in T_{n_0}$.

Calculate:

$$\sigma(\tilde{t}) = -\Delta_{q_i}(\tilde{t})/\partial\delta_{q_i}(\tilde{t}),$$

$$\sigma(t_0) = \begin{cases} -\Delta_{q_i}(t_0)/\partial\delta_{q_i}(t_0), & \text{if } \Delta_{q_i}(t_0).\partial\delta_{q_i}(t_0) \prec 0, \\ +\infty, & \text{if } \Delta_{q_i}(t_0).\partial\delta_{q_i}(t_0) \succ 0. \end{cases}$$

$$\sigma(t_f) = \begin{cases} -\Delta_{q_i}(t_f)/\partial\delta_{q_i}(t_f), & \text{if } \Delta_{q_i}(t_f).\partial\delta_{q_i}(t_f) \prec 0, \\ +\infty, & \text{if } \Delta_{q_i}(t_f).\partial\delta_{q_i}(t_f) \succ 0. \end{cases}$$

Introduce a set $T^0_n = T_{n_0} \cup \{t_0\} \cup \{t_f\}$. From the sequence $\sigma(t), \quad t \in T^0_n$, we choose:

$$\sigma^* = \sigma(t^*) = \min_{t \in T^0_n} \sigma(t).$$

Construct a new support:

$$\overline{T}_{sup} = (T_{sup}\backslash\{t^0\}) \cup \{t^*\}. \tag{13}$$

Thus, the algorithm presented is used to construct an optimal support control $\{u^0(t), T^0_{sup}\}$ for problem (4-6) with a fixed τ.

4 Optimal time instant of transition

The adaptive method is used to construct an optimal support control for problem (4-6), with a fixed switching instants.

The method that can invoked to determine the optimal instant τ is based on the gradient of objective functional of problem (4) with respect to the instants $\tau_1, \tau_2, ..., \tau_r$. Thus, we must calculate the derivative $\partial L(u,\tau)/\partial\tau_s, \;\; s = \overline{1,r}$ of the objective functional with respect to the instants $\tau_1, \tau_2, ..., \tau_r$.

Denote by $(u^0_{sup} = \{u(t), t \in T^0_{sup}\})$ the support values of the optimal control $u^0(t), \;\; t \in T_h$ for the fixed instant τ. Consider a small variation $\Delta\tau_s$ of τ_s that does not change the support T^0_{sup}, the optimal control corresponding to $\tau_s + \Delta\tau_s$, differs from $u^0(t), \;\; t \in T_h$ only in the support components $u^0_{sup} + \Delta u^0_{sup}$.

5 Example

Example 1 (Mass oscillatory system). To illustrate some of the results obtained here, consider the system presented in figure 3 :

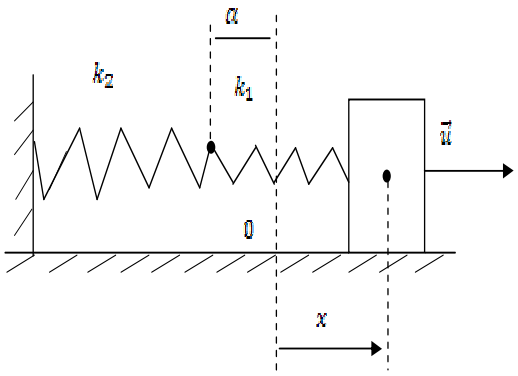


Fig. 3. Mass oscillatory system

The mathematical model of the problem has the form:

$$L(u,\tau) = \dot{x}(t_f) \rightarrow \max,$$

There are two discrete modes:

$$\dot{x}(t) = \begin{cases} x_2(t), \\ -x_1(t) + u, \text{ if } x \succeq \alpha. \end{cases}$$

$$\dot{x}(t) = \begin{cases} x_2(t), \\ -3x_1(t) - 2\alpha + u, \text{ if } x \preceq \alpha. \end{cases}$$

$$x_0 = (1, 0), \quad t_0 = 0, \quad t_f = 6, \quad \alpha = 0.5.$$

To solve the problem, we consider tree initial switching instants $\tau = \{0.77, 3.3, 3.96\}$. As an initial support, a set $T_{sup} = \{1.5\}$. this support corresponds to the set of nonsupport zeroes of the co-control $T_{n_0} = \{3.327, 5.908\}$. the problem was solved in 46 iterations to construct the optimal open loop control. The optimal value of the control criterion corresponding to the fixed instants was equal to 0.8754. The result is shown in figure 4:

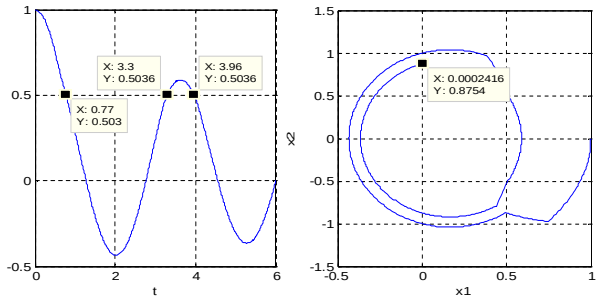


Fig. 4. A trajectory with the fixed instants

The optimal control corresponding to a fixed instants is shown in figure 5:

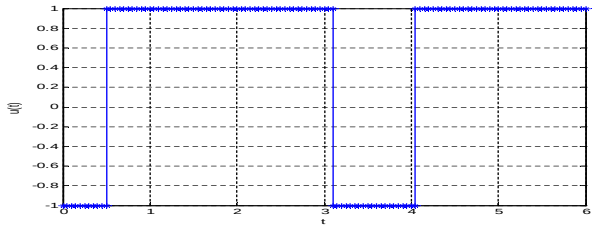


Fig. 5. Control with the fixed instants

The optimal values of transition times are $\tau^* = \{0.76, 3.26, 3.98\}$. The corresponding optimal value of the objective functional is $L(u^*, \tau^*) = 0.9268$. And the corresponding optimal control is illustrated by:

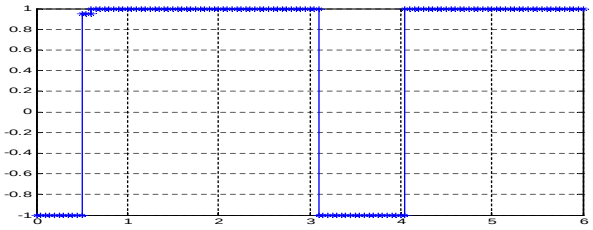


Fig. 6. Optimal control with optimal switching instants

The optimal control corresponding to the optimal switching instants differs from the control obtained with the fixed instants, only in the support component. Thus, $T^0_{sup} = \{0.5\}$ and $u_{sup} = u(T^0_{sup}) = 0.85$.

6 Conclusion

In this paper, we formulated an optimal control problem of autonomous switching systems. A classical adaptive method of linear programming is extended to this class of hybrid systems. Particularly, we proposed a study of a problem where the number of switching instants is given.

This method however guarantees both optimal piecewise controls and optimal switching instants. It can be extended to optimal control problems for other classes of hybrid system.

References

1. Kalman,R.E.: mathematical description of linear dynamical systems. SIAM journal on control. 1963 1,152-192
2. Cassandras,C.: Discrete Event Systems: Modeling and Performance Analysis. Asken Associates Incorporated Publishers. 1993
3. Branicky,M.S.,Mitter, S.K.,BORKAR, V.: A unified framework for hybrid control: Model and optimal control theory . IEEE Trans.on Automatic Control. **43** 1998 31-45
4. Branicky,M.S.:Studies in Hybrid Systems. Ph.D.dissertation, Dept.Elec.Eng.and Computer Sci., Massachusetts Inst.Techaol., Cambridge. (June 1995)
5. Branicky,M.S.,Mitter,S.K.: Algorithms for Optimal Hybrid Control. Prooceeding of the 34rd IEEE Conference on Decision and Control, La nouvelle Orléans. (1995) 2661-2666
6. Sussmann,H.J.: A maximum principle for hybrid optimal control problems. In Proceedings of the 38th IEEE Conference on Decision and Control, Phoenix,AZ,December. 1999

7. Balashevich,N.V.,Gabasov,R.F.,Kirillova,F.M.: Numerical Methods for open-loop and Closed-Loop Optimization of Linear control systems. Zh.Vychisl.Mat.Mat.Fiz. **40** (2000) 838-859
8. Gabasov,R.F.,Kirillova F.M.,Kostyukova O.I: Construction of closed Loop Optimal controls in a linear Problem. Dokl.Akad.Nauk SSSR **320**,no.6 (1991) 1294-1299
9. Cebron,B.,Sechilariu,M.,Burger,J.:Optimal Control of hybrid dynamical systems with hysteresis. Prooceeding of European Control Conference 1999
10. Bardi,M.,Capuzzo-Dolcetta,I.: Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations. Volume 17 Systems and Control: Foundations and Applications. Birkausser. 1997
11. Seidman,T.I.: Optimal control for switching systems. In Proceedings of the 21st Annual Conference on Information Sciences and systems. 1987 485-489
12. Sussmann,H.J.: A maximum principle for hybrid optimal control problems. In Proceedings of the 38th IEEE Conference on Decision and Control, Phoenix,AZ,December. 1999
13. Hedlund,S.,Rantzer,A.:Optimal Control of hybrid Systems. Proceeding of IEEE Conf. on Decision and Control, Phoenix. 1999
14. Johansson,M.: Piecewise Linear Control Systems. Lectures Notes in Control and Information Sciences. Springer. **284** 2003
15. Pontryaguine,L.,Boltiansky,V.,Gamkrelidze, R.,Michtchenko,E.:The mathematical theory of optimal processes. Editions de Moscou. 1962
16. Riedinger,P.,Iung,C.,Kratz,F.: An optimal control approach for hybrid systems. European Journal of Control. **9** 2003 449-459.

This article was processed using the LaTeX macro package with LLNCS style

Local symmetry breaking in the satisfiability problem

Belaïd Benhamou, Tarek Nabhani, Richard Ostrowski, Mohamed Réda Saïdi

Université de Provence
Laboratoire des Sciences de l'Information et des Systèmes (LSIS)
Centre de Mathématiques et d'Informatique
39, rue Joliot Curie - 13453 Marseille cedex 13, France
Email:{benhamou; nabhani; ostrowski; saidi}@cmi.univ-mrs.fr

Abstract. The SAT problem is shown to be the first decision NP-complete problem (Cook,71). It is central in complexity theory. In the last decade, the satisfiability proof procedures are improved by symmetry elimination. A CNF formula usually contains an interesting number of symmetries. There are two kinds of symmetry exploitation. The first one corresponds to global symmetry breaking, that is, only the symmetries of the initial problem (the problem at the root of the search tree) are detected and eliminated. The second one deals with all local symmetries that appear at each node of the search tree. Local symmetry has to be detected and eliminated dynamically during the search. Exploiting such symmetries seems to be a hard task. Almost all of the known works on symmetry in satisfiability are on global symmetry. Only few works are carried on local symmetry, despite their importance in practice. An important challenge is then to detect and break local symmetries efficiently during the search. The work that we present here is a contribution towards an answer to this hard challenge. We present a new method for local symmetries breaking that consists in detecting dynamically local symmetries by reducing the remaining partial SAT instance at each node of the search tree to a graph that has an equivalent automorphism group than the symmetry group of the partial SAT instance. We used the software Saucy to compute the automorphism group and implemented a local symmetry cut in a SAT solver. We experimented this method on several SAT instances and compared it with a method exploiting global symmetries. The results obtained are very promising. Local symmetry improves global symmetry on some hard instances and is complementary to global symmetry.

1 Introduction

Krishnamurthy in [20] introduced the symmetry principle in propositional calculus and showed that some tricky formulas can have short proofs when augmenting the resolution proof system by the symmetry rule. Symmetries are used earlier for the resolution of many problems such as the eight queens [15]. They are also introduced into the resolution of a constraints satisfaction problem [13, 26, 3], in an intelligent algorithm of Backtracking [9] and in the first order logic [11]. Symmetry becomes an important notion in constraint programming. During the last

decade, several works on symmetry breaking for the satisfiability problem and in CSPs appeared. Nevertheless, few search works deal with dynamic symmetry detection and elimination [4–7, 14]. Most of the methods exploiting symmetries deal only with global symmetries [17, 2, 1], that is, the symmetries of the initial problem corresponding to the root of the search tree . A symmetry of a logical formula is a literal permutation leaving invariant the formula. There are many problems in artificial intelligence which contain an important number of symmetries when expressed in CNF formulas. The importance of symmetry breaking when solving these problems can be seen on some difficult problems which are badly handled by the classic search methods. Take for instance the Pigeon-holes problem [8, 16], or the Ramsey problem [18]. Both problems are known to be hard for the classic resolution methods, and they are well represented in first order logic by a small set of formulas that becomes very wide when we calculate all the propositional terminal instantiations. The set of propositional clauses obtained, contains an important number of symmetries. That is, the set of clauses remains invariant under several variable permutations. Exploiting such permutations results in a polynomial complexity for the satisfiability proof while both problems are known to be exponential for the methods that do not take into account symmetry breaking. The satisfiability problem is generic, several problems in other field can be reduced to the satisfiability checking. For example, automatic deduction, configuration, planning, scheduling, etc. Several symmetry elimination methods for the satisfiability problem are introduced [17, 2, 1]. But, almost all of them deal only with global symmetry and ignore the treatment of local symmetries. This is due to the difficulty of detecting and exploiting them dynamically, contrary to global symmetries that can be treated by static approaches that are easier to implement. An approach of dynamic detection of local symmetries in propositional logic was proposed in [4–6]. but this method is incomplete, in the sense that it detects only some local symmetries, not the total group of local symmetry. An alternative to this method is to adapt and use the graph automorphism computational tool Saucy [2] that is able to detect all the local symmetries during search, since the group of automorphisms of the graph deduced from the SAT instance is identical to the symmetry group of the SAT instance. In this paper, we present an alternative local symmetry breaking method for the SAT problem that exploits the total group of symmetry. This method consists in reducing incrementally the logical sub-formula defined at each search node to a graph on which we apply graph automorphism detection tools like Saucy [2]. Symmetry elimination is implemented in a SAT solver that we experimented and compared on several SAT instances. The obtained results are very promising and show that local symmetry breaking outperforms global symmetry breaking on some SAT instances and its combination with global symmetry seems to be complementary. The rest of the paper is organized as follows: Section 2 gives some background on the satisfiability problem and permutations. Section 3 defines the symmetry principle and gives some symmetry properties. The fourth section describes the new symmetry detection and elimination method that we propose. Section 5 shows how the symmetry cut is integrated in a tree search

method like Davis and Putnam procedure. We evaluate the proposed method in the sixth section where several SAT instances are tested and where a comparison of our method with some other existing methods is given. Finally, we conclude the work in Section 7.

2 Some background on propositional logic

2.1 Propositional logic

We shall assume that the reader is familiar with the propositional calculus. We give here, a short description, a more complete description can be found in [21]. Let V be the set of propositional variables called only variables. Variables will be distinguished from literals, which are variables with an assigned parity 1 or 0 that means True or False, respectively. This distinction will be ignored whenever it is convenient, but not confusing. For a propositional variable p, there are two literals: p the positive literal and $\neg p$ the negative one.

A clause is a disjunction of literals $\{p_1, p_2, \ldots, p_n\}$ such that no literal appears more than once, nor a literal and its negation at the same time. This clause is denoted by $p_1 \vee p_2 \vee \ldots \vee p_n$. A system $\mathcal{F}$ of clauses is a conjunction of clauses. In other words, we say that $\mathcal{F}$ is in the conjunctive normal form (CNF).

A truth assignment to a system of clauses $\mathcal{F}$ is a mapping I defined from the set of variables of $\mathcal{F}$ into the set {True, False}. If $I[p]$ is the value for the positive literal p then $I[\neg p] = 1 - I[p]$. The value of a clause $p_1 \vee p_2 \vee \ldots \vee p_n$ in I is True, if the value True is assigned to at least one of its literals in I, False otherwise. By convention, we define the value of the empty clause ($n = 0$) to be False. The value $I[\mathcal{F}]$ of the system of clauses is True if the value of each clause of $\mathcal{F}$ is True, False, otherwise. We say that a system of clauses $\mathcal{F}$ is satisfiable if there exists some truth assignments I that assign the value True to $\mathcal{F}$, it is unsatisfiable otherwise. In the first case I is called a model of $\mathcal{F}$. Let us remark that a system which contains the empty clause is unsatisfiable.

It is well-known [27] that for every propositional formula $\mathcal{F}$ there exists a formula $\mathcal{F}'$ in conjunctive normal form(CNF) such that the length of $\mathcal{F}'$ is at most 3 times as long as the formula $\mathcal{F}$ and $\mathcal{F}'$ is satisfiable iff $\mathcal{F}$ is satisfiable. In the following we will assume that the formulas are given in a conjunctive normal form.

2.2 Permutations

Let $\Omega = \{1, 2, \ldots, N\}$ for some integer N, where each integer might represent a propositional variable. A permutation of Ω is a bijective mapping σ from Ω to Ω that is usually represented as a product of cycles of permutations. We denote by $Perm(\Omega)$ the set of all permutations of Ω and $\circ$ the composition of the permutation of $Perm(\Omega)$. The pair $(Perm(\Omega), \circ)$ forms the permutation group of Ω. That is, $\circ$ is closed and associative, the inverse of a permutation is a permutation and the identity permutation is a neutral element. A pair $(T, \circ)$ forms a sub-group of $(S, \circ)$ iff T is a subset of S and forms a group under the operation $\circ$.

The orbit $\omega^{Perm(\Omega)}$ of an element ω of Ω on which the group $Perm(\Omega)$ acts is $\omega^{Perm(\Omega)}$=$\{\omega^{\sigma} : \omega^{\sigma} = \sigma(\omega), \sigma \in Perm(\Omega)\}$.

A generating set of the group $Perm(\Omega)$ is a subset Gen of $Perm(\Omega)$ such that each element of $Perm(\Omega)$ can be written as a composition of elements of Gen. We write $Perm(\Omega)$=$< Gen >$. An element of Gen is called a generator. The orbit of $\omega \in \Omega$ can be computed by using only the set of generators Gen.

3 Symmetry

Since Krishnamurthy's [20] symmetry definition in propositional logic, several other definitions are given in the CP community. Freuder in his work [13], introduced the notions of full and neighborhood interchangeabilities, where two domain values are interchangeable in a CSP, if they can be substituted for each other without any effects to the CSP. In the other hand Benhamou in [3] defined two levels of semantic symmetry and a notion of syntactic symmetry. He also showed that the Full interchangeability of Freuder is a particular case of semantic symmetry and Neighborhood interchangeability is a particular case of syntactic symmetry. More recently a work of Cohen et al [10] discussed most of the known symmetry definitions in CSPs and gathered them in two definitions: symmetry of solutions (semantic) and symmetry of constraints (syntactic). Almost all of these definitions can be identified to belong to the two families of symmetry: syntactic symmetry or semantic symmetry. We will define in the following both semantic and syntactic symmetry in propositional logic and show their relationship with the solution and constraint symmetries in CSPs.

3.1 Symmetry in propositional logic

Definition 1 (Semantic symmetry). *Let $\mathcal{F}$ be a propositional formula given in CNF and $L_{\mathcal{F}}$ its complete* [1] *set of literals. A semantic symmetry of $\mathcal{F}$ is a permutation σ defined on $L_{\mathcal{F}}$ such that $\mathcal{F} \models \sigma(\mathcal{F})$ and $\sigma(\mathcal{F}) \models \mathcal{F}$.*

In other words a semantic symmetry of a formula is a literal permutation that conserves the set of the models of the formula. We recall in the following the definition of syntactic symmetry given in [4, 5]

Definition 2 (Syntactic symmetry). *Let $\mathcal{F}$ be a propositional formula given in CNF and $L_{\mathcal{F}}$ its complete set of literals. A syntactic symmetry of $\mathcal{F}$ is a permutation σ defined on $L_{\mathcal{F}}$ such that the following conditions hold:*

1. $\forall \ell \in L_{\mathcal{F}}, \sigma(\neg\ell) = \neg\sigma(\ell)$,
2. $\sigma(\mathcal{F}) = \mathcal{F}$

In other words, a syntactical symmetry of a formula is a literal permutation that leaves the formula invariant. If we denote by $Perm(L_{\mathcal{F}})$ the group of permutations of $L_{\mathcal{F}}$ and by $Sym(L_{\mathcal{F}}) \subset Perm(L_{\mathcal{F}})$ the subset of permutations of $L_{\mathcal{F}}$ that are the syntactic symmetries of $\mathcal{F}$, then $Sym(L_{\mathcal{F}})$ is trivially a sub-group of $Perm(L_{\mathcal{F}})$.

[1] The set of literals containing each literal of $\mathcal{F}$ and its negation

Remark 1. The symmetry definitions introduced in CSPs [10] are related to the former ones introduced in propositional logic. Consider for instance the direct SAT encoding $\mathcal{F}$ of a CSP P [19] where a boolean variable is introduced for each CSP variable-value pair, and where a clause forbidding each tuple disallowed by a specific constraint is added as well as another clause ensuring that a value is chosen for each variable in its domain. It is then trivial that the solution symmetry of the CSP P is equivalent to the semantic symmetry (Definition 1) of its SAT encoding $\mathcal{F}$ and the constraint symmetry of P is equivalent to the syntactic symmetry (Definition 2) of $\mathcal{F}$.

Theorem 1. *Each syntactical symmetry of a formula $\mathcal{F}$ is a semantic symmetry of $\mathcal{F}$.*

Proof. It is trivial to see that a syntactic symmetry is a sufficient condition to a semantic symmetry. Indeed, if σ is syntactic symmetry of $\mathcal{F}$, then $\sigma(\mathcal{F}) = \mathcal{F}$, thus it results that $\mathcal{F}$ and $\sigma(\mathcal{F})$ have the same set of models. Each syntactic symmetry is a semantic symmetry and the converse is in general not true.

Example 1. Let $\mathcal{F}$ be the following set of clauses: $\mathcal{F}$=$\{a \vee b \vee c, \neg a \vee b, \neg b \vee c, \neg c \vee a, \neg a \vee \neg b \vee \neg c\}$ and σ_1 and σ_2 two permutations defined on the complete set $L_{\mathcal{F}}$ of literals occurring in $\mathcal{F}$ as follows:
σ_1=$(a, b, c)(\neg a, \neg b, \neg c)$
σ_2=$(a, \neg a)(b, \neg b)(c, \neg c)$
Both σ_1 and σ_2 are syntactic symmetries of $\mathcal{F}$, since $\sigma_1(\mathcal{F})$=$\mathcal{F}$=$\sigma_2(\mathcal{F})$.

In the sequel we deal only with syntactic symmetry, we say only symmetry to designate syntactic symmetry.

Definition 3. *Two literals ℓ and ℓ' of a formula $\mathcal{F}$ are symmetrical if there exists a symmetry σ of $\mathcal{F}$ such that $\sigma(\ell) = \ell'$.*

Definition 4. *Let $\mathcal{F}$ be a formula, the orbit of a literal $\ell \in L_{\mathcal{F}}$ on which the group of symmetries $Sym(L_{\mathcal{F}})$ acts is $\ell^{Sym(L_{\mathcal{F}})}$=$\{\sigma(\ell) : \sigma \in Sym(L_{\mathcal{F}})\}$*

Proposition 1. *All the literals in the orbit of a literal ℓ are symmetrical two by two.*

Proof. The proof is a trivial consequence of the previous two definitions

Example 2. In Example 1, the orbit of the literal a is $a^{Sym(L_{\mathcal{F}})}$= $\{a, b, c, \neg a, \neg b, \neg c\}$. We can see that all the literals are in the same orbit. Thus, they are all symmetrical.

If I is a model of $\mathcal{F}$ and σ a symmetry, we can get another model of $\mathcal{F}$ by applying σ on the variables which appear in I. That is, if I is a model of $\mathcal{F}$ then $\sigma(I)$ is a model of $\mathcal{F}$. A symmetry σ transforms each model into a model and each no-good into a no-good. In the following propositions, we assume that σ is a symmetry of the set of clauses $\mathcal{F}$.

Proposition 2. *Let ℓ be a literal, σ a symmetry such that $\ell' = \sigma(\ell)$ and $I' = \sigma(I)$. If I is such that $I[\ell] = True$, then I' is such that $I'[\ell'] = true$*

Proof. The proof is trivial. Indeed, if ℓ is true in the model I then $\sigma(\ell)= \ell'$ will be true in the model $\sigma(I)=I'$.

we deduce the following proposition.

Proposition 3. *If a literal ℓ has the value true in a model of $\mathcal{F}$, then $\sigma(\ell)$ will have the value true in a model of $\mathcal{F}$.*

Theorem 2. *Let ℓ and ℓ' be two literals of $\mathcal{F}$ that are in the same orbit with respect to the symmetry group $Sym(L_{\mathcal{F}})$, then ℓ is true in a model of $\mathcal{F}$ iff ℓ' is true in a model of $\mathcal{F}$.*

Proof. If ℓ is in the same orbit as ℓ' then it is symmetrical with ℓ' in $\mathcal{F}$. Thus, there exists a symmetry σ of $\mathcal{F}$ such that $\sigma(\ell) = \ell'$. If I is a model of $\mathcal{F}$ then $\sigma(I)$ is also a model of $\sigma(\mathcal{F}) = \mathcal{F}$, besides if $I[\ell] = true$ then $\sigma(I[\ell']) = true$ (Proposition 2). For the converse, consider $\ell = \sigma^{-1}(\ell')$, and make a similar proof.

Corollary 1. *Let ℓ be a literal of $\mathcal{F}$, if ℓ is not true in any model of $\mathcal{F}$, then each literal $\ell' \in orbit^{L_{\mathcal{F}}}$ is not true in any model of $\mathcal{F}$.*

Proof. The proof is a direct consequence of Theorem 2

Corollary 1 expresses an important property that we will use to break local symmetry at each node of the search tree. That is, if a failure is detected after assigning the value True to the current literal ℓ, then we compute the orbit of ℓ and assign the value false to each literal in it, since by symmetry the value true will be contradictory, then will not participate in any model of the considered formula.

Many hard problems for resolution have been shown to be polynomial when using symmetry in resolution. For instance, finding some of the Ramsey's numbers or solving the pigeon-hole problem are known to be exponential for classical resolution, while short proofs can be made for both them when adding the symmetry rule to the resolution proof system. We will show now how to detect dynamically the local symmetry.

4 Local symmetry detection and elimination

Local symmetries have to be detected dynamically at each node of the search tree. Dynamic symmetry detection had been studied in [4, 5] where a local syntactic symmetry search method had been given. However, this method is not complete, it detects only one symmetry σ at each node of the search tree when failing in the assignment of the current literal ℓ. A heuristic is used on the variable permutations of σ in order to get a maximal number of literals in the same cycle of permutations as the one where ℓ appears. Despite this heuristic, this method does not detect all the symmetrical literals with ℓ corresponding to the orbit of ℓ, since it does not use all the local symmetries.

As an alternative to this incomplete symmetry search method, we adapted Saucy [2] to detect all the local syntactic symmetries and show how to break such symmetries during search. Saucy is a tool for computing the automorphism group of a graph. Other tools like Nauty [22] or the most recent methods AUTOM [25] or the one described in [23] can be adapted to search local symmetry. It is shown in [25] that AUTOM is the best method. Because the source code of AUTOM is not free, and more recently Saucy had been improved [12]; we chose Saucy. It is shown in [17, 2, 1] that each CNF formula $\mathcal{F}$ can be represented by a graph $G_{\mathcal{F}}$ that is built as follows:

- Each boolean variable is represented by two vertices (literal vertices) in $G_{\mathcal{F}}$: the positive literal and its negation. These two vertices are connected by an edge in the graph $G_{\mathcal{F}}$.
- Each non binary clause is represented by a vertex (a clause vertex). An edge connects this vertex to each vertex representing a literal of the clause.
- Each binary clause is represented by an edge connecting the vertices representing its two literals. We do not need to add vertices for binary clauses.

An important property of the graph $G_{\mathcal{F}}$ is that it preserves the syntactic group of symmetries of $\mathcal{F}$. That is, the syntactic symmetry group of the formula $\mathcal{F}$ is identical to the automorphism group of its graph representation $G_{\mathcal{F}}$, thus we use Saucy on $G_{\mathcal{F}}$ to detect the syntactic symmetry group of $\mathcal{F}$. Saucy returns a set of generators Gen of the symmetry group from which we can deduce each symmetry. Saucy offers the possibility to color the vertices of the graph such that, a vertex is allowed to be permuted with another vertex if they have the same color. This restricts the permutations to the nodes having the same color. Two colors are used in $G_{\mathcal{F}}$, one for the vertices corresponding to the clauses of $\mathcal{F}$ and the other color for the vertices representing the literals of $L_{\mathcal{F}}$. This allows to distinguish the clause vertices from the literal vertices, then prevent the generation of symmetries between clauses and literals. The source code of Saucy can be found at (http://vlsicad.eecs.umich.edu/BK/SAUCY/).

Example 3. Let $\mathcal{F}$ be the CNF formula given in Example 1. Its associated $G_{\mathcal{F}}$ is given in Figure 1

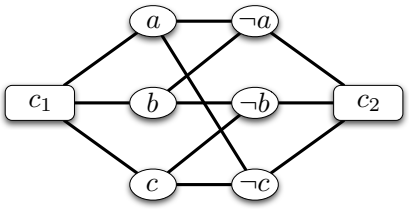


Fig. 1. The graph $G_{\mathcal{F}}$ corresponding to $\mathcal{F}$

Dynamic symmetry detection: Consider a CNF formula $\mathcal{F}$, and a partial assignment I of $\mathcal{F}$ where ℓ is the current literal under assignment. The assignment I simplifies the given formula $\mathcal{F}$ into a sub-formula $\mathcal{F}_I$ that defines a state in the search space corresponding to the current node n_I of the search tree. The main idea is to maintain dynamically the graph $G_{\mathcal{F}_I}$ of the sub-formula $\mathcal{F}_I$ corresponding to the local sub-problem defined at the current node n_I, then color the graph $G_{\mathcal{F}_I}$ and compute its automorphism group $Aut(\mathcal{F}_I)$. The sub-formula $\mathcal{F}_I$ can be viewed as the remaining sub-problem corresponding to the unsolved part. By applying Saucy on this colored graph we can get the generator set Gen of the symmetry sub-group existing between literals of $L_{\mathcal{F}_I}$ from which we can compute the orbit of the current literal ℓ that we will use to make the symmetry cut.

Symmetry elimination: We use Corollary 1 to prune search spaces of tree search methods. Indeed, if the assignment of the value true to the current literal ℓ defined at a given node n_I of the search tree is shown to be a failure, then the assignment of the value true to each literal in the orbit of ℓ will result in a failure too. Thus, the value $false$ has to be assigned to each literal in the orbit of ℓ. Therefore we prune the sub-space which corresponds to the assignment of the alternative value $true$ to these literals in the search tree. That is what we call the symmetry cut.

5 Symmetry advantage in tree search algorithms

Now we will show how these detected symmetrical literals can be used to increase the efficiency of CNF SAT algorithms. We choose in our implementation the Davis Putnam (DP) procedure to be the baseline method that we want to improve by the advantage of local symmetry elimination.

If I is an inconsistent partial interpretation in which the assignment of the value $true$ to the current literal ℓ is shown to be conflicting, then according to Corollary 1, all the literals in the orbit of ℓ computed by using the group $Sym(\mathcal{F}_I)$ returned by Saucy are symmetrical to ℓ. Thus, we assign the value false to each literal in $\ell^{Sym(L_{\mathcal{F}})}$ since the value true is shown to be contradictory, and then we prune the sub-space which corresponds to the value true assignments. The resulting procedure called Satisfiable is given in Figure 2.

The function $orbit(\ell, Gen)$ is elementary, it computes the orbit of the literal ℓ from the set of generators Gen returned by Saucy.

6 Experiments

Now we shall investigate the performances of our search techniques by experimental analysis. We choose for our study some SAT instances to show the local symmetry behavior in satisfiability. We expect that symmetry breaking will be more profitable in real-life applications. Here, we tested and compared four methods:

```
Procedure Satisfiable(F);
begin
  if F = ∅ then F is satisfiable
  else if F contains the empty clause, then F is unsatisfiable
    else begin
      if there exists a mono-literal or a monotone literal ℓ then
        if Satisfiable(F_ℓ) then F is satisfiable
        else F is unsatisfiable
      else begin
        Choose an unsigned literal ℓ of F
        if Satisfiable(F_ℓ) then F is satisfiable
        else
        begin
            Gen=Saucy(F);
            ℓ^{Sym(L_F)}=orbit(ℓ,Gen)={ℓ_1, ℓ_2, ..., ℓ_n};
            if Satisfiable(F_{¬ℓ_1∧¬ℓ_2∧...∧¬ℓ_n}) then F is satisfiable
            else F is unsatisfiable
        end
      end
end
```

Fig. 2. The Davis Putnam procedure with local symmetry elimination

		No-sym		Global-sym		Local-sym		Global-Local-sym	
Instance	*Vars* : *clauses*	*Nodes*	*Time*	*Nodes*	*Times*	*Nodes*	*Time*	*Nodes*	*Time*
fpga10_8_SAT	120 : 448	6,637,776	44.41	449	0.02	9835	2.09	449	0.71
fpga10_9_SAT	135 : 549	-	>1,000	284	0.02	57080	20.37	284	0.53
fpga12_8_SAT	144 : 560	6,637,776	35.79	165	0.00	9835	2.14	165	0.32
fpga13_10_SAT	195 : 905	-	>1,000	4261	0.41	304,830	134.89	4261	14.08
Chnl10_11	220 : 1122	3,628,800	100.09	382	0.09	512	2.42	382	3.33
Chnl10_12_3	240 : 1344	3,628,800	120.72	322	0.10	512	2.63	322	3.41
Chnl11_12_3	264 : 1476	-	>1,000	1123	0.26	1024	6.28	1123	12.09
Chnl11_13	286 : 1742	-	>1,000	814	0.25	1024	7.38	814	10.96
Chnl11_20	440 : 4220	-	>1,000	523	0.38	1024	18.93	523	18.90
Urq3_5	46 : 470	-	>1,000	16384	0.16	30	0.09	15	0.00
Urq4_5	74 : 694	-	>1,000	-	>1,000	44	0.32	31	0.10
Urq5_5	121 : 1210	-	>1,000	-	>1,000	73	1.43	44	0.27
Urq6_5	180 : 1756	-	>1,000	-	>1,000	110	4.76	84	2.03
Urq7_5	240 : 2194	-	>1,000	-	>1,000	147	9.32	108	3.44
Urq8_5	327 : 3252	-	>1,000	-	>1,000	225	27.11	171	9.18

Table 1. Results on some SAT instances

1. **No-sym:** search without symmetry breaking by using the LSAT [24] as the baseline method;
2. **Global-sym** search with global symmetry breaking. This method uses in pre-processing phase the program SHATTER [2, 1] that detects and eliminates the global symmetries of the considered instance by adding on it symmetry breaking clauses, then apply the solver LSAT to the resulting instance. The CPU time of *Global-sym* in Table 1 includes the time that SHATTER spends to compute the global symmetry.

3. **Local-sym:** search with local symmetry breaking. This method implements in LSAT the dynamic local symmetry detection and elimination strategy described in this work. The CPU time of *Local-sym* includes local symmetry search time.
4. **Global-Local-sym:** search that combines both the global and local symmetries. It consists in applying LSAT with local symmetry elimination on the instance produced by SHATTER in the pre-processing phase.

on different SAT instances that are FPGA (Field Programmable Gate Array), Chnl, Urquhart and some random graph coloring instances. The common baseline search method for the three previous methods is LSAT. The complexity indicators are the number of nodes of the search tree and the CPU time. Both the time needed for computing local symmetry and global symmetry are added to the total CPU time of search. The source codes are written in C and compiled on a Pentium 4, 2.8 GHZ and 1 Gb of RAM.

6.1 The results on the different SAT instances

Table 1 shows the first results of the methods on some SAT instances. It gives the instance, the instance size ($variables/clauses$), the number of nodes of the search tree and the CPU time for each method.

Table 1 shows that *Global-sym* is in general better than *Local-sym* and No-sym in both the number of nodes and the CPU time on the $FPGA$, and $Chnl$ problems, but *Local-sym* still able to solve them too. These problems contain a great amount of global symmetries, that is why it is sufficient to break only the global symmetry to solve them efficiently, eliminating local symmetry in these problems may sometimes slow the resolution. The Urq instances are known to be harder than the $FPGA$ and the $Chnl$, we can see that No-sym is not able to solve them and *Global-sym* solved only the $Urq3\_5$ and failed to solve all the other ones under the time limit. *Local-sym* solved all the Urq instances efficiently, local symmetry elimination is then more profitable than global symmetry on the Urq instances. We can see that in average the method *Global-Local-sym* is better than all the other methods, it solved all the instances efficiently. It compares well to *Global-sym* on the $FPGA$ and $Chnl$ instances and to *Local-sym* on the Urq instances. It is then profitable to combine both symmetry eliminations to solve these problems, the results confirmed that both methods could be complementary.

6.2 The results on the graph coloring instances

Random graph coloring problems are generated with respect to the following parameters: (1) n : the number of vertices, (2) $Colors$: the number of colors and (3) d: the density which is a number between 0 and 1 expressed by the ratio : the number of constraints (the number of edges in the graph) to the number of all possible constraints. For each test corresponding to some fixed values of the parameters n, $Colors$ and d, a sample of 100 instances are randomly generated and the measures (CPU time, nodes) are taken on the average.

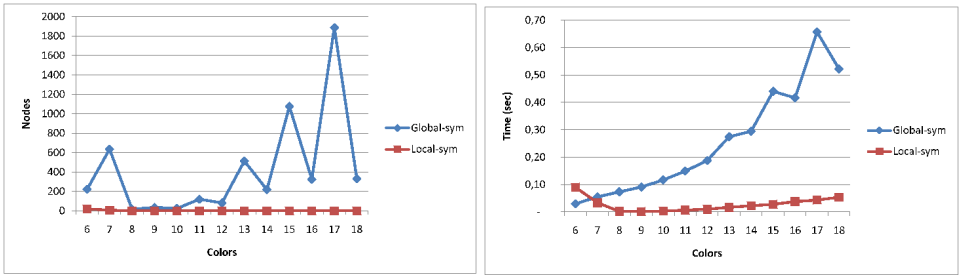


Fig. 3. Node and Time curves of the two symmetry methods on random graph coloring where $n = 30$ and $d = 0.5$

We reported in Figure 3 the practical results of the methods: *Global-sym*, and *Local-sym*, on the random graph coloring problem where the number of variables is $n = 30$ and where the density is ($d = 0.5$). The curves give the number of nodes respectivily the cpu time with respect to the number of colors for each search method.

We can see on the node curves (the curves at the left) that *Local-sym* detects and eliminates more symmetries than the *Global-sym* method and *Global-sym* is not stable for graph coloring. From the CPU time curves (the curves at the right), we can see that *Local-sym* is in average faster than *Global-sym* even that Saucy is run at each node. Local symmetry elimination is profitable for solving random graph coloring instances and outperforms dramatically global symmetry breaking on these problems.

7 Conclusion and perspectives

Here, we extended symmetry detection and elimination to local symmetry. That is, the symmetries of each CNF sub-formula defined at a given node of the search tree and which is derived from the initial formula by considering the partial assignment corresponding to that node. We adapted Saucy to compute this local symmetry by maintaining dynamically the graph of the sub-formula defined at each node of the search tree. Saucy is called with the graph of the local sub-formula as the main input, and then returns the set of generators of the automorphism group of the graph which is shown to be equivalent to the local symmetry group of the considered sub-formula. The proposed local symmetry detection method is implemented and exploited in the tree search method $LSAT$ to improve its efficiency. Experimental results confirmed that local symmetry breaking is profitable for SAT solving and improves global symmetry breaking on some of the considered problems, and its combination with global symmetry seems to be complementary.

As a future work, we are looking to implement some weakened symmetry conditions under which we may detect more symmetries, then experiment it and compare its results with the ones given here.

Another interesting point is to try to detect local variable symmetries and post dynamic constraints to break them, it will be important to compare the static approaches that detect only global symmetries with this approach.

References

1. F. A. Aloul, A. Ramani, I. L. Markov, and K. A. Sakallak. Symmetry breaking for pseudo-boolean satisfiabilty. *In ASPDAC'04*, pages 884–887, 2004.
2. Fadi A. Aloul, Arathi Ramani, Igor L. Markov, and Karem A. Sakallah. Solving difficult SAT instances in the presence of symmetry. In *Proceedings of the 39th Design Automation Conference (DAC 2002)*, pages 731–736. ACM Press, 2002.
3. B. Benhamou. Study of symmetry in constraint satisfaction problems. *PPCP'94*.
4. B. Benhamou and L. Sais. Theoretical study of symmetries in propositional calculus and application. *In CADE'11*, 1992.
5. B. Benhamou and L. Sais. Tractability through symmetries in propositional calculus. *In JAR*, 12:89–102, 1994.
6. B. Benhamou, L. Sais, and P. Siegel. Two proof procedures for a cardinality based language. *In STACS*, pages 71–82, 1994.
7. Belaïd Benhamou and Mohamed Réda Saïdi. Local symmetry breaking during search in csps. In *In CP*, pages 195–209, 2007.
8. W. Bibel. Short proofs of the pigeon hole formulas based on the connection method. *Automated reasoning*, (6):287–297, 1990.
9. Brown, C. A.Finkelstein, and L. P. W. Purdom. Bactrack searching in the presence of symmetry. *In Algebraic algorithms and error-correcting codes.*, (6):99–110, 1988.
10. D. Cohen, P. Jeavons, C. Jefferson, K.E. Petrie, and B. Smith. Symmetry definitions for constraint satisfaction problems. *In CP*, pages 17–31, 2005.
11. J. Crawford. A theoretical analysis of reasoning by symmetry in first-order logic. *Workshop on Tractable Resonning, AAAI-92*, 1992.
12. P. T. Darga, K. A. Sakallah, and I. L. Markov. Faster symmetry discovery using sparsity of symmetries. In *DAC*, 2008.
13. E.C. Freuder. Eliminating interchangeable values in constraints satisfaction problems. *AAAI-91*, pages 227–233, 1991.
14. I. P. Gent, T. Kelsey, S. A. Linton, J. Pearson, and C. M. Roney-Dougal. Groupoids and conditional symmetry. In *CP*, pages 823–830, 2007.
15. J. W. L. Glaisher. On the problem of the eight queens. *Philosophical Magazine*, 48(4):457–467, 1874.
16. A. Haken. The intractability of resolution. *T.Computer Science*, 39:297–308, 1985.
17. J.Crawford, M. L. Ginsberg, E. Luck, and Amitabha Roy. Symmetry-breaking predicates for search problems. In *KR'96*, pages 148–159. 1996.
18. J.G. Kalbfleisch and R.G. Stanton. On the maximal triangle-free edge-chromatic graphs in three colors. *combinatorial theory*, (5):9–20, 1969.
19. J. De Kleer. A comparison of atms and csp techniques. *IJCAI'89*, pages 290–296.
20. B. Krishnamurty. Short proofs for tricky formulas. *Acta Inf.*, (22):253–275, 1985.
21. R.C. Lyndon. *Notes of logic*. Van Nostrand Mathematical Studies, 1964.
22. B McKay. Practical graph isomorphism. In *Congr. Numer. 30*, pages 45–87, 1981.
23. C. Mears, M. Garcia de la Banda, and M. Wallace. On implementing symmetry detection. In *SymCon'06*, pages 1–8, 2006.
24. R. Ostrowski, B. Mazure, and L. Sais. Lsat solver. In *SAT*, 2002.
25. J. F. Puget. Automatic detection of variable and value symmetries. In *CP'05*, pages 474–488.
26. J.F. Puget. On the satisfiability of symmetrical constraint satisfaction problems. In *proceedings of ISMIS*, pages 350–361, 1993.
27. P. Siegel. Representation et utilisation de la connaissance en calcul propositionnel, 1987. Thèse d'état, GIA - Luminy (Marseille).

Solving linear bilevel programming by DC algorithm

M. S. Radjef* and A. Anzi **

Laboratory of Modelling and Optimization
of systems (LAMOS), Béjaia, Algeria

Abstract. In this paper we propose an algorithm for solving bilevel linear programming problems, in which the second level problem is replaced by its Karush-Kuhn-Tucker optimality conditions. This algorithm is a combination of the DCA algorithm in DC programming and the exact penalty methods.

Keywords : Bilevel linear programming, DC programming, DCA algorithm, KKT optimality conditions, exact penalty.

1 Introduction

Bilevel programming is a tool for modelling two level hierarchical systems. This class of programs constitutes a branch of mathematical programming in which the constraints are, partially, determined by another optimization problem.

Bilevel programming is motivated by the static noncooperative game theory of Stackelberg. In these problems, the upper level is termed Leader and the lower level is termed Follower. The control of variables is partitioned between the decision maker's who attempt to optimize their individual objectives. The Leader goes first in order to optimize his objective function. The Follower observes the Leader's decision and constructs his decision.

Bilevel linear programming (BLP) is one of the basic models of bilevel programming where the objective function and the constraints of the upper level and the lower level problems are all linear. The (BLP) problem can be formulated as follows:

$$\max_x \ F(x,y) = c_1^t x + d_1^t y, \tag{1a}$$

$$s.t. \ \ A_1 x + B_1 y \leq b_1, \tag{1b}$$

$$x \geq 0; \tag{1c}$$

$$\max_y \ f(x,y) = c_2^t x + d_2^t y, \tag{1d}$$

$$s.t. \ \ A_2 x + B_2 y \leq b_2, \tag{1e}$$

$$y \geq 0, \tag{1f}$$

* LAMOS, University of Béjaia, Algeria.
** LAMOS, University of Béjaia, Algeria.

where x, c_1, $c_2 \in \mathbb{R}^{n_1}$; y, d_1, $d_2 \in \mathbb{R}^{n_2}$; $b_1 \in \mathbb{R}^{m_1}$; $b_2 \in \mathbb{R}^{m_2}$; $A_1 \in \mathbb{R}^{m_1 \times n_1}$; $B_1 \in \mathbb{R}^{m_1 \times n_2}$; $A_2 \in \mathbb{R}^{m_2 \times n_1}$ and $B_2 \in \mathbb{R}^{m_2 \times n_2}$.
Following [9] we give these definitions:

1. *Constraint set of the problem*

$$S = \{(x, y) : A_1x + B_1y \leq b_1, A_2x + B_2y \leq b_2, x \geq 0, y \geq 0\}.$$

2. *Feasible set for the Follower for each fixed x*

$$S(x) = \{y \in \mathbb{R}^{n_2} : B_2y \leq b_2 - A_2x, y \geq 0\}.$$

3. *Projection of S onto the Leader's space*

$$P(X) = \{x \in \mathbb{R}^{n_1} : \exists y \in \mathbb{R}^{n_2}, A_1x + B_1y \leq b_1, A_2x + B_2y \leq b_2, x \geq 0, y \geq 0\}.$$

4. *Follower's rational reactions set for $x \in P(X)$*

$$R(x) = \{y \in \mathbb{R}^{n_2} : y = \arg\max[f(x, \hat{y}) : \hat{y} \in S(x)]\}.$$

5. *Inducible region*

$$RI = \{(x, y) \in S,\ y \in R(x)\}.$$

The inducible region represents the feasible set over which the Leader may optimize his objective.

There are mainly two ways to formulate a (BLP) : the pessimistic formulation and the optimistic one. The formulation considered in this paper is the optimistic formulation. In this case, an optimal solution of the (BLP) is defined as follows:

Definition 1 *[15] A point $(x^*, y^*) \in RI$ is an optimal solution of problem (1) if*

$$c_1^t x^* + d_1^t y^* \geq c_1^t x + d_1^t y,\ \forall (x, y) \in RI.$$

The (BLP) is a nonconvex and NP-Hard problem. Such characteristics are proper even if constraints (1b) do not exist. This is the most studied version of (BLP) . To solve this problem, many approaches have been proposed in the literature. These methods can be divided into the following categories:

(a) Methods based on vertex enumeration [8],[10],[13].
(b) Methods based on KKT reformulation [6],[14],[21].
(c) Methods based on meta-heuristics [20],[25].

In this paper, we consider the (BLP) with upper level constraints and use the second approach which consists in replacing the Follower's problem (1d)-(1f)

with its Karush-Khun-Tucker optimality conditions. The resulting problem has the form (see [9], *proposition 5.2.2*) :

$$\max_{x,y} \; F(x,y) = c_1^t x + d_1^t y \tag{2a}$$

$$A_1 x + B_1 y + e = b_1 \tag{2b}$$

$$A_2 x + B_2 y + w = b_2 \tag{2c}$$

$$B_2^t u - v = d_2 \tag{2d}$$

$$v^t y + u^t w = 0 \tag{2e}$$

$$x \geq 0,\; y \geq 0,\; u \geq 0,\; v \geq 0,\; w \geq 0, e \geq 0. \tag{2f}$$

where $w \in \mathbb{R}^{m_2}$ and $e \geq 0$ are slack variables, $v \in \mathbb{R}^{n_2}$ and $u \in \mathbb{R}^{m_2}$ are dual variables. Then we apply an exact penalization to the nonconvex constraints (2e) in order to transform problem (2) in a concave minimization problem under linear constraints. Finally, we use DC programming and DCA algorithm to solve the resulting problem .

DC programming and DCA algorithm [3],[5] have been introduced by P.D. Tao in 1986. DCA is a primal-dual method for solving a general DC program. In general, DCA converges to a local solution, however, il was observed in practice that it converges quite often to a global one. This method has proved its efficiency from both theoretical and numerical viewpoints and has been successfully applied to a large number of nonconvex and nondifferentiable problems in various domains.

The paper is organized as follows. In section 2, we describe how to reformulate the problem via an exact penalty technique. Section 3 is devoted to DC programming and DCA algorithm for solving the resulting penalized problem. Computational results are presented in section 4, while some conclusion is presented in the last section.

2 Reformulation via exact penalty

In this section, we use an exact penalty to reformulate the problem (2) in the form of a concave minimization program. For this we first introduce some useful notations. Let

$z = \begin{pmatrix} x\ y\ e\ w\ v\ u \end{pmatrix}^t \in \mathbb{R}^n, \quad c = \begin{pmatrix} -c_1\ -d_1\ 0\ 0\ 0\ 0 \end{pmatrix}^t \in \mathbb{R}^n,$

$A = \begin{pmatrix} A_1 & B_1 & I_{m_1} & 0 & 0 & 0 \\ A_2 & B_2 & 0 & I_{m_2} & 0 & 0 \\ 0 & 0 & 0 & 0 & -I_{n_2} & B_2^t \end{pmatrix} \in R^{m \times n}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ d_2 \end{pmatrix} \in \mathbb{R}^m,$

$E_u = \begin{pmatrix} 0\ 0\ 0\ 0\ 0\ I_{m_2} \end{pmatrix}, \quad E_v = \begin{pmatrix} 0\ 0\ 0\ 0\ I_{n_2}\ 0 \end{pmatrix},$

$E_w = \begin{pmatrix} 0\ 0\ 0\ I_{m_2}\ 0\ 0 \end{pmatrix}, \quad E_y = \begin{pmatrix} 0\ I_{n_2}\ 0\ 0\ 0\ 0 \end{pmatrix},$

where I_k is $k \times k$ identity matrix ; 0 is zero matrix with appropriate dimension for each case, with $n = n_1 + 2n_2 + m_1 + 2m_2$; $m = m_1 + m_2 + n_2$.
Using these notations, we have :

$u^t w = (E_u z)^t (E_w z) = z^t (E_u^t E_w) z = z^t D^1 z$, and $v^t y = (E_v z)^t (E_y z) = z^t (E_v^t E_y) z = z^t D^2 z$,
which gives : $u^t w + v^t y = z^t D^1 z + z^t D^2 z = z^t D z$ with $D^1 + D^2 = D$.

Note that the elements $d_{ij} (i = \overline{1,n},\ j = \overline{1,n})$ of matrix D are all nonnegative. Setting $Dz = q(z)$, problem (2) can be written as

$$\min \left\{ F(z) = c^t z, Az = b, z^t q(z) = 0, z \geq 0 \right\} \tag{3}$$

with $q(z) \geq 0,\ \forall z \geq 0$.

Consider the convex set $\mathcal{Z} = \{z \in \mathbb{R}^n : Az = b, z \geq 0\}$, and let be the function $\Psi : \mathbb{R}^n \to \mathbb{R}$ defined as $\Psi(z) = \sum_{i=1}^{n} \min\{q_i(z),\ z_i\}$. Ψ is a finite concave and nonnegative function on $\mathcal{Z}$. Then we have

$$\{z \in \mathcal{Z},\ z^t q(z) = 0\} = \{z \in \mathcal{Z},\ \Psi(z) \leq 0\}.$$

Thus, the problem (3) can be rewritten in the form

$$\alpha = \min\{F(z)\ :\ z \in \mathcal{Z},\ \Psi(z) \leq 0\}. \tag{4}$$

If $\mathcal{Z}$ is a nonempty and bounded set, then ([2], *theorem 1*) there exists $k_0 \geq 0$ such that for every $k > k_0$, problem (4) is equivalent to the following penalized problem:

$$\alpha(k) = \min\{F(z) + k\Psi(z) : z \in \mathcal{Z}\}. \tag{5}$$

which is a concave minimization program under linear constraints.

3 DCA for solving problem (5)

This section is devoted to the DC decomposition of problem (5) and its resolution with DCA algorithm.

3.1 DC programming

Let $\Gamma_0(X)$ denotes the set of all lower semicontinuous proper convex functions on X. A general DC program has the form

$$\alpha = \inf\{f(x) = g(x) - h(x)\ :\ x \in X\}, \tag{6}$$

where $g,\ h \in \Gamma_0(X)$ are called DC components of the function f and $g - h$ is the DC decomposition of f.

The dual of (6) is the DC program

$$\alpha = \inf\{h^*(y) - g^*(y) : y \in Y\}, \tag{7}$$

where g^* and h^* are respectively the conjugate function of g and h :

$$g^*(y) = \sup\{\langle x, y\rangle - g(x) : x \in X\}.$$

For problem (6) we have the following necessary local optimality conditions [2]

$$\emptyset \neq \partial h(x^*) \subset \partial g(x^*) \tag{8}$$

$$\emptyset \neq \partial g(x^*) \cap \partial h(x^*). \tag{9}$$

Such a point x^* is called critical point of $g - h$.

A function g is polyhedral convex on a convex polyhedral set $C \subset \mathbb{R}^n$ if it is of the form

$$g(x) = \max\{\langle a_i, x\rangle - \beta_i : \quad i = 1, ..., m\} + \chi_C(x),$$

where $a_i \in \mathbb{R}^n, \beta_i \in \mathbb{R}, i = 1, ..., m$ and χ_C is the indicator function of C

$$\chi_C(x) = 0 \ \text{ if } x \in C, \quad +\infty \ \text{ otherwise.}$$

A DC problem is called polyhedral DC program if g or h are polyhedral convex functions. For this class of DC programs condition (8) is also sufficient.

DCA algorithm is a descent method without linesearch, consisting of the construction of the two sequences $\{x^i\}$ and $\{y^i\}$, (candidates for being primal and dual solutions, respectively), such that their corresponding limit points satisfy local optimality conditions. Recall that there are two forms of DCA: the simplified DCA (or DCA) and the complete DCA. In practice we use the first because it is less expensive.

DCA algorithm:

1 : x^0 given.
2 : Compute $y^i \in \partial h(x^i)$.
3 : Compute $x^{i+1} \in \partial g^*(y^i)$.
4 : if a convergence criterion is satisfied **Stop**; **else** $i = i + 1$ and **goto** 2.

3.2 DCA for solving (5)

We first prove that (5) is a DC program then we present DCA applied to the resulting DC program.
Denote by $\chi_{\mathcal{Z}}$ the indicator function of $\mathcal{Z}$ and let g and h given by

$$g(z) = \chi_{\mathcal{Z}}(z) \ \text{ and } \ h(z) = -F(z) - k\Psi(z). \tag{10}$$

Therefore g and h are convex functions and problem (5) is equivalent to the DC program of the form

$$\min\{g(z) - h(z) : z \in \mathbb{R}^n\}. \tag{11}$$

The application of the DCA to problem (11) consists of computing the two sequences $\{t^i\}$ and $\{z^{i+1}\}$ defined by

$$t^i \in \partial h(z^i) \ \text{ and } \ z^{i+1} \in \partial g^*(t^i).$$

Using the rules in convex analysis we compute $\{t^i\}$ and $\{z^{i+1}\}$.

Computation of $t^i \in \partial h(z^i)$: we choose $t^i \in \partial\big(-c^t z^i - k\sum_{j=1}^{n} \min\{q_j(z^i),\ z_j^i\}\big)$ of the form

$$t^i = -c + k\theta^i, \tag{12}$$

where $\theta^i \in \sum_{j=1}^{n} \partial\big(\max\{-q_j(z^i),\ -z_j^i\}\big)$ and $q_j(z^i) = D_j z^i$.
Let be

$$\theta^i = -\sum_{j=1}^{n} \begin{cases} D_j^t, & \text{if } \ z_j^i > D_j z^i, \\ e_j, & \text{if } \ z_j^i < D_j z^i, \\ \gamma e_j + (1-\gamma) D_j^t, & \text{if } \ z_j^i = D_j z^i, \end{cases} \tag{13}$$

where D_j is the *j-th* line of matrix D, e_j is the *j-th* unit vector of $\mathbb{R}^n$ and $\gamma \in [0,1]$.
Hence, θ^i given by (13) is an element of $\sum_{j=1}^{n} \partial\big(\max\{-D_j z^i, -z^i\}$.

Computation of $z^{i+1} \in \partial g^*(t^i)$: following [22], we can choose z^{i+1} as the solution of the following linear programming problem

$$\min\{-\langle z,\ t^i\rangle : z \in \mathcal{Z}\}, \tag{14}$$

DCABLP (DCA for (5))

1 : Let z^0 initial guess, $\epsilon > 0$, $k \in \mathbb{R}_+$, $\gamma \in [0,1]$ and $\lambda > 0$. Set $i = 0$.
2 : Compute $t^i \in \partial h(z^i)$ using (12).
3 : Compute $z^{i+1} \in \partial g^*(t^i)$ by solving (14).
4 : If $y^{i+1} \in \arg\max\big\{f(x^{i+1}, y) : B_2 y \le b_2 - A_2 x^{i+1},\ y \ge 0\big\}$, **then** go to **5**;
otherwise go to **7**.
5 : Compute (v^*, u^*), solution of the dual problem
$\min\{u^t(b_2 - A_2 x^{i+1}) : B_2^t u - v = d_2,\ u \ge 0,\ v \ge 0\}$, hence
$z^{i+1} = (x^{i+1}, y^{i+1}, e^{i+1}, w^{i+1}, v^*, u^*)$.
6 : If $\|z^{i+1} - z^i\|/(\|z^i\| + 1) \le \epsilon$, **then** stop z^{i+1} is optimal solution of (5); and (x^*, y^*) is optimal for (1).
otherwise go to **7**.
7 : Set $z^i = z^{i+1}$, $i = i + 1$, $k = k + \lambda$ and go to **2**.

Remark 1 *Problem (5), with DC decomposition (10), is a polyhedral DC program since $g = \chi_{\mathcal{Z}}$ is polyhedral convex function [22]. In this case DCA applied to (5) has finite convergence [3],[5].*

Remark 2 *In step 4 of the algorithm, we test the feasibility of the solution (x^{i+1}, y^{i+1}) for the (BLP) . If the test in step 4 is satisfied we have $y^{i+1} \in$*

$R(x^{i+1})$ *(*see definition 4*). Since* $(x^{i+1}, y^{i+1}) \in S$*, then we have* $(x^{i+1}, y^{i+1}) \in RI$ *which implies that* (x^{i+1}, y^{i+1}) *is a feasible solution.*

Remark 3 *If* $z^* = (x^*, y^*, e^*, w^*, v^*, u^*)$ *is an optimal solution of (5), then the optimality of* (x^*, y^*) *for problem (1) is provided by step 5. In fact, suppose that* z^* *is an optimal solution for (5) with* $k = \tilde{k}$*. Then, we have*

$$g(z^*) - h(z^*) \leq g(z) - h(z),\ \forall z \in \mathcal{Z}.$$

Since $g = \chi_{\mathcal{Z}}$ *and* $z, z^* \in \mathcal{Z}$*, we have*

$$0 - h(z^*) \leq 0 - h(z),\ \forall z \in \mathcal{Z},$$

which gives

$$F(z^*) + \tilde{k}\Psi(z^*) \leq F(z) + \tilde{k}\Psi(z),\ \forall z \in \mathcal{Z}. \tag{15}$$

$$c^t z^* + \tilde{k}\Psi(z^*) \leq c^t z + \tilde{k}\Psi(z),\ \forall z \in \mathcal{Z}. \tag{16}$$

From remark 2, (x^*, y^*) *is an optimal solution of the follower's problem. Moreover, from duality in linear programming, if* (v^*, u^*) *is the optimal solution of the follower's dual problem, then the complementary constraints are satisfied ; that is the penalty function* $\Psi(z^*)$ *is zero. We have then*

$$c^t z^* \leq c^t z,\ \forall z \in \mathcal{Z}.$$

Hence

$$-c^t z^* \geq -c^t z,\ \forall z \in \mathcal{Z},$$

which gives

$$c_1^t x^* + d_1^t y^* \geq c_1^t x + d_1^t y,\ \forall (x, y) \in RI.$$

Then (x^*, y^*) *is an optimal solution of problem (1).*

Initial point

In order to find an appropriate initial point to DCABLP, we use DCA to solve the following problem

$$0 = \min\{\Psi(z) : \bar{A}z = \bar{b}, z \geq 0\} \tag{17}$$

where

$$\bar{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ A_2 & B_2 & 0 & I_{m_2} & 0 & 0 \\ 0 & 0 & 0 & 0 & -I_{n_2} & B_2^t \end{pmatrix} \quad \text{and} \quad \bar{b} = \begin{pmatrix} 0 \\ b_2 \\ d_2 \end{pmatrix},$$

which is a concave minimization problem and whose optimal value is known (equal to zero).

4 Computational results

In this section we report computational experiments with the proposed algorithm DCABLP. The algorithm has been coded in MATLAB and run on a PC Pentium IV 3.00 GHz, RAM 512Mb using the simplex algorithm for solving linear subproblems. In order to evaluate the performance of the proposed algorithm we tested it on a set of 11 problems whose optimal solutions are known. The initial point was calculated by solving problem (17). For parameter γ in (13), we took the value $\gamma = 0.5$. The algorithm is stopped when $\epsilon \leq 10^{-3}$.

The results of the algorithm are reported in table 1 where we use the following notations:

k: penalty parameter.
λ: increasing parameter.
t: the total time for the algorithm and is given in seconds.
It: the number of iterations of the algorithm.
$(x^*; y^*)$: the optimal solution of the problem.
$(F^*; f^*)$: the Leader's and the Follower's optimal value respectively.
"$-$" the algorithm failed to solve the problem.

Test problems

$$\text{P1 [24]}: \begin{cases} \max F(x,y) = 8x_1 + 4x_2 - 4y_1 + 40y_2 + 4y_3 \\ x \geq 0 \\ \max f(x,y) = -x_1 - 2x_2 - y_1 - y_2 - 2y_3 \\ 0x_1 + 0x_2 - y_1 + y_2 + y_3 \leq 1 \\ 2x_1 + 0x_2 - y_1 + 2y_2 - 0.5y_3 \leq 1 \\ 0x_1 + 2x_2 + 2y_1 - y_2 - 0.5y_3 \leq 1 \\ y \geq 0 \end{cases} \qquad \text{P2 [6]}: \begin{cases} \max F(x,y) = x + 3y \\ x \geq 0 \\ \max f(x,y) = x - 3y \\ -x - 2y \leq -10 \\ x - 2y \leq 6 \\ 2x - y \leq 21 \\ x + 2y \leq 38 \\ -x + 2y \leq 18 \\ y \geq 0 \end{cases}$$

$$\text{P3 [20]}: \begin{cases} \max F(x,y) = 8x_1 + 4x_2 - 4y_1 + 40y_2 + 4y_3 \\ x_1 + 2x_2 - y_3 \leq 1.3 \\ x \geq 0 \\ \max f(x,y) = -2y_1 - y_2 - 2y_3 \\ 0x_1 + 0x_2 - y_1 + y_2 + y_3 \leq 1 \\ 4x_1 + 0x_2 - 2y_1 + 4y_2 - y_3 \leq 2 \\ 0x_1 + 4x_2 + 4y_1 - 2y_2 - y_3 \leq 2 \\ y \geq 0 \end{cases} \qquad \text{P4 [10]}: \begin{cases} \max F(x,y) = x + y \\ x \geq 0 \\ \max f(x,y) = 0x - y \\ -4x - 3y \leq -19 \\ x + 2y \leq 11 \\ 3x + y \leq 13 \\ y \geq 0 \end{cases}$$

$$\text{P5 [26]}: \begin{cases} \max F(x,y) = -x - 5y \\ x \geq 0 \\ \max f(x,y) = 0x + y \\ -x - y \leq -8 \\ -3x + 2y \leq 6 \\ x + 4y \leq 48 \\ x - 5y \leq 9 \\ y \geq 0 \end{cases} \qquad \text{P6 [23]}: \begin{cases} \max F(x,y) = -x_1 - 2x_2 - 2y_1 + y_2 \\ x_1 + x_2 + 0.5y_1 + y_2 \leq 6 \\ x \geq 0 \\ \max f(x,y) = -y_1 + 2y_2 \\ -x_1 + 2x_2 + 0y_1 + y_2 \leq 4 \\ -x_1 - x_2 + y_1 + y_2 \leq 5 \\ y \geq 0 \end{cases}$$

P7 [14] :
$$\begin{cases} \max F(x,y) = -0.4x_1 - y_1 - 5y_2 + 0y_3 + 0y_4 \\ x \geq 0 \\ \max f(x,y) = 0x_1 + 0y_1 + 0.5y_2 - y_3 - 2y_4 \\ -0.1x_1 - y_1 - y_2 + 0y_3 + 0y_4 \leq -1 \\ 0.2x_1 + 0y_1 + 1.25y_2 + 0y_3 - y_4 \leq -1 \\ -x + 6y_1 + y_2 - 2y_3 + 0y_4 \leq 1 \\ y \geq 0 \end{cases}$$

P8 [9] :
$$\begin{cases} \max F(x,y) = -x + 4y \\ x \geq 0 \\ \max f(x,y) = 0x - y \\ -x - y \leq -3 \\ -2x + y \leq 0 \\ 2x + y \leq 12 \\ -3x + 2y \geq -4 \\ y \geq 0 \end{cases}$$

P9 [24] :
$$\begin{cases} \max F(x,y) = 2x_1 - x_2 - 0.5y_1 \\ x_1 + x_2 \leq 2 \\ x \geq 0 \\ \max f(x,y) = 0x_1 + 0x_2 + 4y_1 - y_2 \\ -2x_1 + y_1 - y_2 \leq -2.5 \\ x_1 - 3x_2 + y_2 \leq 2 \\ y \geq 0 \end{cases}$$

P10 [24] :
$$\begin{cases} \max F(x,y) = -x + 4y \\ x \geq 0 \\ \max f(x,y) = 0x - y \\ -2x + y \leq 0 \\ 2x + 5y \leq 108 \\ 2x - 3y \leq -4 \\ y \geq 0 \end{cases}$$

P11 [8] :
$$\begin{cases} \max F(x,y) = 2x_1 - x_2 - x_3 + 2x_4 + x_5 - 3.5x_6 - y_1 - 1.5y_2 + 3y_3 \\ x \geq 0 \\ \max f(x,y) = 0x_1 + 2x_2 + 0x_3 + 0x_4 - x_5 + 0x_6 + 3y_1 - y_2 - 4y_3 \\ -x_1 + 0.2x_2 + 0x_3 + 0x_4 + x_5 + 2x_6 - 4y_1 + 2y_2 + y_3 \leq 12 \\ x_1 + 0x_2 + x_3 - 2x_4 + 0x_5 + 0x_6 + 0y_1 - 4y_2 + y_3 \leq 10 \\ 5x_1 + 0x_2 + 0x_3 + x_4 + 0x_5 + 3.2x_6 + 2y_1 + 2y_2 + 0y_3 \leq 15 \\ 0x_1 - 3x_2 + 0x_3 - x_4 + x_5 + 0x_6 - 2y_1 + 0y_2 + 0y_3 \leq 12 \\ -2x_1 - x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 + 0y_1 - y_2 + y_3 \leq -2 \\ 0x_1 + 0x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 - y_1 - 2y_2 - y_3 \leq -2 \\ 0x_1 - 2x_2 - 3x_3 + 0x_4 - x_5 + 0x_6 + 0y_1 + 0y_2 + 0y_3 \leq -3 \\ y \geq 0 \end{cases}$$

Pb	*(k; λ)*	$(x^*; y^*)$	$(F^*; f^*)$	*t*	*It*
1	(1;0.1)	-	-	-	-
	(1;0.5)	(0 0.9 0 0.6 0.4)	(29.2; -3.2)	4.06	19
	(1;1)	(0 0.9 0 0.6 0.4)	(29.2; -3.2)	2.48	11
	(5;1)	(0 0.9 0 0.6 0.4)	(29.2; -3.2)	2.07	7
	(5;5)	(0 0.9 0 0.6 0.4)	(29.2; -3.2)	1.26	3
	(10;5)	(0 0.75 0 0.5 0)	(23; -2)	0.7	2
2	(0.01;0.01)	-	-	-	-
	(0.1;0.01)	(16 11)	(49; -17)	3.14	3
	(0.1;0.1)	(16 11)	(49; -17)	6.92	6
	(1;1)	(16 11)	(49; -17)	0.65	2
	(5;1)	(12 3)	(21; 3)	0.17	2
	(5;5)	(12 3)	(21; 3)	1.23	2
	(10;5)	(12 3)	(21; 3)	0.18	2
3	(5;5)	-	-	-	-
	(10;5)	(0 0.78 0 0.43 0.26)	(21.36; -0.95)	1.18	2
	(15;5)	(0 0.78 0 0.43 0.26)	(21.36; -0.95)	1.44	2
	(20;5)	(0 0.78 0 0.43 0.26)	(21.36; -0.95)	1.20	2
4	(0.1;0.1)	-	-	-	-
	(1;0.1)	(4 1)	(5; -1)	0.09	2
	(1;1)	(4 1)	(5; -1)	1.09	4
	(5;1)	(1 5)	(6; -5)	0.7	2
	(5;5)	(1 5)	(6; -5)	0.6	2
	(10;5)	(1 5)	(6; -5)	0.6	2
5	(0.1;0.1)	-	-	-	-
	(1;0.1)	(2 6)	(-32; 6)	1.5	3
	(1;1)	(2 6)	(-32; 6)	1.7	3
	(5;1)	(2 6)	(-32; 6)	0.5	2
	(5;5)	(2 6)	(-32; 6)	0.6	2
	(10;5)	(2 6)	(-32; 6)	0.56	2
6	(0.1;0.01)	(4,10)	(4; 10)	1.15	2
	(1;0.1)	(4,10)	(4; 10)	1.01	2
	(5;1)	(4,10)	(4; 10)	1.3	2
	(5;5)	(4,10)	(4; 10)	0.96	2
	(10;5)	(4,10)	(4; 10)	0.5	2

7	(0.1;0.01)	-	-	-	-
	(0.1;0.1)	(0 0 1 0 2.25)	(-5; -4)	0.68	2
	(1;1)	(0 0 1 0 2.25)	(-5; -4)	0.6	2
	(5;1)	(0 0 1 0 2.25)	(-5; -4)	0.56	2
	(5;5)	(0 0 1 0 2.25)	(-5; -4)	1.06	2
	(10;5)	(0 0 1 0 2.25)	(-5; -4)	0.96	2
8	(0.1;0.1)	-	-	-	-
	(1;0.1)	(4 4)	(12; -4)	3.96	8
	(1;1)	(4 4)	(12; -4)	1.51	3
	(5;1)	(2 1)	(2; -1)	0.56	2
	(5;5)	(2 1)	(2; -1)	0.21	2
	(10;5)	(2 1)	(2; -1)	0.65	2
9	(0.1;0.1)	-	-	-	-
	(1;0.1)	(2 0 1.5 0)	(3.25; 6)	2.12	2
	(1;1)	(2 0 1.5 0)	(3.25; 6)	1.40	4
	(5;1)	(2 0 1.5 0)	(3.25; 6)	1.42	2
	(5;5)	(2 0 1.5 0)	(3.25; 6)	1.39	2
	(10;5)	(2 0 1.5 0)	(3.25; 6)	1.43	2
10	(0.1;0.1)	-	-	-	
	(1;0.1)	(19 14)	(37; -14)	0.64	2
	(1;1)	(19 14)	(37; -14)	0.17	2
	(5;1)	(19 14)	(37; -14)	1.28	6
	(5;5)	(19 14)	(37; -14)	0.5	2
11	(0.1;0.1)	-	-	-	-
	(0.5;0.1)	(0 4 0 15 9.2 0 0 0 2)	(41.2; -9.2)	1.17	2
	(1;1)	(0 2 0 11 19.6 0 2 0 0)	(37.6; -13.6)	0.7	2
	(5;1)	(1 0 0 6 21 0 2 0 0)	(33; -19)	2.64	6
	(5;5)	(1 0 0 6 21 0 2 0 0)	(33; -19)	1.2	2
	(10;5)	(0.5 0 0 0 3.93 3.92 0 1 0)	(-8.04; -4.93)	0.6	2

Tab. 1: Numerical results for DCABLP

Comments: from numerical experiments, we observe that

- the algorithm with the starting procedure is efficient: in most problems, with a good choice of the penalty parameter, it computed the global solution.
- the algorithm terminates rapidly; the average number of iterations is 2.
- the total time of the algorithm is small; this result is normal because there are only linear programs to solve at each iteration.
- the computational requirements (choice of penalty parameter and increasing parameter) are greatly dependent on the problem structure.

In table 2 we give the comparison between our results and the results in some references: (/ : means that the execution time is not given in the reference)

	parameters	results in the paper				result in the references			
	$(k;\lambda)$	$(x^*;y^*)$	F^*	t	It	$(x^*;y^*)$	F^*	t	It
1 [24]	(5;1)	(0 0.9 0 0.6 0.4)	29.2	2.07	7	(0 0.89 0 0.59 0.39)	25.92	0.047	22
2 [6]	(1;1)	(16 11)	49	0.65	2	(16 11)	49	/	28
5 [26]	(5;5)	(2 6)	-32	0.6	2	(2.0002 5.9999)	-31.9999	/	34
6 [23]	(5;5)	(4 10)	4	0.96	2	(4 10)	4	/	11
9 [24]	(5;5)	(2 0 1.5 0)	3.25	1.39	2	(2 0 1.5 0)	3.25	0.037	107
10 [24]	(5;5)	(19 14)	37	0.5	2	(18.92 13.95)	36.88	0	8

Tab. 2: Comparison results

From table 2 the results in the paper accord with the results in the references. In addition, we can see that our algorithm gives the exact solution.

5 Conclusion

We have presented a DC optimization approach for solving bilevel linear optimization problems. The resulted DC program is polyhedral and the DCA algorithm has a finite convergence. Computational experiments show that the procedure of calculating the starting point is efficient, but the search of global solution remains sensitive to the choice of the penalty parameter. The proposed algorithm is fast, since it solves only linear subproblems at each iteration.

References

1. F.B. Akoa: Approches de points intérieurs et de la programmation DC en optimisation non convexe. Codes et simulations numériques industrielles. Thèse de Doctorat, Institut national des sciences appliquées de Rouen. (2005)
2. L.T.H. An and P.D. Tao: A continuous approach for globally solving linearly constrained quadratic zero-one programming problems. Optimization, 50: 93 − 120, (2001)
3. L.T.H. An and P.D. Tao: Convex analysis approach to dc programming: theory and applications. Acta Mathematica Vietnamica, 22: 289 − 355, (1997)
4. L.T.H. An and P.D. Tao: Solving a class of linearly constrained indefinite quadratic problems by dc algorithms. J. Glob. Optim., 11: 253 − 285, (1997)
5. L.T.H. An and P.D. Tao: The dc (difference of convex functions) programming and dca revisited with dc models of real world nonconvex optimization problems. Annals Oper. Res., 133: 23 − 46, (2005)
6. G. Anandalingam and D. J. White: A solution for the linear static Stackelberg problem using penalty functions. IEEE Trans. on Aut. Cont., 35: 1170 − 1173, (1990)
7. G. Anandalingam and T. L. Friesz: Hierarchical optimization : An introduction. Annals of Oper. Res., 34: 1 − 11, (1992)
8. J.F. Bard: An efficient point algorithm for a linear two stage optimization problem. Oper. Res., 31: 670 − 684, (1983)
9. J.F. Bard: Practical bilevel optimization : algorithms and applications. Kluwer academic publishers, Dordrecht (1998)
10. O. Ben-Ayed and C. E. Blair: Computational difficulties of bilevel linear programming. Oper. Res., 38: 556 − 560, (1990)
11. W.F. Bialas: Multilevel mathematical programming : An introduction. Technical report, Department of Industrial Engineering. University at Buffalo (2002)
12. W.F. Bialas and M.H. Karwan: Multilevel linear programming. Technical Report, Oper. Res. Program, Dept. of Industrial Eng., State University of New York at Buffalo, 1 − 78, (1978)
13. W.F. Bialas and M.H. Karwan: Two-level linear programming. Management Science, 30(8) :1004 − 1020, (1984)
14. M. Campêlo, S. Dantas and S. Scheimberg: A note on a penalty function approach for solving bilevel linear programs. J. Glob. Optim., 16: 245 − 255, (2000)

15. B. Colson, P. Marcotte and G. Savard: Bilevel programming : A survey. 4OR A Quarterly, J. Oper. Res., (2007)
16. S. Dempe: Foundations of Bilevel Programming. Kluwer academic publishers, Dordrecht (2000)
17. S. Dempe and A.G. Mersha. Linear bilevel programming with upper level constraints depending on the lower level solution. App. Math. and Comp., 180: 247−254 (2006)
18. J. Fortuny-Amat, B. McCarl: A representation and economic interpretation of a two level programming problem. Oper. Res., 321: 783 − 792, (1981)
19. A. Haurie, G. Savard, and D.J. White. A note on : An efficient point algorithm for a linear two stage optimization problem. Operations Research, 38(3) :553 − 555, (1990)
20. S.R. Hejazi, A. Memariania, G. Jahanshahloob and M.M. Sepehria. Linear bilevel programming solution by genetic algorithm. Comp. and Oper. Res., 29: 1913 − 1925, (2005)
21. Y. Lv, T. Hu, G. Wang and Z. Wan. A penalty function method based on Kuhn–Tucker condition for solving linear bilevel programming. App. Math. and Comp., (2006)
22. R.T. Rockafellar: Convex analysis. Princeton, USA (1970)
23. N.V. Thoai, Y. Yamamoto and A. Yoshise: Global optimization method for solving mathematical programs with linear complementarity constraints. J. Opt. Theory and App., 124(2) :467 − 490, (2005)
24. H. Tuy, A. Migdalas and N. T. Hoai-Phuong: A novel approach to Bilevel nonlinear programming. J. Glob. Optim., 38: 527 − 554, (2007)
25. G. Wang, Z. Wan, and X. Wang: Solving method for a class of bilevel linear programming based on genetic algorithms. Supported by the National Natural Science Foundation and the Doctoral Foundation in Ministry of Education of China.
26. D.L Zhu, Q. Xu and Z. Lin: A homotopy method for solving bilevel programming problem. J. Nonlinear Analysis, 57: 917 − 928, (2004)

QCSP$^+$non-bloquants : un cas spécial de problèmes quantifiés

Arnaud Lallouet et Jérémie Vautard

Université d'Orléans — LIFO
BP 6759 — F-45067 Orléans cedex 2
jeremie.vautard@univ-orleans.fr

Résumé. Ce papier présente un cas spécial de QCSP$^+$appelé QCSP$^+$non-bloquants, dans lesquels la les restrictions posées sur les quantificateurs ne vident jamais le domaine d'une variable. Intuitivement, ces cas spéciaux correspondent à des jeux opposant deux adversaires dans lesquels il n'est pas possible de bloquer toute possibilité de mouvement d'un joueur. Nous présentons des techniques de résolution basées sur ce cas spécial, ainsi que des exemples de modèles non-bloquants.

1 Introduction

Le formalisme de la programmation par contraintes quantifiées (QCSP) étend les CSP classiques en ajoutant des quantificateurs universels ou existentiels sur les variables, ce qui permet de modéliser des problèmes de complexité supérieure à NP, comme décider si, dans un jeu à deux joueurs, l'un des joueurs a la possibilité de jouer de manière à toujours gagner, quoi que puisse faire son adversaire.

La solution d'un tel problème ne peut pas être une simple affectation de toutes ses variables : en effet, les contraintes doivent être satisfaites quelles que soient les valeurs affectées aux variables quantifiées universellement. On peut voir une telle solution comme une famille de fonctions de Skolem, chacune de ces fonctions affectant à une variable existentielle une valeur de son domaine en fonction des valeurs des variables universelles qui la précède. On appelle une telle famille de fonctions une *stratégie*. Lorsque cette famille est telle que, pour toutes les affectations possibles des variables universelles, l'affectation des variables existentielles qui en découle satisfait toutes les contraintes du problèmes, il s'agit d'une *stratégie gagnante*. Un QCSP est vrai si, et seulement si il admet une telle stratégie gagnante.

Le terme *stratégie* vient du fait qu'un QCSP peut être vu comme un jeu opposant le "joueur universel" au "joueur existentiel", consistant à affecter tour à tour les variables du problème, le joueur existentiel cherchant à résoudre toutes les contraintes du problème tandis que le joueur universel cherche à l'en empêcher. De ce point de vue, les fonctions de Skolem constituant une stratégie gagnante indiquent au joueur existentiel les coups à jouer de manière à ce qu'il soit sûr de gagner.

Cette ressemblance a conduit [1] à définir une extension à ce formalisme, appelée QCSP$^+$. Cette extension rend la modélisation de problèmes plus aisée en permettant de décrire directement sous la forme d'un CSP les règles du jeu, en restreignant la portée des quantificateurs universels aux seules solutions de ce CSP. Ceci permet même à ce formalisme de décrire des cas où l'un des joueurs "perd la partie" faute de coup possible à jouer. Cependant, pour les instances où l'on sait que ce dernier cas ne peut pas se produire, il est possible d'améliorer l'algorithme de résolution décrit dans [1]. Cet article présente ces améliorations.

2 QCSP

Notations. Soit V un ensemble de variables, D étant la famille de leurs domaines. On note D_X le domaine d'une variable X. Soit W un sous-ensemble de V. On note D^W l'ensemble des n-uplets sur W, c'est à dire le produit cartésien $\Pi_{X \in W} D_X$ des domaines des variables de W. Par ailleurs, on note $|$ la projection d'un (ensemble de) tuples sur une ou plusieurs variables.

Contraintes et CSP. Une *contrainte* $c = (W, T)$ est composée d'un sous-ensemble de variables $W \subseteq V$ et d'une relation $T \subseteq D^W$. W et T sont aussi respectivement notés $var(c)$ e $sol(c)$. Une contrainte vide, c'est-à-dire telle que $sol(c) = \emptyset$ est *fausse* et notée $\bot$, tandis qu'une contrainte est dite *pleine* et notée $\top$ si, et seulement si $sol(()c) = D^W$.

Un *problème de satisfaction de contraintes*, ou *CSP*, est un ensemble de contraintes. On note $var(C) = \bigcup_{c \in C} var(c)$ l'ensemble de ses variables et $sol(C) = \bowtie_{c \in C} sol(c)$ l'ensemble de ses solutions, i.e. l'ensemble de toutes les affectations des variables de $var(C)$ qui satisfont toutes les contraintes. Un CSP vide est *vrai* et sera noté $\top$ tandis qu'un CSP contenant une contrainte vide est luimeme trivialement faux et sera noté $\bot$.

Contraintes quantifiées et QCSP. On appelle *qset* un couple (q, W) où $q \in \{\exists, \forall\}$ est un quantificateur et $W \subseteq V$ un sous-ensemble de variables.

Définition 1 (Préfixe). *Un* préfixe P *est une séquence de qsets* $[(q_0, W_0), \ldots, (q_{n-1}, W_{n-1})]$ *dans laquelle* $i \neq j \Rightarrow W_i \cap W_j = \emptyset$.

On note $P|_W$ la restriction de P aux variables de l'ensemble W.

On dit qu'une variable X est *déclarée* dans un qset (q_i, W_i) si $X \in W_i$. Un QCSP est alors défini en ajoutant un CSP à un préfixe ;

Définition 2 (QCSP). *Un* CSP quantifié *ou* QCSP *est un couple* (P, G) *dans lequel* P *est un préfixe et* G *un CSP appelé* goal *tel que* $var(G) \subseteq var(P)$.

Exemple 3 (QCSP). La formule suivante :

$$\exists X \in \{0,1\},\ \forall Y \in \{0,1\},\ \exists Z \in \{1,2\}\ .\ X + Y = Z$$

est représentée par le QCSP suivant :

$$Q = ([(\exists, X), (\forall, Y), (\exists, Z)], \{X + Y = Z\})$$

Dans cet exemple, le préfixe est $[(\exists, X), (\forall, Y), (\exists, Z)]$ tandis que le goal se limite à l'unique contrainte $\{X + Y = Z\}$

Résolution Comme nous l'évoquions en introduction, contrairement au cas des CSP, la notion de solution d'un QCSP ne peut plus être une simple affectation des variables du problème. Une solution doit en effet expliciter les valeurs prises par les variables existentielles en fonction des variables universelles de manière à ce que le goal reste satisfait quelles que soient les valeurs prises par ces dernières. On appelle une telle solution une *stratégie gagnante*, qui peut être vue comme un ensemble de fonctions de Skolem, donnant une valeur à chaque variable existentielle en fonction des variables universelles précédentes (comme c'est le cas par exemple dans [2]). Le fait qu'une telle stratégie gagnante existe pour un QCSP donné prouve la formule logique associée à ce QCSP. On note $\text{WIN}(Q)$ l'ensemble des stratégies gagnantes d'un QCSP Q.

Sémantique Nous utilisons dans ce papier une sémantique purement *décisionnelle* des QCSP, c'est-à-dire donnant une valeur $\top$ ou $\bot$ à un QCSP.

Définition 4 (Sémantique décisionnelle d'un QCSP). *La sémantique décisionnelle $[\![Q]\!]_d$ d'un QCSP Q est :*

$[\![Q]\!]_d = \top$ *si* $Win(Q) \neq \emptyset$, $\bot$ *sinon.*

QCSP$^+$*.* Les QCSP$^+$ introduisent des restricteurs sur les quantificateurs d'un QCSP en modifiant la nature du préfixe : en plus du quantificateur et d'un ensemble de variables, chaque qset inclut un CSP dont les solutions sont les valeurs "autorisées" pour ses variables. Les QCSP$^+$ ont été présentés pour la première fois dans [1] et motivés principalement par des problèmes de modélisation en QCSP classiques.

On définit un *rqset* comme un triplet (q, W, C) où (q, W) contitue un qset tel que définit plus haut, et C un CSP que l'on appelle *restricteur*. Le sens d'un tel objet est de restreindre les variables de W à prendre les seuls valeurs satisfaisant C. On étend la notion de préfixe aux rqsets en continuant à requérir la condition $i \neq j \Rightarrow W_i \cap W_j = \emptyset$. De plus, chaque CSP C_i doit être tel que $var(C_i) \subseteq (W_1 \cup \ldots \cup W_i)$.

Définition 5 (QCSP$^+$). *Un QCSP*$^+$ *est un couple $Q = (P, G)$ avec P un préfixe de rqsets et G le CSP goal.*

Notons qu'un QCSP classique peut être vu comme un QCSP$^+$dont tous les C_i sont vides. La définition de la stratégie d'un QCSP$^+$est la même que celle d'un QCSP. en revanche, celle de stratégie gagnante change légèrement : une

stratégie gagnante est une stratégie pour laquelle tout coup possible du joueur universel se termine par un scénario gagnant. Comme dans un QCSP classique, cela peut se produire quand toutes les contraintes du goal sont satisfaites, mais il peut aussi arriver que le CSP d'un rqset universel devienne inconsistant, auquel cas le scénario devient vrai, quelle que soit l'affectation des variables restantes. La sémantique d'un QCSP$^+$ peut donc être définie de manière inductive :

Définition 6 (Sémantique d'un QCSP$^+$). *La sémantique $[\![Q]\!]$ d'un QCSP$^+$ Q est définie de la manière suivante :*

- $[\![([], G)]\!] = (sol(G) \neq \emptyset \ ? \ \top : \bot)$.
- $[\![([(\exists, W, C)|P'], G)]\!] = \bigvee_{t \in sol(C)}([\![P'[W \leftarrow t], G[W \leftarrow t]]\!])$
- $[\![([(\forall, W, C)|P'], G)]\!] = \bigwedge_{t \in sol(C)}([\![P'[W \leftarrow t], G[W \leftarrow t]]\!])$

Comme nous l'avons évoqué plus haut, $sol(C)$ peut être vide. Dans ces cas, on considère que $\bigvee(\emptyset) = \bot$ et $\bigwedge(\emptyset) = \top$.

3 QCSP$^+$ non-bloquants

3.1 Intuition et définition

Soit Q un QCSP$^+$ (P, G) tel que $P = [(q_0, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$. Considérons-le sous l'angle du jeu entre le joueur $\exists$ et le joueur $\forall$. Le premier joueur affecte des valeurs aux variables W_0 de manière à satisfaire C_0. Ensuite, son adversaire affecte W_1 tel que C_1 soit satisfait, et ainsi de suite jusqu'à ce que toutes les variables soient instanciées. Dès lors, le joueur $\exists$ gagne si le goal est satisfait, son adversaire remportant la partie dans le cas contraire.

Dans le cas général, il est possible que ce jeu se termine "prématurément", du fait que l'un ces CSP C_i devient inconsistant. Par exemple, si le préfixe P contient dès le début un rqset de la forme $(q_i, W_i, \bot)$, alors le gagnant sera toujours connu avant que l'on atteigne la fin du jeu : si on en vient à atteindre la profondeur i, le joueur correspondant n'a pas de mouvement valide (il est impossible d'affecter les variables de W_i demanière à satisfaire C_i), et perd donc instantanément la partie.

Ceci est un exemple (trivial) de ce que nous appelons un préfixe *bloquant* : dans au moins un scénario, l'un des joueur perd le jeu, non pas en fonction de la valeur de vérité du goal, mais parce qu'il a été empêché de jouer à un moment donné du jeu. Notons que dans le cas général, seul quelques scénarios sont bloquants dans un préfixe. Dans le cas contraire, on dit que le préfixe est *non-bloquant*, si et seulement si tous les scénarios possibles selon le préfixe atteignent le goal.

Formellement, cette propriété signifie qu'à tout niveau i du préfixe, (1) il existe une affectation A des variables de $W_0 \cup \ldots \cup W_{i-1}$ consistante avec l'ensemble des CSP $C_0 \cup \ldots \cup C_{i-1}$, et (2) pour toutes les affectations répondant à cette définition, il existe au moins une affectation A_i des variables de W_i telle que $A \cup A_i$ est une solution de C_i. Notons que cette propriété porte uniquement sur le *préfixe* du problème. Nous définissons donc cette propriété comme étant une propriété du préfixe d'un QCSP$^+$.

Définition 7 (préfixe non-bloquant).
Soit $P = [(q_0, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$ *un préfixe de longueur* n. P *est* non-bloquant *si et seulement si : (1) il s'agit du préfixe vide OU (2)* C_0 *admet au moins une solution, et, pour toute solution* s *de* C_0*, le préfixe* $P|_a$ *défini comme étant* $[(q_1, W_1, C_1[W_0 \leftarrow a]), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}[W_0 \leftarrow a])]$ *est lui-même non-bloquant.*

On étend cette propriété au QCSP lui-même en transposant celle de son préfixe :

Définition 8 (QCSP$^+$ non-bloquant). *Un QCSP*$^+$ $Q = (P, G)$ *est non-bloquant si et seulement si* P *l'est.*

Notons que les QCSP classiques sont des exemples triviaux de QCSP$^+$ non-bloquants.

Exemple : le jeu de Nim. Le jeu de Nim consiste en un tas d'allumettes duquel deux joueurs retirent tour à tour entre un et trois éléments. Le joueur qui vide le tas gagne la partie. On peut modéliser ce jeu en QCSP$^+$ de différentes manières, certaines donnant des QCSP$^+$ non-bloquants. Considérons par exemple un tas initial de 20 allumettes. Voici un premier modèle du jeu :

$$
\begin{array}{l}
\exists X_1 \in [1..3][] \\
\quad \forall Y_1 \in [1..3], S_1^{\forall} \in [0..20][S_1^{\forall} = X_1 + Y_1] \\
\quad\quad \exists X_2 \in [1..3], S_2^{\exists} \in [0..20][S_2^{\exists} = X_2 + S_1^{\forall} \wedge S_1^{\forall} \neq 20] \\
\quad\quad\quad \vdots \\
\quad\quad\quad \exists X_i \in [1..3], S_i^{\exists} \in [0..20][S_i^{\exists} = X_i + S_{i-1}^{\forall} \wedge S_{i-1}^{\forall} \neq 20] \\
\quad\quad\quad\quad \forall Y_{i+1} \in [1..3], S_{i+1}^{\forall} \in [0..20][S_{i+1}^{\forall} = Y_{i+1} + S_i^{\exists} \wedge S_i^{\exists} \neq 20] \\
\quad\quad\quad\quad\quad \vdots \\
\quad\quad\quad\quad\quad \top
\end{array}
$$

Dans ce modèle, les variables $S_i^{\exists}$ et $S_i^{\forall}$ représentent le nombre d'allumettes retirées depuis le début de la partie au tour i. La condition de victoire est directement modélisée dans les rqsets par les contraintes $S_i^{\exists} \neq 20$. En effet, on viole une telle contrainte à partir du moment où l'adversaire s'est emparé de la dernière allumette du tas. Ainsi, dans ce cas, on rend le rsqet suivant inconsistant, de sorte que l'adversaire perde la partie. Ce modèle est clairement bloquant, étant donné qu'un joueur est empêché de jouer dès que son adversaire s'est emparé de la dernière allumette. De fait, aucun scénario n'atteindra jamais le goal, étant donné que la dernière allumette sera toujours prise.

Proposons un modèle non-bloquant pour ce jeu. Dans celui-ci, nous explicitons les conditions de victoire du joueur $\exists$ directement dans le goal : la modélisation du décompte des allumettes reste dans les rqsets, mais ceux-ci ne bloqueront plus l'adversaire du joueur qui retire la dernière allumette. Le goal se voit en effet augmenté des contraintes assurant que le joueur $\exists$ a bien pris la dernière allumette :

$\exists X_1 \in [1..3][]$
$\forall Y_1 \in [1..3], S_1^\forall \in [0..60][S_1^\forall = X_1 + Y_1]$
$\exists X_2 \in [1..3], S_2^\exists \in [0..60][S_2^\exists = X_2 + S_1^\forall]$
$\vdots$
$\exists X_i \in [1..3], S_i^\exists \in [0..60][S_i^\exists = X_i + S_{i-1}^\forall]$
$\forall Y_{i+1} \in [1..3], S_{i+1}^\forall \in [0..60][S_{i+1}^\forall = Y_{i+1} + S_i^\exists]$
$\vdots$
$[S_2^\exists = 20 \vee S_3^\exists = 20 \vee S_4^\exists = 20 \vee \ldots$

Ici, les domaines des variables S restent assez grands afin de ne jamais pouvoir se vider (cette taille n'est cependant pas un problème, la valeur de ces variables étant fixées dès que les autres variables sont affectées). Le goal est vrai si et seulement si le nombre d'allumettes retirées par le joueur $\exists$ est égal au nombre initial d'allumettes présentes dans le tas en début de partie, c'est à dire si ce joueur vide le tas à un moment ou à un autre. Dans le cas ou son adversaire vide le tas, cette contrainte ne peut plus être satisfaite, et le goal devient inconsistant.

Ce modèle peut aisément être reconnu comme étnant non-bloquant : en effet, les CSP C_i se contentent de lier les valeurs des sommes partielles S aux nombres d'allumettes retirées à chaque tour X et Y et ne peuvent en aucun cas devenir totalement inconsistants.

3.2 Propagation dans les QCSP$^+$ non-bloquants

Le fait qu'un QCSP$^+$ soit non-bloquant a deux conséquences : d'une part, nous savons qu'une affectation des variables W_i qui rendrait inconsistant un CSP C_j situé plus loin dans le préfixe ne sera pas consistante non plus avec le CSP C_i. D'autre part, si à un moment donné, le goal devient inconsistant ou trivialement vrai, alors étant donné que tous les scénarios arrivent toujours au goal, le problème prend automatiquement la valeur de vérité du goal.

La formalisation de ces deux points donne lieu à une procédure de propagation additionnelle pour les QCSP$^+$ non bloquants : la première conduit à une fusion de tous les ensembles de contraintes présent dans les rqsets, et la seconde à une vérification du goal à chaque affectation de l'ensemble de variables d'un rqset.

Fusion des restricteurs. Du point de vue de la recherche de solution, le premier point évoqué nous permet de considérer n'importe quelle contrainte de n'importe quel rqset comme faisant aussi partie du restricteur courant. La propagation peut ainsi tirer parti de toutes ces contraintes lors de la recherche d'une affectation valide. Pour prouver la correction de cette technique, il nous faut montrer que toute valeur x d'une variable X de W_0 qui serait inconsistante dans un restricteur C_i l'est aussi dans C_0. Étant donné la définition inductive de la propriété d'être non-bloquant, cette démonstration s'étendra à toutes les variables du problème.

Théorème 9 (Fusion des restricteurs). *Soit $Q = (P, G)$ un QCSP$^+$ non-bloquant tel que $P = [(q_0, W_0, C_0), \ldots, (q_n, W_n, C_n)]$, et X une variable de son*

premier rqset ($X \in W_0$). Toute valeur x de D_X qui est inconsistante avec un des restricteurs C_i tel que $1 \leq i \leq n$ est inconistante avec C_0.

Preuve. Q étant non-bloquant, C_0 possède au moins une solution. Supposons qu'il existe une valeur x de D_X consistante avec C_0, mais pas avec un certain C_k. Soit S_0 une solution de C_0 affectant x à X.

Hypothèse d'nduction : Si un restricteur C_i avec $0 < i < k$ admet une solution S_k affetant x à X, et telle que $S_i|W_0 \in sol(C_0), \ldots, S_i|(W_0 \cup .. \cup W_{i-1} \in sol(C_{i-1})$, alors, comme P est non-bloquant, il existe donc une solution S_{i+1} de C_{i+1} telle que $S_{i+1}|(W_0 \cup .. \cup W_k) = S_i$ (et donc, où $X = x$).

Donc, par induction, C_k admet une solution S_k affectant la valeur x) X, ce qui contredit l'hypothèse selon laquelle x est inconsistante pour C_k. Par conséquent, si l'affectation $X \leftarrow x$ est inconsistante dans C_k, elle est aussi inconsistante dans C_0.

Vérification du goal. Comme nous l'avons vu plus haut, le fait qu'un QCSP$^+$ soit non-bloquant implique que si une inconsistance est détectée dans le goal (e.g. par propagation), alors le problème entier peut être évalué à $\bot$. De manière similaire, si le goal devient trivialement vrai avec les domaines de variables atuels, alors le QCSP$^+$ entier peut être directement évalué) $\top$:

Théorème 10 (Vérification du goal). *Soit $Q = (P, G)$ un QCSP$^+$ non bloquant, tel que $P =$ $[(q_0, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$. Si $sol(G) = \emptyset$, alors $[\![Q]\!] = \bot$, et si $sol(G) = D^{W_0 \cup \ldots \cup W_{n-1}}$, alors $[\![Q]\!] = \top$.*

Preuve. Si $sol(G) = \emptyset$, par définition, on a $[\![([], G)]\!] = \bot$. Considérons un préfixe non-bloquant $P' = [(q, W, C)|P'']$ et supposons que pour toute solution S de C, $[\![(]\!]P'', G[C \leftarrow S]) = \bot$ (

La déinition de la sémantique d'un QCSP$^+$ nous donne, selon que $q = \exists$ ou que $q = \forall$, soit $[\![(]\!]P', G) = \bigvee_{t \in sol(C_i)}([\![P''[W \leftarrow t], G[W \leftarrow t])]\!]$, soit $[\![(]\!]P', G) = \bigwedge_{t \in sol(C_i)}([\![(P''[W \leftarrow t], G[W \leftarrow t])]\!])$. Mais, P' étant lui-même un préfixe non)bloquant, $sol(C)$ n'est pas vide. Par l'hypothèse d'induction, on a donc soit $[\![(P', G)]\!] = \bigvee_{t \in sol(C_i)}(\bot)$ ou $[\![(P', G)]\!] = \bigwedge_{t \in sol(C_i)}(\bot)$ ce qui, dans les deux cas, nous mène à $[\![(P', G)]\!] = \bot$.

La démonstration de la seconde partie est analogue.

3.3 QCSP$^+$semi-bloquants

La procédure de résolution d'un QCSP$^+$ passe le plus clair de son temps à rechercher une solution pour un rqset. Cette recherche de solution peut bénéficier de la technique de fusion des restricteurs, mais la vérification du goal ne peut se faire que lorsque l'ensemble des variables d'un qset vient d'être affecté. En effet, il est impossible, en plein milieu de la résolution d'un restricteur, de savoir si l'état de recherche courant va aboutir à une solution ou non. On ne peut

donc pas utiliser la vérification du goal en prenant en compte les réductions de domaines faites sur ce restricteur, étant donné que rien ne dit si ces réductions vont effectivement aboutir à une solution ou non

Considérons le cas d'un QCSP$^+$ $Q = ([(q_0, W_0, C_0)|P'], G)$. Si ce problème à tout point de la recherche de solution du premier restricteur C_0, il n'y a que deux possibilités :

- soit l'affectation partielle de W_0effectuée par la recherche de cette solution ne rend pas ce dernier inconsistant, et donc Q est toujours non-bloquant ;
- soit C_0 est devenu inconsistant (mais on ne l'a pas encore détecté), et la prise en compte de l'affectation partielle des variables de W_0 rend donc Q vrai si $Q_0 = \forall$, ou faux si $Q_0 = \exists$.

Dans les deux cas, notons que si $q_0 = \exists$ et $G = \bot$ (resp. $q_0 = \forall$ et $G = \top$), on a $[\![Q]\!] = \bot$ (resp.$[\![Q]\!] = \top$).

Formalisons ce cas spécial :

Définition 11 (Préfixe semi-bloquant).
soit $P = [(q_0, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$ *un préfixe de longueur* n. P *est* $\exists$-semi-bloquant *(resp.* $\forall$-semi-bloquant *si, et seulement si : (1)* $q_0 = \exists$ *(resp.* $q_0 = \forall$*) et (2) pour toute solution* S *de* C_0*, le préfixe* $P|_S = [(q_1, W_1, C_1[W_0 \leftarrow S]), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}[W_0 \leftarrow S])]$ *est non-bloquant.*

Définition 12 (QCSP$^+$ semi-bloquant).
Un QCSP$^+$ $Q = (P, G)$ *est* $\exists$-semi-bloquant *(resp.* $\forall$-semi-bloquant*) si, et seulement si* P *is* $\exists$-semi-bloquant *(resp.* $\forall$-semi-bloquant*).*

La différence entre cette définition et la définition de préfixe / QCSP$^+$ non-bloquant est que l'existence d'une solution pour C_0 n'est plus requise. Cependant, Si il en existe, alors le problème est bien non-bloquant. Dans chacun de ces cas, on peut appliquer une "moitié" du théorème de vérification du goal :

Théorème 13 ($\exists$-semi vérification du goal).
Soit $Q = (P, G)$ *un QCSP*$^+$ $\exists$*-semi-bloquant, avec* $P = [(\exists, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$. *Si* $sol(G) = \emptyset$*, alors* $[\![Q]\!] = \bot$.

Preuve. Si $sol(C_0) = \emptyset$, alors par définition de la sémantique des QCSP$^+$, $[\![Q]\!] = \bot$. Sinon, Q est non-bloquant et le théorème de vérification du goal s'applique.

Théorème 14 ($\forall$-semi vérification du goal).
Soit $Q = (P, G)$ *un QCSP*$^+$ $\forall$*-semi-bloquant, tel que* $P = [(\forall, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1}]$. *Si* $sol(G) = \prod(D(W_0) \ldots D(W_{n-1}))$*, alors* $[\![Q]\!] = \top$.

Preuve. Si $sol(C_0) = \emptyset$, $[\![Q]\!] = \top$. Sinon, Q est non-bloquant et le théorème de vérification du goal s'applique également.

4 Discussion

Une propriété à indiquer a-priori. Intuitvement, un QCSP$^+$ est non-bloquant si, et seulement si on ne rencontre jamais pendant la résolution de restricteur sans solution. Du point de vue du jeu opposant le joueur-$\exists$ au joueur-$\forall$, tout joueur a au moins un coup valide à jouer jusqu'à la fin de la partie. En d'autres termes, il existe un mouvement possible pour le joueur actuel, et pour tous ces mouvements possibles, son adversaire aura un mouvement possible, et pour tous ses mouvements, etc.

Donc, décider si un préfixe $P = [(q_0, W_0, C_0), \ldots, (q_{n-1}, W_{n-1}, C_{n-1})]$ est non-bloquantrevient à décider la valeur de vérité de la formule suivante :

$$\exists w_0 \in D_{W_0} C_0 \wedge \forall w'_0 \in D_{W_0} C_0 \rightarrow \exists w_1 \in D_{W_1} C_1 \wedge \forall w'_1 \in D_{W_1} \ldots$$

ce qui est équivalent à résoudre le QCSP$^+$ suivant :
$\exists W_0[C_0(W_0)]$
$\forall W'_0[C_0(W'_0)]$
$\exists W_1[C_1(W'_0, W_1]$
$\forall W'_1[C'_1(W'_0, W'_1)]$
$\ddots$
$\top$

Décider si un QCSP$^+$ donné est non-bloquant est donc en général aussi complexe que de le résoudre. Ainsi, utiliser les techniques présentées dans ce papier n'a de sens que si le modeleur fournit un QCSP$^+$ construit de manière à être non-bloquant et le déclare comme tel.

Cependant, on pourrait détecter certaines instances comme "aisément non-bloquantes". Par exemple, on peut imaginer un QCSP$^+$ où les restricteurs portent sur des ensembles de variables disjoints deux à deux. Dans ce cas, il suffit, pour assurer qu'un tel QCSP$^+$ est non-bloquant, de vérifier que tous les restricteurs ont au moins une solution.

Planification avec adversaire "non-bloquante". Les QCSP$^+$ non-bloquants ne se limitent pas à la modélisation de jeux. [3] présente un petit modèle de planification avec adversaire, où un ordonnancement de tâches doit rester faisable quoi que puisse faire un ennemi. Ce modèle est du type :
$\exists(S)[Faisable(S)]$
$\forall(A)[Possible(A, S)]$
$Faisable(A(S))$
où S est un ordonnancement (*schedule*), A la modélisation d'une attaque de l'ennemi, $A(S)$ les données de l'ordonnancement après cette attaque, $Faisable$ un ensemble de contraintes assurant qu'un ordonnancement est faisable, et $Possible$ un ensemble de contraintes déterminant si une attaque est dans les possibilités de l'ennemi ou non.

Dans ces problèmes, il est toujours possible de trouver un ordonnancement réalisable avant passage de l'ennemi, et celui-ci peut toujours déclencher une attaque. Ces problèmes sont donc non-bloquants, et les techniques présentées ici peuvent donc s'appliquer.

Problèmes partiellement non-bloquants. Beaucoup de problèmes, notamment des jeux, sont naturellement modélisés de manière à ce que les premiers restricteurs aient toujours des solutions. Par exemple, considérons le premier modèle du jeu de Nim présenté en section 3. Ce modèle est clairement bloquant. Cependant, étant donné que chaque joueur peut prendre au maximum trois allumettes, la vingtième et dernière allumette ne peut pas être prise avant le septième tour. On peut imagine un extension de la propriété des QCSP$^+$ non-bloquants qui définirait ce modèle comme *non-bloquant jusqu'au* 7^{eme} *rqset*, ce qui permettrait de fusionner les sept premiers restricteurs.

Modélisation. Dans [4], Peter Nightingale présente des modèles en QCSP standatrd pour le jeu du morpion et celui du Puissance-4. Dans ces modèles, il créé des "clones" existentiels des variables universelles, et pose des contraintes *shadow* assurant qu'une variable et son clone doivent être égales si le coup joué est valide. Dans le cas contraire (i.e. si le joueur universel "triche"), le clone existentiel peut être fixé à n'importe quelle valeur valide, ce qui correspond intuitivement au fait que le joueur existentiel joue à la place de son adversaire, et peut donc effectuer un mouvement qui soit en sa faveur. Cependant, cette technique ne peut pas s'appliquer dans les cas où le joueur-$\forall$ n'a plus aucun coup valide à jouer. en effet, dans ce cas, le clone existentiel de la variable universelle voit aussi son domaine se vider par les contraintes modélisant les règles du jeu. Le QCSP s'évalue donc dans ce cas à $\bot$ alors qu'il devrait au contraire être vrai. Ainsi, cette technique de modélisation impose que le joueur-$\forall$ aie toujours un coup valide à jouer, i.e. qu'il soit soumis à des règles non-bloquantes.

On remarque donc une certaine corrélation entre les problèmes pouvant naturellement se modéliser par un QCSP$^+$ non-bloquant sont ceux qui peuvent se modéliser aisément en QCSP standard grâce à cette technique présentée par Nightingale. Cependant il existe aussi, comme pour le jeu de Nim présenté plus haut, des modèles bloquants pour ces problèmes. Il reste donc à étudier les différents couples (modèle,technique de résolution) pour différents problèmes, afin de comparer ces approche entre elles sur une gamme de problèmes large.

5 Conclusion

Ce papier a exhibé le cas spécial des QCSP$^+$ non-bloquants, où des techniques de résoution additionnelles peuvent être appliquées. Ce cas spécial correspond naturellement à toute une classe de jeux où aucun joueur ne peut être empêché de jouer, de même qu'a d'autres problèmes moins ludiques, comme certains problèmes d'ordonnancement avec adversaire. Savoir a-priori qu'un QCSP$^+$ est non-bloquant (e.g. par construction) est susceptible d'accélérer sensiblement sa résolution.

Les travaux présentés dans ce papier sont préliminaires à une étude plus large des problèmes quantifiés pouvant aisément se modéliser de manière non-bloquante, afin de situer les différentes techniques de modélisation les unes par rapport aux autres.

References

1. Benedetti, M., Lallouet, A., Vautard, J.: QCSP Made Practical by Virtue of Restricted Quantification. In Veloso, M., ed.: International Joint Conference on Artificial Intelligence, Hyderabad, India, AAAI Press (2007) 38–43
2. Bordeaux, L., Cadoli, M., Mancini, T.: CSP properties for quantified constraints: Definitions and complexity. In Veloso, M.M., Kambhampati, S., eds.: National Conference on Artificial Intelligence, AAAI Press (2005) 360–365
3. Benedetti, M., Lallouet, A., Vautard, J.: Modeling adversary scheduling with QCSP+. In: ACM Symposium on Applied Computing, Fortaleza, Brazil, ACM Press (2008)
4. Nightingale, P.: Consistency and the Quantified Constraint Satisfaction Problem. PhD thesis, University of St Andrews (2007)

Traitement d'images

Recalage hybride des images médicales basé sur l'information mutuelle et l'ICP accéléré

Leila Benaissa Kaddar, Nacéra Benamrane

Département d'Informatique, Faculté des Siences, USTOMB
B.P 1505, EL'Mnaouer 31000, Oran, Algérie
nabenamrane@ yahoo.com, Bkleila_usto@yahoo.fr

Abstract. Le recalage d'images trouve de nombreuses applications médicales aussi bien dans le suivi thérapeutique que dans le diagnostic d'un patient. En s'inspirant des méthodes proposées dans la littérature nous proposons dans ce papier, une méthode de recalage d images médicales basée sur une hybridation entre deux techniques géométrique et iconique afin de diminuer le temps de calcul et au même temps d'améliorer la qualité visuelle de l'image recalée en utilisant l'information mutuelle et l'ICP accéléré. Cette approche a été testée sur une panoplie d'images IRM et les résultats obtenus sont encourageants.

Keywords: Recalage iconique, recalage géométrique, information mutuelle, ICP accéléré, imagerie médicale.

1 Introduction

Avant d'étudier deux images (ou deux volumes) correspondant au même objet physique, il faut qu'il existe une concordance de position spatiale entre les deux images (ou volumes). Ceci est réalisé par une opération de recalage.

Le recalage consiste à mettre en correspondance, avec une grande précision, deux images d'un même patient qui n'ont pas été acquises simultanément. En effet, même si deux acquisitions d'images sont successives, le patient bouge de quelques millimètres dans une ou plusieurs des trois dimensions de l'espace. De plus, les mouvements physiologiques du patient sont responsables d'un décalage entre le plan programmé et les images obtenues lors de séquences successives. Cela rend difficile, voire impossible, de savoir si la variation de la taille d'une lésion est le fait d'une modification biologique de la lésion ou au contraire d'un changement de la position du patient entre les deux examens.

Le recalage s'avère indispensable pour la comparaison avec une image source normale (atlas), pour la comparaison par rapport au malade lui-même soit au cours d'une même séance d'acquisition soit par rapport à un examen précédent (notamment pour le suivi temporel). Le choix des attributs utilisés pour guider le recalage est crucial. Il est largement conditionné par la nature des images à traiter. Dés lors, quatre critères caractérisent une méthode de recalage : les attributs, le critère de similarité, le modèle de déformation et la stratégie d'optimisation [1]

En classifiant de manière simple les différentes méthodes de recalage d'images, on distingue deux grandes catégories. Les approches géométriques basées sur l'extraction

de primitives géométriques dans l'image et les approches iconiques basées sur la comparaison des valeurs d'intensité des voxels dans l'image. Par ailleurs, des méthodes hybrides [2] combinant ces différentes approches ont aussi été proposées.

Les méthodes hybrides, ce sont des méthodes qui reposent sur la combinaison de plusieurs types d'information différents. L'idée est d'améliorer la robustesse de l'algorithme de recalage en combinant les avantages liés à chaque type d'information utilisé. Trois cas peuvent être distingués : la combinaison de primitives géométriques de natures différentes, la combinaison de différentes informations issues des niveaux de gris et la combinaison des approches géométriques et iconiques.

Concernant la combinaison de primitives géométriques de natures différentes, on pourra se référer aux travaux de Yuille et al [3] pour la combinaison de points et de courbes, ceux de Loew et al [4] pour la combinaison de courbes et de surfaces et aux travaux de Maurer [5] pour la combinaison de différents types de surfaces. Concernant les exemples de combinaison de différentes informations iconiques, on peut citer les travaux de Pluim et al [6]. qui utilisent à la fois le gradient de l'image et l'information des niveaux de gris ainsi que ceux de Shen et al[7] qui proposent d'associer à chaque voxel un vecteur d'attributs composé de l'intensité du voxel en question, de différents moments géométriques invariants caractéristiques du voisinage du voxel et d'une information issue de la segmentation en trois classes de l'image (soit les probabilités d'appartenance à chacune des classes dans le cas d'une segmentation floue, soit une étiquette associée au type de frontière entre classes dans le cas d'une segmentation dure).

Enfin, concernant l'utilisation de certaines primitives géométriques pour contraindre des méthodes iconiques, on peut citer Sorzano & al [8] pour la contrainte par des amers ponctuels, et Cahier & al [9] pour la contrainte par des primitives courbes (sillons corticaux). Liu et al [10] proposent par ailleurs une méthode hybride de recalage volumique et surfacique pour la mise en correspondance des zones corticales.

Dans cet article, nous proposant une approche hybride basée sur l'information mutuelle et l'ICP accéléré. Le reste de cet article est organisé comme suit : Dans la section 2, nous exposons notre approche avec plus de détails. La section 3, présente les résultats expérimentaux obtenus. Une conclusion résumera notre contribution.

2. Approche proposée

Pour pallier aux différents inconvénients des techniques géométriques et iconiques telle que la qualité de l'image recalée qui est dégradée dans la méthode géométrique et le coût de calcul qui est considérable, ce qui est dû au fait que chacun des pixels ou voxels de l'image est pris en considération dans la méthode iconique[11], nous proposons une approche de recalage permettant d'hybrider les deux méthodes géométrique et iconique. Notre approche opère en deux passages, le passage global utilise l'ICP accéléré et le local l'information mutuelle comme mesure de similarité et la descente du gradient comme méthode d'optimisation. La figure 1 illustre les différentes étapes de notre approche.

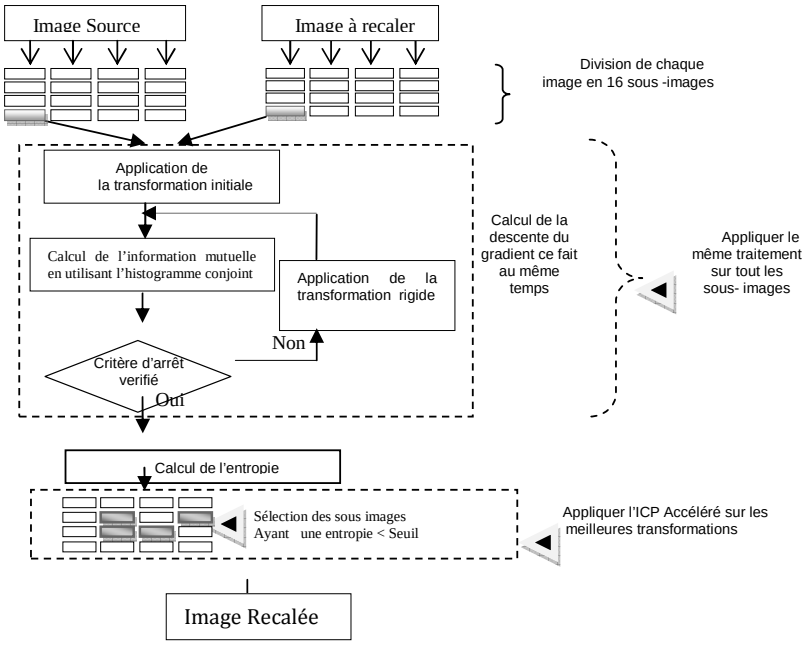


Fig. 1 : Schéma de l'approche proposée

2.1. Transformation rigide

Les transformations rigides sur une image sont des transformations qui s'effectuent sur toutes les coordonnées d'une image en suivant la même modalité. Nous pouvons ainsi translater une image et lui faire effectuer des rotations.

Les coordonnées de chaque point subiront des translations et des rotations suivant une matrice de transformation :

Soit P les coordonnées d'un point d'une image 2d. $P=(x\ y)$

Soit P' les coordonnées du point équivalent dans l'image modifiée. P'= (x' y')

M la matrice de transformation 3x3.

Alors :

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = M \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad \textbf{(1)}$$

La forme d'une une matrice de transformation pour une translation est:

$$M = \begin{pmatrix} 1 & 0 & x_{trans} \\ 0 & 1 & y_{trans} \\ 0 & 0 & 1 \end{pmatrix} \quad \textbf{(2)}$$

Pour une rotation de θ radians autour de l'axe des x, la matrice est la suivante:

$$M = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \textbf{(3)}$$

Une fois les coordonnées de tous les points de l'image modifiée, nous nous retrouvons avec une image ayant des coordonnées non entières. Il est donc nécessaire de la rééchantilloner. Ceci consiste à créer une nouvelle image ayant des coordonnées entières se rapprochant le plus possible de l'image ayant des coordonnées réelles. Dans notre implémentation, nous avons utilisé comme méthode de ré-échantillonnage l'interpolation bilinéaire. C'est une méthode simple pour éliminer le phénomène d'aliasing. Les 4 points les plus proches des coordonnées calculées dans l'image source sont utilisés en les pondérant par des coefficients inversement proportionnels à la distance et dont la somme vaut 1: Le poids affecté à chaque point est:

$$p(x, y) = \frac{1}{\sqrt{(x - ix)^2 + (y - iy)^2}} \quad \textbf{(4)}$$

La valeur du point obtenu par interpolation sur les 4 points les plus proches est:

$$v(i,j) = \frac{p(ix,iy).v(ix,iy) + p(ix+1,iy).v(ix+1,iy) + p(ix,iy+1).v(ix,iy+1) + p(ix+1,iy+1).v(ix+1,iy+1)}{p(ix,iy) + p(ix+1,iy) + p(ix,iy+1) + p(ix+1,iy+1)} \quad \textbf{(5)}$$

2.2 Maximisation de l'information mutuelle

L'information mutuelle est la quantité d'information d'une image contenue dans une seconde image. Ainsi, lorsque l'information mutuelle entre deux images est au maximum, elles sont identiques. Elle permet de transformer une image pour qu'elle ressemble le plus à une image donnée en maximisant l'information mutuelle des deux images concernées.

Le calcul d'information mutuelle de deux images est basé sur la densité conjointe de probabilité des niveaux de gris des images. Il est nécessaire pour estimer la densité conjointe de probabilité de calculer un histogramme conjoint des niveaux de gris entre ces deux images.

Concrètement, l'histogramme conjoint est un graphe tridimensionnel. Chaque point de l'histogramme représente le nombre de fois qu'une combinaison de niveau de gris entre les deux images est rencontrée.

Soit $g(x, y)$ la valeur de l'histogramme conjoint au point $[x, y]$.
On pose :

$$p_{1,2}(x,y) = \frac{g(x,y)}{\sum_{a,b} g(a,b)} \quad \textbf{(6)}$$

$$p_1(x) = \sum_b p_{1,2}(x,b)$$

$$p_2(y) = \sum_b p_{1,2}(a,y)$$

On a ainsi :

$$MI = \sum_{a,b} p_{1,2}(a,b) \log_2 \frac{p_{1,2}(a,b)}{p_1(a).p_2(b)} \quad \textbf{(7)}$$

Où :

1. $\sum_{a,b} g(a,b)$ représente le nombre de points utilisés pour créer l'histogramme conjoint.

2. $p_{1,2}$ est l'histogramme conjoint normalisé. En effet la somme de ces valeurs vaut 1. C'est ainsi une distribution de probabilité. $p_{1,2}(x,y)$ peut donc se lire comme la probabilité qu'un point pris au hasard dans l'image *A* soit la combinaison du niveau de gris *x* sur l'image *A* et du niveau de gris y sur l'image *B*.

3. $p_1(x)$ est aussi une distribution de probabilités. Pour un *x* donné c'est la probabilité que l'on trouve un point de niveau de gris *x* sur l'image *A*.

4. $p_2(y)$ est comme $p_1(x)$. Pour un y donné c'est la probabilité que l'on trouve un point de niveau de gris y sur l'image *B*.

2.3 La méthode d'optimisation (descente de gradient)

Il n'existe pas d'algorithme universel efficace pour minimiser toutes les fonctions. Pour notre approche nous avons opté pour la descente de gradient comme méthode d'optimisation. Le gradient d'une fonction de R^n dans *R* est un vecteur de dimension *n*. Chacun de ses termes contient la dérivée de la fonction par rapport à une variable. Le gradient a la propriété de donner la direction dans laquelle une fonction continue et dérivable varie le plus. La méthode du gradient consiste donc à utiliser la direction donnée par le gradient comme direction dans laquelle des tests vont être fait pour trouver un point où l'information mutuelle est maximale.

Le calcul du gradient se fait comme suit, On calcule tout d'abord la dérivée par rapport à chaque variable.

Soient *PTC* = (*decx,decy,,rot)* le point courant et *eps* une petite valeur, on note *MI*(*x*) l'information mutuelle au point *x*.

La dérivée de l'information mutuelle en fonction du décalage en *x* sera estimée par :

$$der_{dec_x} = \frac{MI(PT_c + (eps,0,0)) - MI(PT_c - (eps,0,0))}{eps + eps} \qquad \textbf{(8)}$$

La dérivée de l'information mutuelle en fonction du décalage en y sera estimée par

$$der_{dec_y} = \frac{MI(PT_c + (0,eps,0)) - MI(PT_c - (0,eps,0))}{eps + eps} \qquad \textbf{(9)}$$

La dérivée de l'information mutuelle en fonction de la rotation autour de *x* sera estimée par :

$$der_{rot} = \frac{MI(PT_c + (0,0,esp)) - MI(PT_c - (0,0,eps))}{eps + eps} \qquad \textbf{(10)}$$

Ensuite le gradient est rempli :

$$grad\ (MI\ (PT_c)) = \begin{pmatrix} der_{dec_x} \\ der_{dec_y} \\ der_{rot} \end{pmatrix} \qquad \textbf{(11)}$$

Une fois la direction trouvée, le maximum dans la direction du gradient est recherché.

2.4 Entropie

Après l'obtention des meilleures transformations locales de chaque sous-image, l'entropie de chaque sous-image est calculée sachant que l'entropie est la quantité d'information contenue dans une série d'événements. Nous avons opté pour la définition de l'entropie [12] suivante :

$$H = \sum_i p_i \log \frac{1}{p_i} = -\sum_i p_i \log p_i \qquad \textbf{(12)}$$

Après le calcul de l'entropie, un seuil S est fixé afin de choisir que les sous-images ayant une quantité d'information importante, ce qui permet de rendre le recalage plus rapide et plus robuste. En fait, toutes les sous-images de l'image *A* ne sont pas utilisées pour construire les appariements. Les sous-images sont en fait triées selon leur valeur de l'entropie et on ne considère que les sous-images ayant une entropie < S. En effet, les sous-images avec une forte entropie sont susceptibles d'être dans une zone homogène de l'image *A.* Elles pourront s'apparier avec n'importe quelle région homogène de l'image *B*, et les appariements ainsi construits ne pourront que gêner le calcul de la transformation.

2.5 ICP accéléré

L'ICP accéléré est appliqué sur les meilleures transformations locales des sous-images sélectionnée précédemment. Rappelons tout d'abord l'algorithme ICP.

Soient deux ensembles de points *P* et *X*. L'algorithme ICP produit une transformation rigide optimale $\vec{q}$ pour un minimal local $X_{k,}$.

Notre implémentation est basée sur les quaternions qui est seulement utilisé en 2D et 3D. Ce pendant l'algorithme de décomposition en valeur singulière SVD peut être utilisé afin de généraliser la méthode à N dimension [13][14].

Le centre de masse $\vec{\mu}_p$ pour l'ensemble de point mesuré *P* et le centre de masse $\vec{\mu}_x$ pour l'ensemble de point *X* est donné comme suit :

$$\vec{\mu}_p = \frac{1}{N_P} \sum_{i=1}^{N_P} \vec{P}_i \quad et \quad \vec{\mu}_x = \frac{1}{Nx} \sum_{i=1}^{N_P} \vec{x}_i \qquad \textbf{(13)}$$

La matrice de covariance de l'ensemble de *P* et *X* est :

$$\sum_{px} = \frac{1}{N_P} \sum_{i=1}^{N_P} [(\vec{p}_i - \vec{\mu}_p)(\vec{x}_i - \vec{\mu}_p)^t] = \frac{1}{N_P} \sum_{i=1}^{N_P} [\vec{p}_i \vec{x}_i^{\,t}] - \vec{\mu}_p \vec{\mu}_x^{\,t} \qquad \textbf{(14)}$$

Les composants cycliques de la matrice antisymétrique $A_{ij} = (\sum_{px} - \sum_{px}^{T})_{ij}$ sont utilisés pour former un vecteur colonne $\Delta = [A_{23} A_{31} A_{12}]^T$. Ce vecteur est ainsi utilisé pour former la matrice $Q(\sum_{px})$ 4x4 symétrique.

$$Q(\sum_{px}) = \begin{bmatrix} tr(\sum_{px}) & \Delta^T \\ \Delta & \sum_{px} + \sum_{px}^{T} - tr(\sum_{px}) I_3 \end{bmatrix} \qquad \textbf{(15)}$$

Où I_3 est une matrice identité 3x3. Le vecteur unitaire propre de la matrice $\vec{q}_r$ correspondante au maximum des valeurs propres de la matrice $Q(\sum_{px})$ est sélectionné comme la rotation optimale. Le Jacobi peut être utilisé à la matrice $Q(\sum_{px})$ afin d'obtenir le vecteur propre [15][16].

Le vecteur de transformation optimale est donné par :

$$\vec{q}_T = \vec{\mu}_x - R(\vec{a}_R)\vec{\mu}_p \quad \textbf{(16)}$$

Le vecteur d'espace 7 $\vec{q}_k$ correspondant à la transformation rigide est construit comme la concaténation de $\vec{q}_R$ et $\vec{q}_T$ qui est : $\vec{q}_k = \left[\vec{q}_R \middle| \vec{q}_T \right]^t$ **(17)**

Initialement $\vec{q}_k$ est un vecteur identité et P_k=P.

Au début de chaque itération, un nuage X_k, qui est une projection de P_k sur X, est construit. Ensuite le vecteur courant $\vec{q}_k$ est calculé en fonction de P et X. Enfin $\vec{q}_k$ est appliqué à P. La convergence est atteinte lorsque l'erreur quadratique soit en dessous de la valeur prédéfinit t.

L'algorithme :

1. Calculer la projection X_k = CP (P_k, X)
2. Calculer le recalage $\vec{q}_k$ =Q (P_0, X_k)
3. Appliquer le recalage $P_{k+1} = \vec{q}_k$ (P_0)
4. Répéter jusqu'à la convergence : MSE ($\vec{q}_k$ (P), X<t

L'ICP accéléré est originalement proposé par Besl et McKay [17], l'idée est de prédire l'évolution de $\vec{q}_k$, la transformation rigide.

Durant l'exécution de l'ICP une séquence des vecteurs de recalage est généré : $\vec{q}_0$, $\vec{q}_1$, $\vec{q}_2$, $\vec{q}_3$, $\vec{q}_4$, ….., qui tracent le chemin dans la forme de l'espace des états du recalage à partir de la transformation identité jusqu'à la correspondance d'une forme optimal locale.

Considérant la séquence de vecteur de différence défini par :

$$\Delta \vec{q}_k = \vec{q}_k - \vec{q}_{k-1} \quad \textbf{(18)}$$

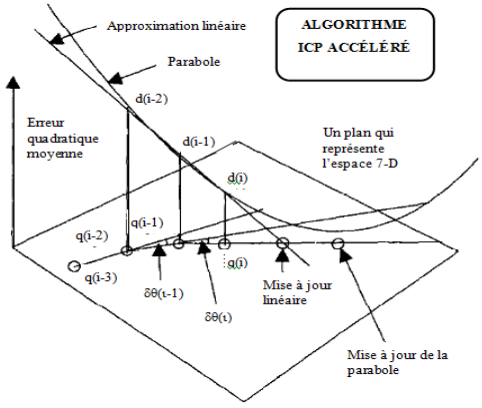


Fig.2 Variation de q_k dans l'espace d'état du recalage

Soit l'angle dans l'espace 7 entre les deux dernières directions :

$$\theta_k = \cos^{-1} \frac{\Delta \vec{q}_k^{\,t} \, \Delta \vec{q}_{k-1}^{\,t}}{\left\| \vec{q}_k \right\| \left\| \vec{q}_{k-1} \right\|} \qquad \textbf{(19)}$$

Soit $\delta\theta$ une tolérance angulaire suffisamment petite :

$$\text{Si } \theta_k < \delta\theta \quad \text{et} \quad \theta_{k-1} < \delta\theta$$

Alors, il y a un bon alignement de direction pour les trois derniers vecteurs d'état de recalage $\vec{q}_k$, $\vec{q}_{k-1}$, et $\vec{q}_{k-2}$.

Soient d_k, d_{k-1} et d_{k-2} les erreurs quadratiques moyennes associées, soient v_k, v_{k-1} et v_{k-2} une approximation des valeurs d'argument de la longueur d'arcs associés :

$$\begin{aligned} v_k &= 0, \\ v_{k-1} &= -\left\| \Delta \vec{q}_k \right\|, \\ v_{k-2} &= -\left\| \Delta \vec{q}_{k-1} \right\| + v_{k-1} \end{aligned} \qquad \textbf{(20)}$$

Après, une approximation linéaire et une interpolation parabolique, les trois derniers points de repères sont calculés :

$$\begin{aligned} d_1(v) &= a_1 v + b_1 \\ d_2(v) &= a_2 v^2 + b_2 v + c_2 \end{aligned} \qquad \textbf{(21)}$$

Ce qui nous donne une modification linéaire possible basée sur le passage par zéro de la ligne et une modification de parabole possible basée sur le point extrême de la parabole.

$$v_1 = -b_1 / a_1 > 0, v_2 = -b_2 / 2a_2 \qquad \textbf{(22)}$$

Nous avons utilisé une valeur maximale possible v_{max} . en fonction des modifications de $\vec{q}_k$.

1. Si $0 < v_2 < v_1 < v_{max}$ ou $0 < v_2 < v_{max} < v_1$, on utilise la parabole basé sur le vecteur de recalage modifié $\vec{q}_k' = \vec{q}_k + v_2 \Delta \vec{q}_k / \left\| \Delta \vec{q}_k \right\|$ à la palace du vecteur usuel $\vec{q}_k$ lorsqu' on effectue la mise à jour sur l'ensemble de points ce qui veut dire $C_{k+1} = \vec{q}_k'(C_0)$.

2. Si $0 < v_1 < v_2 < v_{max}$ ou $0 < v_1 < v_{max} < v_2$ ou $v_2 < 0$ et $0 < v_1 < v_{max}$, on utilise la ligne basée sur le vecteur de recalage modifié $\vec{q}_k' = \vec{q}_k + v_1 \Delta \vec{q}_k / \left\| \Delta \vec{q}_k \right\|$ à la place du vecteur usuel $\vec{q}_k$.

3. Si $v_1 > v_{max}$ et $v_2 > v_{max}$, on utilise un maximum de modification autorisé $\vec{q}_k' = \vec{q}_k + v_{\max} \Delta \vec{q}_k / \left\| \Delta \vec{q}_k \right\|$ à la place du vecteur usuel $\vec{q}_k$

3. Résultats expérimentaux

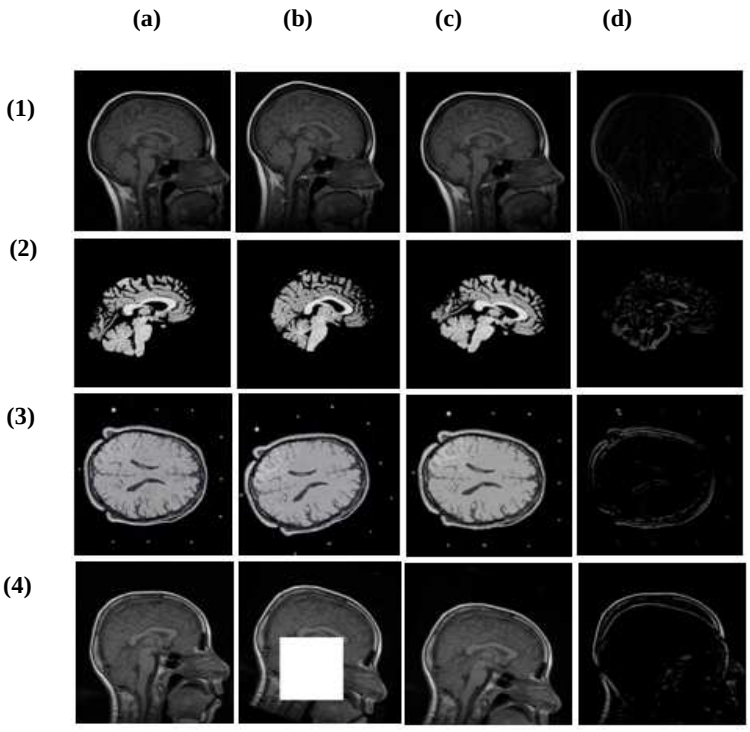
Notre méthode a été testée sur des images IRM de taille 256 × 256. Le seuil S concernant l'entropie est fixé à 0.3 et le test d'arrêt (valeur de la norme de différence entre l'image résultat et l'image source) doit être inférieur à 0.3. Le temps de calcul global est aux environs de 1 seconde. Le tableau 1 montre quelques caractéristiques sur les bases d'images et pour ces bases le temps de calcul et l'information mutuelle obtenus.

La figure 3 montre les résultats obtenus, les exemples (1) et (2) c'est dans le cas où nous avons deux prises successives du même patient, les résultats du recalage sont encourageants vu que l'image de différence montre une légère différence entre l'image source et l'image résultat.

Notre méthode traite aussi le cas d'une image segmentée manuellement par un expert comme étant une image source et une autre image non segmentée comme étant une image cible, c'est ce qui est illustré dans l'exemple (3). Le dernier exemple (4), montre le cas d'une image ayant des données manquantes qu'on a pu la compléter.

Table 1. Résultats obtenus par notre approche.

N° de la base de données	*Vecteur de la transformation (Trx, Try, Rot)*	*Information mutuelle*	*Temps d'exécution(s)*
(1)	**(4, 10, 0.2)**	**0.9073**	**0.639**
(2)	**(20, 5,-0.4)**	**0.0578**	**0.661**
(3)	**(8, 0, -0.3)**	**0.3252**	**0.696**
(4)	**(5, 22, -0.28)**	**0.3911**	**0.694**



(a) image source, (b) image cible (c) image résultat et (d) l'image de différence

Fig.3 Exemples des images recalées

4. Conclusion

Dans ce papier, nous proposons une approche hybride de recalage basée sur les deux techniques géométrique et iconique. Cette approche opère en deux passages, le premier est local en utilisant l'information mutuelle et le deuxième passage est global en utilisant la dernière extension de l'algorithme ICP, qui est l'ICP accéléré.

Les résultats obtenus sont encourageants et ont permis de diminuer le temps de calcul et au même temps d'améliorer la qualité visuelle de l'image recalée. L'intégration d'autres critères de similarité et l'utilisation d'un schéma multi-echelle peuvent améliorer les résultats obtenus.

Références

1. L.G. Brown. A survey of image registration techniques. ACM Computing Surveys, vol. 24, no. 4, pages 325-376, décembre 1992.
2. C. Barillot. Fusion de données et imagerie 3D en médecine. Habilitation à diriger des recherches, Université de Rennes I, 1999.
3. A.L. Yuille, P.W. Hallinan & D.S. Cohen. Feature Extraction from Faces Using Deformable Templates. International Journal of Computer Vision, vol. 8, no. 2, pages 99-111, 1992
4. L. Hsu, M. H. Loew & J. Ostuni. Automated Registration of Brain Images Using Edge and Surface Features. IEEE Engineering in Medicine and Biology Magazine, vol. 18, no. 6, pages 40-47, November 1999.
5. C.R. Jr Maurer, R.J. Maciunas & J.M. Fitzpatrick. Automated Registration of Brain Images Using Edge and Surface Features. IEEE Transactions on Medical Imaging, vol. 17, no. 5, pages 753-761, october 1998.
6. J.P.W. Pluim, J.B.A. Maintz & M.A. Viergever. Image registration by maximization of combined mutual information and gradient information. IEEE Transactions on Medical Imaging, vol. 19, no. 8, pages 809-814, août 2000.
7. D. Shen & C. Davatzikos. HAMMER : Hierarchical Attribute Matching Mechanism for Elastic Registration. IEEE Transactions on Medical Imaging, vol. 21, no. 11, pages 1421-1439, november 2002.
8. C. O.S. Sorzano, P. Thévenaz & M. Unser. Elastic Registration of Biological Images Using Vector-Spline Regularization. IEEE Transactions on Biomedical Engineering, vol. 52, no. 4, pages 652-663, April 2005.
9. P. Cachier, E. Bardinet, D. Dormont, X. Pennec & N. Ayache. Iconic feature based non rigid registration : the PASHA algorithm. Computer Vision and Image Understanding, vol. 89, no. 2-3, pages 272-298, 2003.
10. T. Liu, D. Shen & C. Davatzikos. Deformable registration of cortical structures via hybrid volumetric and surface warping. NeuroImage, vol. 22, no. 4, pages 1790-1801, 2004.
11. Vincent NOBLET, « Recalage non rigide d'images cérébrales 3D avec contrainte de conservation de la topologie », thèse doctorat, pages 29-30, 2006.
12. C. Shannon, "A mathematical theory of communication," technical report, Bell System.
13. Arun, K. S., Huang, T., and Blostein, S. D. (1987). Least square fitting of two 3-d point sets. IEEE Trans. Pact. Anal. Machine Intelligence. vol. PAMI-9
14. Gu, M., Demmel, J. W., and Dhillon, I. (1994). Efficient computation of the singular value decomposition with applications to least squares problems. Technical Report CS-94-257, institute Knoxville, TN, USA.
15. Flannery, B. P., Teukolsky, S. A., and Vetterling,W. T. (1992). Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed. Cambridge England: Cambridge University Press.
16. Golub, G. H. and Loan., C. F. V. (1989). Matrix Computations, 2nd Edition. Johns Hopkins. University Press
17. Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. IEEE transactions on pattern analysis and machine intelligence, Vol. 14, NO. 2.

Une Approche Basée Agent pour la Détection de Régions

KAZAR Okba[1], GUIA Sana Sahar[1]

[1] Département de l'Informatique Université de Biskra 07000 Algérie

{kazarokba,guia_sana}@yahoo.fr

Résumé. La segmentation d'images est une opération de bas niveau de traitement d'images qui consiste à localiser dans une image les régions (ensemble de pixels) appartenant à une même structure, et qui conditionne fortement la réussite globale d'une entreprise d'analyse d'images. L'objectif de ce travail est de proposer une nouvelle approche de la segmentation d'images, commençant par la transposition d'un phénomène collectif en biologie qui est une source d'inspiration pour proposer des méthodes multi-agents de résolution de problèmes, et terminant par une coopération entre agents pour la fusion et la division de régions.

Mots-Clés: Système Multi-agent, Détection de régions, Inspiration biologique.

Abstract. The Image segmentation is an important step for several applications in both industrial vision and medical imagery. It is a low-level image processing that consist to locate in an image the regions (all pixels) belonging to the same structure, which strongly determines the overall success of an analysis images. The objective of this work is to offer a new approach to the problem of segmentation, starting with the implementation of a collective phenomenon in biology that is a source of inspiration to propose methods of multi-agent problem solving, and ended by cooperation between agents for the merger and division of regions.

Keywords: Multi agent-system, Region Detection, Biological inspiration

1 Introduction

La segmentation d'images est l'un des domaines les plus actifs en analyse d'images et en vision par ordinateur. Elle est une étape algorithmique fondamentale de nombreux systèmes d'analyse automatique d'images. Son rôle est de délimiter dans l'image étudiée un ensemble de zones pertinentes pour l'interprétation ou la modélisation de la scène perçue.

La recherche d'une méthode performante pour une application donnée, passe la plupart du temps par la comparaison de quelques méthodes disponibles et bien maîtrisées et par la modification d'une méthode existante afin de l'adapter. Quand on

parle de segmentation d'images et malgré les avancées significatives, les solutions actuelles ne permettent pas de résoudre le problème de l'unification de la segmentation sous un formalisme commun. Il s'agit d'un domaine fondamental et très vaste, qui a suscité et suscite encore de nombreuses recherches.

C'est avec la mise en œuvre des agents, puis des systèmes multi-agents, que l'on voit apparaître l'ensemble des activités individuelles et, peut-être surtout, des interactions entre composants qui permet d'obtenir le contrôle distribué de l'application informatique sur l'ensemble des composants, qui acquièrent de ce fait une forme d'autonomie, d'abord opérationnelle, puis, éventuellement, décisionnelle, domaine dans lequel on s'intéressera à la Résolution Distribuée de Problèmes.

Une méthode peut être conçue pour la résolution de problème de segmentation d'images en se basant sur les capacités offertes par les systèmes multi-agents destinés à effectuer de la résolution collective de problème.

Dans la suite nous allons passer en vu les travaux existants dans ce domaine, ensuite le processus d'inspiration que nous avons suivi dans notre approche va être mentionné puis la modélisation de l'approche proposée est détaillée dans la quatrième section, et nous terminons par une conclusion et quelques perspectives.

2 Travaux existants

IL existe des modèles inspirés des insectes sociaux qui ont déjà été élaborés et utilisés pour la segmentation d'image par une approche multi-agents. Dans [RAM 00] le comportement est inspiré des fourmis pour détecter des contours, bien que ce modèle est loin d'être compatible avec la théorie de Gestalt de perception. Dans [LIU 99] il est inspiré des automates cellulaires et différentes catégories d'agents associés chacune à une région explorent l'image et marquent les pixels lorsqu'ils appartiennent à la région correspondante.

Dans [BOU 01], une approche d'inspiration biologique pour la résolution collective de problème a été proposée. Dans ce travail les mécanismes de construction de toile chez les araignées sociales ont été simulés, puis le modèle comportemental est transposé pour l'appliquer à la détection de régions dans des images à niveaux de gris.

Dans le travail de [OUA 02] est proposée de segmenter l'image en utilisant la méthode de classification en s'inspirant des comportements collectifs et auto-organisé des fourmis dans la nature, ils ont utilisé le système de fourmi Max Min (Max Min Ant System « MMAS ») pour résoudre le problème de classification.

D'autres travaux utilisant le système multi-agents pour la segmentation d'images sans pour autant s'inspirer des phénomènes collectifs ont été réalisés. Dans [BUR 00] est proposé de combiner deux approches, une approche procédant par croissance de région et l'autre exploitant un algorithme génétique, toutes deux travaillant de manière concurrente dans un cadre multi-agents.

Le travail de [BOV 01] décrit une approche de segmentation d'image dans laquelle des agents synchronisés combinent le traitement bas niveau d'images et le raisonnement haut niveau.

Dans [RIC 01] et [RIC 02], des agents situés coopèrent pour segmenter des IRM cérébrales. On y trouve diverses catégories d'agents : un agent de contrôle global, des

agents de contrôle locaux et au niveau le plus bas, les agents de segmentation, spécialisés dans la détection des trois types de tissus cérébraux (matière blanche, matière grise et liquide céphalorachidien).

Le travail de [DUC 01] s'appuie sur la structure de pyramide irrégulière pour gérer le processus de fusion de régions et assurer la convergence de la segmentation.

[SET 02] a développé une plate forme multi-agents pour la segmentation d'images en s'appuyant sur la coopération des agents régions et contours (coopération région-région et contour-région pour la fusion de régions, et une coopération région-contour pour la division d'une région).

Dans [MAZ 05], une méthode basée sur l'utilisation d'un système multi-agents auto-adaptatif permettant une segmentation fiable d'une image 3D dense a été proposée.

Dans [IDI 05], est proposé une méthode hybride de segmentation d'image par une approche multi-agent basée sur une pyramide irrégulière duale.

D'après les approches citées ci-dessus, nous constatons que ces travaux présentent encore quelques limitations, et que les méthodes de segmentation bio-inspirée et en particulier celles basées sur les techniques d'intelligence en essaim constitue une voie de recherche très intéressante et mérite une étude approfondie. Les méthodes basées sur les insectes sociaux apportent des solutions originales pour l'obtention d'une segmentation optimale, et peuvent être couplés ensemble pour pallier à leurs insuffisances tout en réunissant leurs qualités.

Dans la suite nous présentons la modélisation de l'approche proposée, en utilisant l'inspiration du modèle de construction de toiles observé chez les araignées pour la segmentation d'images en régions homogènes.

3 Principe du processus de l'inspiration biologique pour la résolution du problème

Le processus d'inspiration des phénomènes collectifs de la biologie passe par plusieurs étapes : il commence d'abord par l'étude du modèle biologique, en étudiant quelques généralités concernant les araignées sociales. Ensuite la construction du modèle de simulation par lequel nous décrivons le modèle comportemental qui a été utilisé pour les simulations informatiques en termes d'environnement ; d'agents et leur comportements ; et de dynamique du système. Enfin la transposition du modèle pour la résolution d'un problème spécifique, ici le modèle comportemental a été transposé pour l'appliquer à la détection de régions dans des images à niveaux de gris.

Dans ce travail, nous essayons de détecter toutes les régions dans une image, par l'utilisation des informations de recouvrent de toiles pour la différenciation et la distinction des régions.

4 Modélisation

Nous distinguons deux catégories d'agents :

- Un agent de gestion du système (agent moniteur), contient les informations nécessaires au fonctionnement du système multi-agents.
- Des agents nommés agents de segmentation (agents araignées, agents toiles), chargés de l'amélioration du pré segmentation initiale pour obtenir des régions homogènes.

Donc notre système est constitué des agents suivants :

- Agent moniteur qui est le centre de contrôle de notre système. C'est lui qui crée les agents de segmentation (agents araignée et agents toile) et les initialisent en utilisant les informations obtenus lors de l'étape de pré-traitement, il est aussi responsable de l'ordonnancement et du lancement des taches de ses agents (agents de segmentation), de plus, il décide de la suppression des agents devenus inactifs.
- Agent araignée qui parcoure l'image à la recherche des régions homogène en construisant des toiles sur les régions pré segmentées qui les considère comme homogènes, en simulant le comportement des araignées sociales.
- Agent toile qui sert à l'interprétation des toiles construites par les agents araignées à des régions finales.

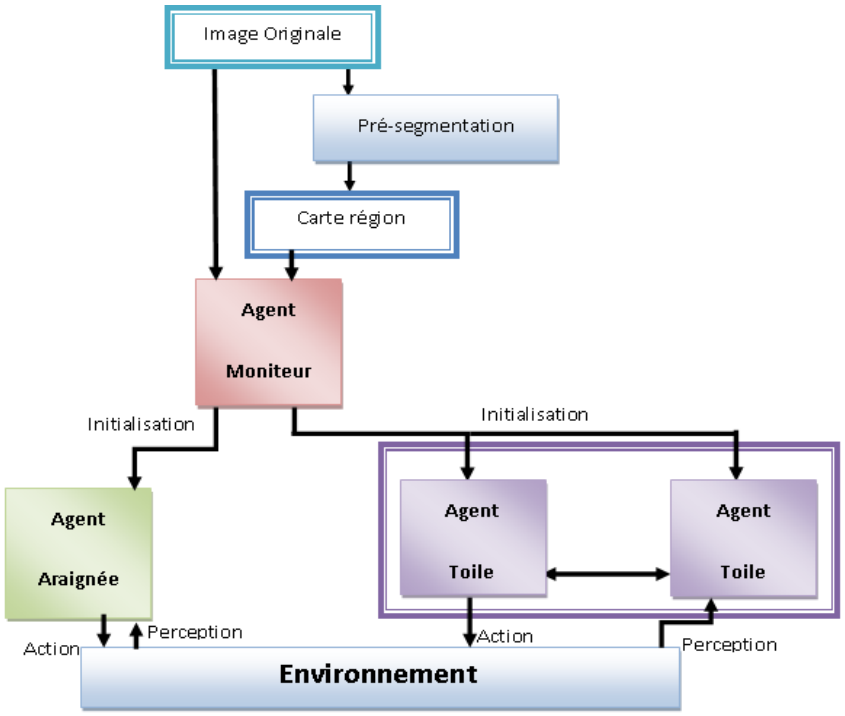


Fig. 1 Architecture générale du système proposé

4.1 Architecture de chaque agent

Nous allons maintenant détailler l'architecture de chaque agent du système

4.1.1 L'agent moniteur

C'est lui qui gère le système et les agents lors de leurs exécutions. Son rôle peut être décrit dans les notes suivantes :

- Création et initialisation des agents.
- Ordonnancement et lancement des taches.
- Mise à jour du système
- Suppression des agents

4.1.2 L'agent araignée

L'agent araignée représente une primitive région présente dans la carte de région. Chaque agent de ce type marque de son identifiant la zone de l'environnement correspondant à sa primitive. Au fur et à mesure de l'exploration de leurs régions, les araignées construisent leurs toiles.

4.1.2.1 Caractéristiques de l'agent araignée

Les caractéristiques de l'agent correspondent aux paramètres conditionnant ses comportements et son état interne :

- Le niveau de gris de référence : RefLev $\in$ [0 .. 255] qui représente le niveau de gris du pixel initial.
- Les paramètres d'attraction de soie deux types de paramètres possibles :
  - Pdragline : probabilité de suivre ou non un fil de soie (dans le cas de détection d'une seule région), ou bien
  - Attractself et Attractother qui déterminent respectivement :
    - La probabilité d'attraction de soie créée par l'araignée même, et
    - La probabilité d'attraction de soie créée par une autre araignée (dans le cas de détection de plusieurs régions simultanément)

4.1.2.2 Perceptions

Les perceptions offrent les informations locales disponibles dans l'environnement et sur lesquelles se base la décision. Les informations requis dans ce système sont les suivants :

- L'ensemble des pixels voisins du pixel p « Neighp »
- L'ensemble des pixels accessible en suivant un fil posé « Scutsp »
- L'union des deux ensembles précédents « Accessp »

4.1.2.3 les différents comportements de l'agent

Les mêmes comportements comme dans [BOU 01] sont nécessaires pour la construction de toiles :

- Déplacement : consiste à choisir un pixel parmi ceux accessibles selon une probabilité de distribution. La probabilité de déplacement à un pixel accessible (nommé p) dépend de la façon de lui accéder
  - i) par le déplacement à un pixel adjacent ($p \in Neihgp$)
  - ii) par la poursuite d'un fil posé ($p \in Scutsp$)

Le comportement de déplacement est implémenté selon la façons de calcul de probabilité de distribution qui correspond à la compétition entre plusieurs agents pour la détection de plusieurs régions, dont représente. Pour la détection de plusieurs régions dans une image, nous mettons le coefficient Attractother à zéro, pour avoir différents processus s'exécutent dont chacun ignore l'autre, et aucun agent n'influe le comportement de l'autre.

- Pose de fil : il est contextuel et dépend du niveau de gris de référence
- Retour : afin que l'araignée ne parcourt pas l'intégralité de l'image et ne tisse toutes les régions ayant le même niveau de gris.

4.1.2.4 Les interactions des agents araignées

Les interactions des agents sont basées sur le principe de coordination par stigmergie : les actions des agents modifient l'environnement, de son coté l'environnement aussi modifie les actions futurs des agents. Ce principe est modélisée implicitement dans le comportement de déplacement influencé par la soie, plus il y a de soie vers une position, plus celle-ci a de chance d'être choisie.

4.1.3 L'agent toile

Après la construction de toile par les araignées l'agent toile se charge de l'interprétation des toiles construites en régions et ceci par l'application du principe de distinction de toile selon l'individu, il est caractérisé par les paramètres suivants :

- l'individu qui désigne l'agent araignée qui a créé la toile,
- la région initiale RégInit par laquelle l'individu a commencé la construction de sa toile,
- l'ensemble des régions EnsRég qui sont inclus dans la toile
- l'ensemble des toiles EnsToile qui représentent les toiles des régions appartenant à EnsRég

Chaque agent toile coopère avec les toiles appartenant à EnsToile pour décider de fusionner ou non les régions appartenant à sa toile, selon le recouvrement de toile, nous avons trois cas possibles :

4.1.3.1 Recouvrement total de toiles

Nous avons un recouvrement total lorsque la toile représentée par l'agent courant et les toiles représentées par les agents appartenant à EnsToile recouvrent le même ensemble de régions EnsRég, ceci est modélisé comme suit :

Un agent toile regroupe un ensemble de régions EnsRégi, il y a recouvrement total si :

∀ l'agent toilej ∈ EnsToilei , EnsRégj = EnsRégi

La figure suivante illustre ce concept :

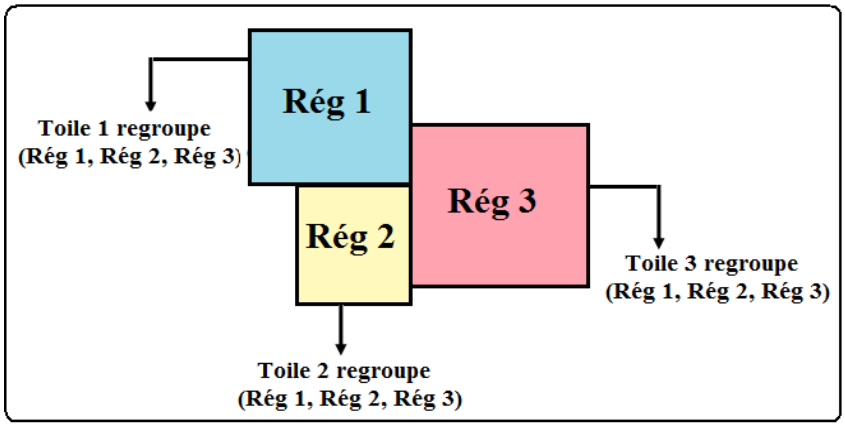


Fig. 2 Exemple de recouvrement total

Lorsqu'un agent toile se trouve dans une situation de recouvrement total avec les agents toile appartenant à EnsToile, les régions regroupées par ces agents toile sont fusionnées. Cet agent toile envoie à l'agent moniteur un message lui informant qu'une fusion d'un recouvrement total est effectuée. L'agent moniteur change l'état de cet agent, ainsi que les agents toile appartenant à l'EnsToile de l'état actif à l'état inactif.

4.1.3.2 Recouvrement partiel de toiles

Nous disons qu'il y a un recouvrement partiel lorsqu'il existe des agents toile appartenant à l'EnsToile de l'agent toile courant et dont l'EnsRég de ces agents toile n'est pas égale à celui de l'agent toile courant. En d'autres termes :

Un agent toile regroupe un ensemble de régions EnsRégi, il y a recouvrement partiel si :

∃ un agent toilej ∈ EnsToilei, tel que EnsRégj ≠ EnsRégi

Ceci est illustré dans la figure suivante :

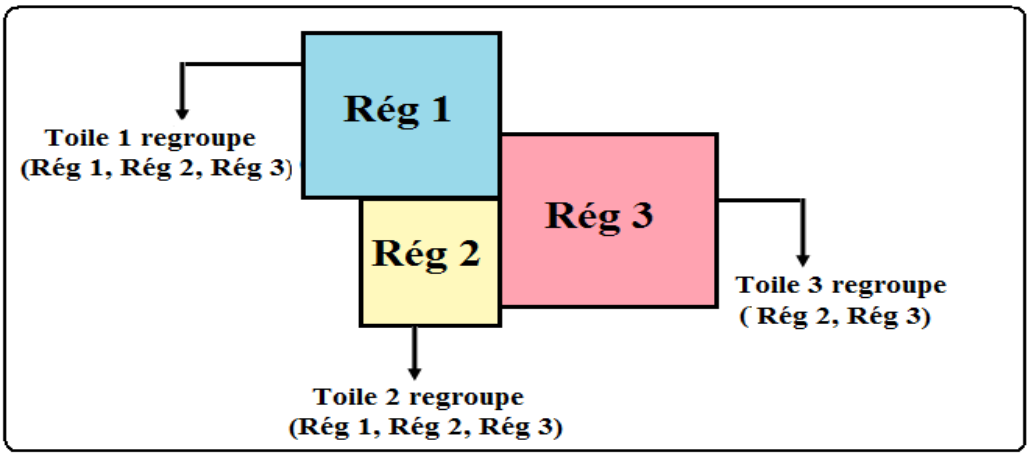


Fig. 3 Exemple de recouvrement partiel

Dans ce cas les régions initiales des agents toiles qui ont le même ensemble EnsRég sont fusionnées. Un message est envoyé à l'agent moniteur lui informant qu'une fusion d'un recouvrement partiel est effectuée. Un seul agent toile reste actif parmi ceux dont leurs RégInit sont fusionnées, c'est celui dont son identifiant est le plus petit, les autres passent à l'état inactif. Dans la figure ci-dessus les régions Rég 1 et Rég 2 sont fusionnées, agent Toile 2 devient inactif, et les paramètres de l'agent Toile 1 sont mises à jour (la région initiale de l'agent Toile 1 devient Rég 1 et Rég 2).

Ensuite l'agent restant actif envoie un désir de fusion aux agents appartenant à EnsToile dont leurs RégInit n'ont pas été fusionnées dans la dernière fusion effectuée, s'il reçoit une acceptation d'un de ces agents, la région initiale de ce dernier est fusionnée avec la région initiale de l'agent courant, et il passe à l'état inactif. Dans l'exemple précédent l'agent Toile 1 coopère avec l'agent Toile 3 en leur envoyant un désir de fusion, si cet agent (Toile 3) lui envoie une acceptation, la fusion des régions initiales des ces agents toile est effectuée et l'agent Toile 3 devient inactif.

Si un agent toile reçoit plusieurs désirs de fusion de plusieurs agents, il décide quelle demande va l'acceptée suivant l'EnsRég :

Il fait l'intersection entre son EnsRég et l'EnsRég de l'agent qui lui a envoyé le désir de fusion, l'agent dont la cardinalité de l'intersection de leurs EnsRég est plus grande accepte sa demande de fusion.

4.1.3.3 Division

Si au sein d'une même région deux toiles distinctes sont construites, celle-ci est divisée en deux régions chacune.

La communication entre les agents est réalisée par l'envoi de message, nous avons deux types de messages : le désire de fusion et l'acceptation ou le refus de la fusion (lorsque cet agent veut fusionner sa région initiale avec une région initiale d'un autre agent)

4.2 Fonctionnement du système

Le problème est d'extraire les différentes régions existantes dans une image. L'approche proposée dans ce mémoire utilise des systèmes multi-agents pour l'inspiration du phénomène collectif de la construction de toile chez les araignées sociales. Chaque agent araignée explore en construisant sa toile. Il peut dépasser les limites de sa région s'il trouve qu'il y a une homogénéité par rapport aux régions voisines. La gestion de la coopération entre agents et la dynamique du système est assurée par le principe de la stigmergie : coordination par stigmergie qui est modélisée implicitement dans le comportement (déplacement influencé par la soie, plus il y a de soie vers une position, plus celle-ci a de chance d'être choisie). Enfin, les agents toiles sont créés pour l'interprétation des toiles construites en régions, par une coopération entre ces agents pour la fusion ou la division de leurs régions.

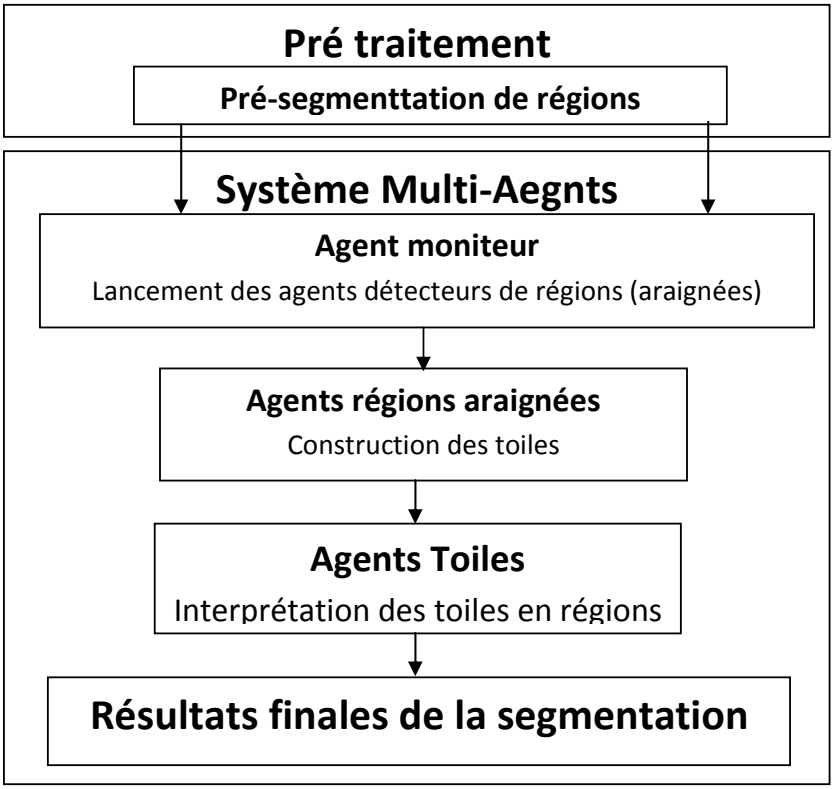


Fig. 3 Exemple de recouvrement partiel

5 Conclusion

Dans cet article nous avons donné les détails de modélisation du système, en détaillant l'architecture de chaque composant, et leurs types de communication, dans ce contexte nous avons modélisé d'abord en s'inspirant des araignées sociales, la détection de région des images en niveau de gris et ensuite nous avons utilisé le résultat de cette étape pour la fusion et la division de régions, sans que celle-ci ne soit inspirée des phénomènes collectifs en biologie. C'est cette combinaison que nous espérons être utiles pour les résultats de la segmentation.

En effet, nous envisageons plusieurs perspectives, pour poursuivre ce travail, et qui concernent les points de vus suivants :

Il nous parait intéressant de modéliser notre système dans un cadre plus générale en exploitant l'information contour pour améliorer le critère de décision de fusion ou de division de régions soit par la coopération région contour pour la fusion ou la division de région, soit par l'implémentation de la coordination stigmergique pour la prise de décision de fusion ou de division de régions. Dans ce contexte nous pouvons observer d'autres comportements collectifs des araignées sociales et les exploiter pour améliorer la segmentation finale de l'image. En effet, la recherche d'une méthode efficace de segmentation d'image demeure l'objectif de toute proposition.

6 References

[BOU 01] Christine Bourjot, Vincent Chevrier : « De la simulation de construction collective à la détection de régions dans des images à niveaux de gris : l'inspiration des araignées sociales ». LORIA, UMR 7503 JFIADSMA, 2001.

[BOV 01] E.G.P. Bovenkamp, J. Dijkstra, J.G. Bosch, and J.H.C. Reiber « Collaborative Multi-agent IVUS Image Segmentation » Springer-Verlag Berlin Heidelberg 2001

[DUC 01] E. DUCHESNAY « Agents situés dans l'image et organisés en pyramide irrégulière. Contribution à la segmentation par une approche d'agrégation coopérative et adaptative ». Thèse de Doctorat de l'Université de Rennes-1 (2001)

[IDI 05] Idir karima, Merouani Hayet et Tlili Yamina « Proposition d'une Pyramide Duale D'agents pour la Segmentation d'Image » SETIT 2005 3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunications – TUNISIA

[LIU 99] LIU J. TANG Y. « Adaptive image segmentation with distributed behavior based agents. » IEEE Trans. On Pattern Analysis and Machine Intelligence, 21(6):544-551, June 1999.

[MAZ 05] S. Mazouzi, M.C. Batouche et Z. Guessoum « Un Système multi-agents auto-adaptatif pour la segmentation et la reconstruction de scènes 3D » SETIT 2005 3rd International Conference: Sciences of Electronic, Technologies of Information and Telecommunications – TUNISIA

[OUA 02] Salima Ouadfel, Mohamed Batouche « Unsupervised Image Segmentation Using a Colony of Cooperating Ants » Springer-Verlag Berlin Heidelberg 2002

[RAM 00] RAMOS V., ALMEIDA F., « Artificial Ant Colonies in Digital Image Habitats – A Mass Behaviour Effect Study on Pattern Recognition », Second International Workshop on Ants Algorithmes, Ants2000, Bruxelles, 2000.

[RIC 01] N. RICHARD, M. DOJAT, C. GARBAY « Dynamic adaptation of cooperative agents for MRI brain scans segmentation ». Artificicial Intelligence in Medecine (2001)

[RIC 02] N. RICHARD, M. DOJAT, C. GARBAY « Situated cooperative agents: a powerful paradigm for MRI brain scans segmentation ». European Conf. On AI – ECAI-2002.

[SET 02] Hakim SETTACHE « Une plate-forme multi-agent pour la segmentation d'images : Application dans le domaine des IRM cérébrales 2D » Université de Caen Septembre 2002

Segmentation d'image de biopuces par Champ de Markov tolérant les déformations locales de grille.

Christophe Gouinaud

LIMOS-ISIMA, Campus des Cézeaux, 63177 Aubière, gouinaud@isima.fr

Résumé Notre travail a pour but de fournir une méthode de segmentation d'image permettant d'augmenter la qualité des mesures faites grâce à des biopuces. Nous présentons toutes les étapes nécessaires à la réalisation d'une segmentation et proposons diverses améliorations tout en les comparant aux méthodes usuelles. Enfin, nous montrons des résultats sur deux jeux d'images présentant des situation différentes.

1 Introduction

L'intérêt de l'utilisation des biopuces cdna pour la génétique n'est plus à démontrer [7]. Cette technologie complexe arrive maintenant à maturité et son utilisation s'étend à la modélisation des relations gène-expression-individu. De ce fait, le défi actuel est l'amélioration de la précision des mesures réalisées de façon à augmenter la qualité des expressions estimées et donc les résultats fonctionnels.

D'un point de vue concret, l'utilisation des biopuces fait appel à des technologies très différentes. Les sources d'imprécision dans le process de production sont nombreuses et il est donc fondamental de maîtriser chaque étape et d'appliquer avec rigueur chaque procédure. Parmi toutes les étapes, celle du traitement d'images est certainement celle où les gains de précision les plus importants peuvent être obtenus [4] [11]. Cette phase est évidemment affectée par toutes les étapes précédentes. Le travail présenté ici s'attaque donc à l'amélioration du calcul des expressions en proposant une nouvelle méthode de segmentation des images.

La première motivation de ce travail est de construire une méthode de traitement d'images plus générique que celle proposée par les concepteurs de robot spoteur et de scanner à biopuces. En effet, ces derniers sont souvent associés à des logiciels de traitement d'images développés dans le but d'utiliser les métadonnées décrivant la lame fournie par les robots d'un même constructeur [20]. Ceci complique en particulier les comparatifs entre des biopuces produites avec des appareils différents et donc les échanges de données images entre laboratoires.

Ce travail vise aussi à arriver à une qualité plus importante de segmentation des images et en particulier à gérer correctement les déformations de spots, à corriger au mieux les déplacements de sonde et donc à fournir les meilleurs masques possibles pour le calcul des expressions. La méthode proposée ici est originale car elle combine des approches différentes pour chaque étape nécessaire à la détection des spots. Elle s'inspire de méthodes utilisées dans d'autres contextes tels que la télédétection sur images SAR qui sont également acquises en lumière cohérente et présentent donc le même genre de

bruit [8]. Elle présente aussi l'intérêt de faire intervenir au minimum un opérateur tout en tolérant l'utilisation d'images issues de matériel quelconque.

Ce document est présenté en trois parties : la première est consacrée au contexte de cette étude, la seconde traite du problème de positionnement de la grille sur l'image et la dernière présente notre méthode de segmentation et ses résultats.

2 Contexte

Le calcul de l'expression des spots nécessite deux prètraitement, un de localisation des zones où se trouvent les sondes sur l'image, nous l'appellerons positionnement de la grille, et une réalisant le calcul de la liste des pixels appartenant à un spot qui est la segmentation. Le positionnement est essentiel pour éviter des erreurs d'association risquant de conduire à des résultats erronés d'expérience [19]. Il est couramment réalisé suivant deux méthodes qui sont l'utilisation de métadonnées ou l'association zone-spot après segmentation de l'image.

- l'utilisation de métadonnées nécessite l'emploi d'appareils et de logiciels d'un unique constructeur ou au minimum une interopérabilité entre tous les acteurs de la chaîne.
- la stratégie inverse de la précédente méthode consiste à réaliser la segmentation de l'image en spot puis à plaquer dessus la grille des sondes fournie par le spoteur [14]. Le but de cette méthode est de corriger les éventuels effets de glissement des spots ou autres artefacts introduits durant la production de la lame, la phase d'hybridation et l'acquisition de l'image. Il s'agit ici soit de processus d'association individuelle de spots, soit de méthode associant recherche des blocs de dépôt et association des spots dans les blocs. Après l'essai de diverses variantes de cette classe de méthodes [10], nous devons constater qu'il s'agit d'un problème ouvert car ces techniques ne peuvent s'affranchir des risques inhérents au glissement de sonde.

Face à ce constat, nous proposons ici une méthode semi-automatique de positionnement, décrite dans la partie 3, permettant de corriger les défauts de la deuxième classe de méthode de positionnement sans métadonnées.

Le but de la segmentation est de séparer les pixels d'un spot de ceux du fond. Les méthodes les plus couramment employées fonctionnent soit à l'aide d'algorithmes s'appuyant sur la reconnaissance des formes [9] ou bien à l'aide de classifications basées sur la valeur des pixels [21].

- dans le cas où on utilise des critères basés sur la forme, on présuppose que les spots visibles sur l'image ont des caractéristiques géométriques conformes aux sondes déposées. Les caractéristiques présupposés sont donc le plus souvent la circularité et la convexité du spot. Les techniques employées utilisent soit des cercles fixes ou ellipses adaptatifs mais le positionnement et la forme sont présupposés [17]. Bien évidemment, ces conditions ne sont que rarement réalisées et les résultats produits conduisent à l'intégration de nombreux pixels de fond dans les expressions calculées.
- l'utilisation de classification est basée sur l'existence d'une différence statistique entre les valeurs les pixels du fond et ceux dles spots. Ces méthodes sont le plus

souvent basées sur l'analyse de statistiques du premier ordre discriminant localement les pixels du spot de ceux du fond [14]. Elles sont parfois associées à des techniques de croissance de région ou de contours actifs qui réintroduisent l'aspect reconnaissance de forme.

Nous proposons ici une méthode hybride basée sur une double prise en compte de la forme et des signaux visant à s'affranchir des petites erreurs de positionnement et intégrant des hypothèses de connexité par arcdes spots. D'autres travaux ont développé des approches similaires telles que celles basées sur les classifications de Man-Whitney ou basées sur des techniques d'association dissociation ([3]). Notre approche se distingue par l'emploi de critères topologiques de voisinage associé une approche statistique évoluée. La partie **??** expose notre approche basée sur une segmentation markovienne.

3 Positionnement de la grille.

La première des tâches nécessaire à la segmentation de biopuces est le repositionnement de la grille des spots sur l'image scannée [13]. Ce sujet a fait l'objet de nombreux travaux, utilisant des grilles déformables, basés sur une prédétection des spots ou basés sur la reconnaissance de blocs de spots [1]. Cependant, les résultats de ces travaux se confrontent difficilement à la réalité des laboratoires et ont du mal à positionner correctement les grilles.

La double incertitude induite par les spots non exprimés et le positionnement incertain de la lame implique l'utilisation d'un positionnement semi-automatique. Le positionnement global de la grille se fait donc en désignant des pixels de l'image et leurs positions idéales et en utilsant un modèle de déformation.

Le choix du modèle doit être fait en fonction de la physique du système d'acquisition de façon à compenser les déformations par rapport à la prise de vue idéale. Dans le domaine des scanners à objectif confocal pour biopuces, la prise de vue est correcte quand ll'orientation de la lame est telle que son bord le plus long est parallèle à la trajectoire du centre de rotation du bras du scanner. Les déformations globales de l'image sont donc modélisables sous forme d'une combinaison d'applications affines en 2D dans le plan de l'image qui s'écrit :

$$\begin{cases} X^{'} = a_{11}X + a_{21}Y + a_{31} \\ Y^{'} = a_{12}Y + a_{22}Y + a_{32} \end{cases}$$

où X, Y sont les coordonnées d'un spot dans le repère de la grille, $X^{'}, Y^{'}$ dans le repère de l'image et les a_{ij} les coefficients du modèle.

Afin de limiter les erreurs, nous utilisons une méthode de détermination des paramètres par moindres carrés utilisant au minimum quatre points, ce qui permet d'absorber des petites erreurs de positionnement tout en distinguant les incohérences flagrantes. Pratiquement, nous affichons l'image et la grille placée sur l'image par un simple centrage puis l'opérateur indique au moins quatre amers en cliquant sur l'image et sur la grille. Ensuite, nous calculons le modèle d'interpolation et déplaçons la grille.

Après le calage global de la grille, nous constatons souvent des erreurs résiduelles de valeur suffisante qui empêche d'utiliser le bord de la zone comme étant certainement du fond.

L'observation des déplacements [2] nous conduit à distinguer le cas où le spot n'est pas à la place prévue mais sans être déformé, ou il est donc quasi circulaire, de celui ou il est déformé

Nous avons donc développé ici une approche duale basée sur la corrélation avec un modèle circulaire associé à un calcul barycentrique basé sur les luminances. Le premier algorithme vise à se recaler sur les spots qui n'ont que peu de déformations alors que le deuxième vise à prendre en charge les spots déformés et non uniformément exprimés.

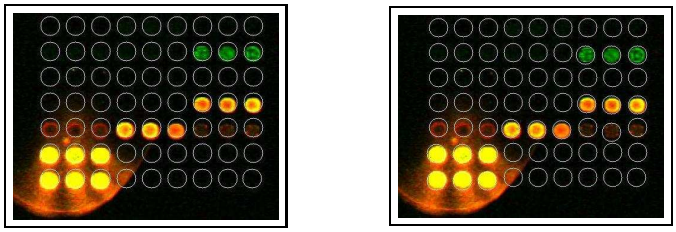


FIGURE 1. Affinage de la postion des spots, en haut avant en bas après.

Pratiquement, nous calculons la corrélation entre le premier canal image et un modèle circulaire binaire de même taille que les spots recherchés. Nous déplaçons ce modèle sur la fenêtre entourant le spot et calculons le corrélogramme la valeur maximum obtenue. Si celle-ci est supérieure à 0,75, nous considérons le point obtenu comme fiable. Nous réalisons ensuite le même calcul sur le deuxième canal. Nous calculons ensuite le barycentre de la fenêtre entourant le centre du spot donné par la grille nous disposons ainsi de positions différentes que nous fusionnons par un modèle linéaire dont les coéficients sont liées au valeurs de corrélation obtenu.

Cette technique s'est avérée particulièrement efficace dans le cas de spots ayant la forme de donuts et produit une grille affinée telle qu'elle est représentée sur la figure 1.

4 Segmentation

Comme cela est dit dans la partie précédente, notre méthode de segmentation nécessite, pour son initialisation, une estimation de la moyenne du spot et du fond l'entourant. Cette estimation est problématique parce qu'on ne connaît pas les pixels constituant le spot et que cette connaissance est notre but.

Pour estimer cette moyenne, on utilise classiquement les pixels entourant le centre du spot donné par la grille ou une forme circulaire représentant la forme idéale du spot. Ces deux approches ont pour défaut d'être trop sensibles au positionnement de la grille et d'augmenter les risques de confusion entre le fond et le spot. La méthode des centiles décrite dans [5] améliore un peu les estimations de moyenne mais reste trop conditionnée par la qualité géométrique des spots.

En observant un grand lot d'images et en réalisant de nombreuses segmentations manuelles, nous nous sommes aperçus que si la forme des spots pouvait être très variable, leur surface varie peu. En fait, les déformations de spot sont la plupart du temps

dues à un glissement de produit sur la lame. Notre méthode d'estimation est donc basée sur ce fait.

Nous procédons à l'estimation de la moyenne du spot en observant son influence sur la moyenne du fond entourant le spot. Pratiquement, cela consiste à écrire que la moyenne d'une zone (Z) contenant un spot est la somme pondérée de la moyenne des valeurs du fond (F) et des pixels (S) ce qui nous permet de déduire la moyenne $E(S)$ de la zone S en fonction de $E(Z)$ et lde $E(F)$:

$$E(S) = \frac{E(Z)card(Z) - E(F)card(F)}{card(S)}$$

La moyenne ($E(F)$) du fond est mesurée sur le bord de la boîte entourant le spot. La surface théorique du spot ($card(S)$) est donnée par les caractéristiques de la lame. La population du fond ($card(F)$) est donnée par $card(Z) = card(F) + card(S)$.

Avant même parler de notre segmentation, il faut poser les conditions d'emploi des deux canaux à notre disposition. Nous avons choisi de combiné les informations produite par les deux canaux car les biopuces sont conçus pour que l'information donnée par les deux images soit différente dans leur signification biologique. En théorie, on doit obtenir une même forme mais avec des intensités très différentes. Ces intensités peuvent être très proches ou très corrélées mais ne le sont pas dans le cas où le niveau d'expression d'un seul canal est très faible. Quand un spot s'exprime, cela signifie que la lumière perçue par les capteurs provient de la fluorescence et que la réponse du fond n'est plus visible. A l'inverse, dans une zone non hybridée, la lumière mesurée par le scanner provient uniquement du fond. Dans ce dernier cas, la sonde est transparente et ne fournit alors aucune information de forme exploitable et cela même si une certaine déformation du signal de fond peut être mesurée. Il convient donc d'adopter le schéma de fusion donnée suivant :

- Dans le cas numéro un, la zone segmentée comme appartenant à la sonde doit provenir uniquement de l'image bien exprimée,
- Dans le cas numéro deux, la zone d'expression est celle où les deux segmentations ont conclu à la classe spot,
- Dans le cas numéro trois, l'information du canal qui a une forme très différente du dépôt doit être rejetée ou sursegmentée vers celle de l'image bien exprimée. Il faut noter que ce dernier cas est très rare.

Ceci étant posé, nous avons choisi de segmenter séparément les images des deux canaux CY5 et CY3 puis d'appliquer le schéma de fusion défini ci-dessus en distinguant les cas par une post-estimation des moyennes. Concrètement, nous calculons les moyennes du fond et du spot, après segmentation sur chaque canal, dans l'image d'un canal en utilisant la forme segmentée sur l'autre canal et comparons les variations de statistiques sur l'un et l'autre. Ceci nous permet d'orienter notre fusion.

Il nous faut également parler des raisons du choix d'une classification Markovienne. Notre segmentation est basée sur le fait que les statistiques du premier ordre des spots sont différentes de celles du fond. Les biologistes sont intéressés par la médiane des pixels constituant les spots car ils considèrent celle-ci comme plus représentative de l'expression des gènes que la moyenne. Notre segmentation est cependant basée sur la moyenne pour au moins deux raisons :

– si nous segmentons grâce à la médiane, nous risquons de biaiser la mesure finale
– la moyenne est beaucoup plus rapide à calculer que la médiane qui demande de plus une réévaluation complète quand on ajoute ou retire un individu de l'espace d'étude. Cette rapidité est essentielle pour pouvoir exploiter notre travail dans un logiciel interactif tournant sur des ordinateurs personnels.

Nous utilisons donc une classification itérative basée sur la distance d'un pixel à la moyenne de la classe à laquelle il est censé appartenir. Dans ce cadre, une première approche très classique [16] qui s'appelle une classification par nuée dynamique peut être tentée. Concrètement, nous calculons dans ce cas pour chaque pixel la distance à la moyenne de chaque classe et en déduisons son appartenance. La distance utilisée est alors de la forme :

$$D(i, c) = |X - E(X_c)|$$

où $E(X_c)$ est la moyenne de la classe des pixels de la classe C. Nous appelerons D attache aux données dans la suite.

Une fois tous les pixels classés, nous réévaluons la moyenne des classes et reclassons chaque pixel. Ce procédé est réalisé localement pour chaque sonde sur une petite fenêtre entourant le spot. En effet, les valeurs d'expression étant très disparates d'un spot à l'autre, leur moyenne est très différente et donc une approche globale à l'image serait aberrante. Le résultat que nous obtenons est une image de masques, superposable à l'image d'origine, dont les pixels ont une valeur zéro pour le fond et une valeur correspondant au numéro du spot.

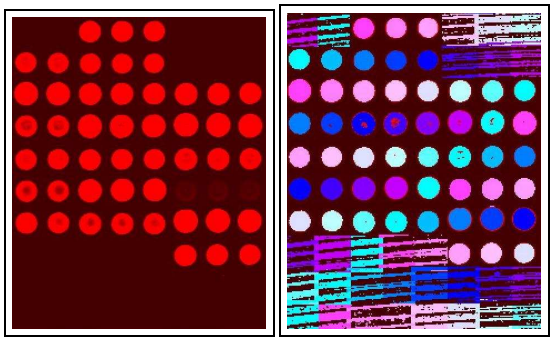


FIGURE 2. Classification par nuées dynamiques : à gauche, le canal rouge d'une image de biopuce, à droite, le résultat.

Sur la figure 2 nous montrons le résultat de ce traitement et nous pouvons constater qu'il est loin d'être parfait. Nous remarquons en particulier que certains spots, au centre de l'image, comportent des trous et que le traitement des zones de faible intensité aboutit à un résultat aberrant.

Les classifications telles que décrites ci-dessus n'utilisent qu'un critère basé sur la valeur des pixels. Ceci explique pourquoi beaucoup de pixels isolés sont mal classés comme on peut le voir sur la figure 2. Le problème vient du fait qu'il existe dans l'histogramme de l'image une zone confondue où se mélangent des pixels appartenant au fond et aux spots. Cette confusion ne peut être compensée que par l'introduction d'informations supplémentaires.

De ce fait, nous allons introduire dans notre classification des critères spatiaux qui nous permettront de mieux classer les pixels isolés ayant une valeur aberrante. Les spots étant le plus souvent connexes par arc, il apparaît qu'un pixel a forcément de nombreux voisins de la même classe que lui. Il est donc possible d'enrichir l'information de distance à l'aide du comptage des voisins de même classe. Nous avons donc modifié le calcul de la fonction d'appartenance à une classe de façon à relâcher l'attache aux données au profit des critères spatiaux. Ceci revient à modifier le calcul de la distance des classes pour mieux gérer la partie incertaine.

Dans la suite, nous allons exprimer le facteur d'appartenance à une classe comme un potentiel qui se manipule plus facilement qu'une probabilité. Celui-ci comporte deux termes, un exprimant l'appartenance à une classe et l'autre l'appartenance à l'autre classe. Soit $I\left(x,y\right)$ l'intensité du pixel de position x,y, nous écrivons le potentiel d'appartenir à la classe c relativement à la classe c' :

$$P(x,y,c,c') = \alpha\left(I\left(x,y\right),c\right) - \alpha\left(I\left(x,y\right),c'\right) + e\left(x,y,c\right) - e\left(x,y,c'\right)$$

où $\alpha\left(I,c\right)$ représente le potentiel d'appartenance à la classe c pour l'intensité I et $e(x,y,c)$ l'énergie potentielle fournie par le voisinage. La difficulté de mise au point d'une telle classification réside dans le choix de ces termes et dans leur équilibrage qui est toujours délicat. Pour la classe "fond" les contraintes sont :

- la nature du bruit fond, composé de chatoiement et de fragments de sonde hybridés et la connexité des spots, qui induit qu'ils sont sous-représentés en surface, nous a conduit à choisir des potentiels symétriques. En effet, nos objectifs pour la classe fond sont d'éliminer des pixels isolés de fortes valeurs classés à tort comme appartenant à un spot ce qui suppose d'utiliser un critère spatial fort.
- nous souhaitons que, pour des zones de moyenne proche de celle du fond mais incluse dans le spot, la classification les considère comme appartenant aux spots. Ceci implique une attache aux données forte autour de la moyenne du fond mais qui décroît rapidement.

Pour la classe "spot", les contraintes sont :

- les valeurs proches de la moyenne doivent être affectées à la classe "spot" sous réserve de voisinage.
- les pixels proches de la moyenne du fond doivent être rejetés sauf en frontière de façon à corriger les effets de sur-représentation surfacique.
- l'attache aux données doit croître au fur et à mesure qu'on s'éloigne de la moyenne du fond et que l'on s'approche de la moyenne du spot.

À partir de ces deux jeux de contraintes, nous avons fabriqué le potentiel linéaire par morceaux $\alpha\left(I,c\right)$ défini par les formules suivantes :

si $c = F$

$$\begin{cases} I < \overline{F} & \alpha(I,F) = 1 \\ \overline{F} < I < \frac{\overline{F}+\overline{S}}{2} & \alpha(I,F) = \frac{3}{8}\left(\frac{\frac{\overline{F}+\overline{S}}{2} - I}{\overline{S}-\overline{F}}\right) + \frac{1}{4} \\ \frac{\overline{F}+\overline{S}}{2} < I & \alpha(I,F) = 0 \end{cases}$$

si $c = S$

$$\begin{cases} I < \overline{F} & \alpha(I,S) = 0 \\ \overline{F} < I < \frac{\overline{F}+\overline{S}}{2} & \alpha(I,S) = \frac{1}{4} \\ \frac{\overline{F}+\overline{S}}{2} < I < \overline{S} & \alpha(I,F) = \frac{3}{8}\left(\frac{I - \frac{\overline{F}+\overline{S}}{2}}{\overline{S}-\overline{F}}\right) + \frac{1}{4} \\ \overline{S} < I & \alpha(I,F) = 1 \end{cases}$$

où $\overline{S}$ et $\_overline F$ sont les moyennes estimées du spot et du fond. Ce potentiel α est représenté graphiquement en fonction de l'intensité I sur la figure 3.

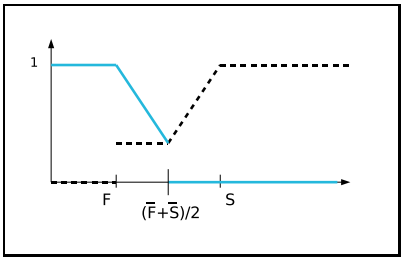


FIGURE 3. Sur cette figure nous voyons le potentiel α pour la classe "fond" en clair et pour la classe "spot" en tireté.

Le terme $e(x, y, c)$ est obtenu en comptant le nombre de voisins de la classe c se trouvant dans un voisinage en huit connexités, c'est-à-dire limité aux huit pixels entourant le spot. Ce choix est gouverné par le fait que les spots n'ont pas de direction privilégier et sont de taille réduite ce qui nous conduit à limité les effets à longue distance du processus de régularisation spatiale. Avec cet ensemble de contraintes, il faut maintenant fabriquer l'image de classe c minimisant la fonctionnelle P en tout point de l'image originale. Pour celà, certaines propriétés de notre fonctionnelle sont précieuses et notamment le fait que le processus de régularisation spatiale en huit connexités nous permet d'affirmer que nous sommes dans un cadre markovien [6]. Cela nous indique que le problème d'optimisation a une solution et que par conséquent, un schéma itératif de recherche par minimisations successives convergera vers cette solution.Cela implique également qu'il est possible de segmenter l'image spot par spot sans changer le résultat. Nous pouvons donc découper arbitrairement en morceaux l'image sans changer l'aspect de la solution.

Le problème principal que pose, par contre, un tel schéma est sa complexité. Une recherche directe du minimum absolu serait beaucoup trop longue, en particulier dans le cadre d'un logiciel interactif. Nous procèderons donc ici par recuit simulé.

Notre algorithme se déroule donc en trois phases : une phase d'initialisation, une phase de classification itérée et une phase de nettoyage éventuel du résultat.

La phase d'initialisation consiste à découper l'image en sous-imagettes autour des spots et estimer par la méthode décrite dans la partie 4 la moyenne du fond et du spot pour chaque imagette. Ensuite pour chaque site, on tire un masque de classe suivant un processus Poissonnien.

La phase de classification consiste pour chaque imagette à prendre un pixel et calculer le potentiel pour la classe courante donnée par l'image de classe. Si ce potentiel est négatif, on change la classe, sinon on calcule l'expression $x = log(\frac{P}{t})$ et on tire aléatoirement un nombre x' suivant une loi uniforme entre 0 et 1 et si $x' > x$, nous changeons la classe du pixel bien que le potentiel indique que le pixel est plutôt bien classé. Ce choix permet de ressortir des minima locaux de potentiel sans devoir essayer toutes les combinaisons possibles. Nous passons ensuite aux pixels suivants. Puis nous réitérons le procédé avec $t_n = 0.95 * t_{n+1}$. La classification est finie quand le nombre de pixels qui ont été modifiés est inférieur à 1 pour 1000. La phase de nettoyage, parfois appelée trempe, consiste à attribuer les éventuels pixels isolés à la classe qui les entoure par simple comptage.

Ces algorithmes demandent beaucoup de calculs mais les propriétés des champs de Markov nous permettent de paralléliser sans risque le calcul des différentes imagettes. Celà raccourcit significativement la durée du traitement sur les machines dual-core ou multiprocesseurs, et l'on obtient une classification en quelques secondes pour des images courantes.

La partie suivante présente nos résultats.

5 Résultats de la segmentation markovienne.

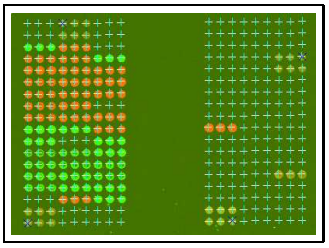


FIGURE 4. Image simple de biopuces.

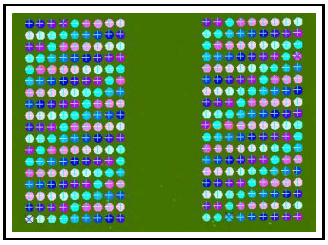


FIGURE 5. Image de la figure 4 segmentée à l'aide la technique du cercle fixe.

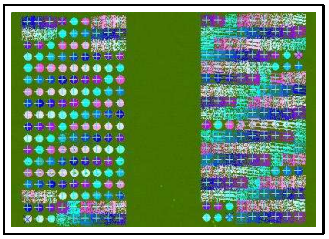


FIGURE 6. Image de la figure 4 segmentée à l'aide d'une classification par nuées dynamiques.

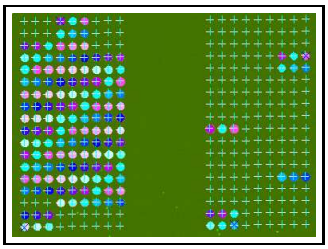


FIGURE 7. Image de la figure 4 segmentée à l'aide d'une classification par notre technique.

Nous ferons la présentation de nos résultats sur deux séries d'images, une série facile, telle celle de la figure 4, et une série plus difficile, telle celle de la figure 8, pour montrer les avantages et les limites de notre approche. Comme cela est dit précédemment, une bonne segmentation doit comporter au moins trois qualités : ne pas segmenter les spots non hybridés, se comporter correctement en présence d'artefacts, segmenter au plus juste les spots hybridés. Nous présentons nos résultats en référence à ceux obtenue par nuée dynamique et par les cercles fixes car se sont deux type de classification utilisé dans les outils professionnels dont dispose nos partenaires biologiste.

Pour juger de la première qualité, nous ne pouvons pas considérer directement le nombre de spots hybridés détectés. Il convient donc de calculer la probabilité de se tromper qu'a un algorithme quand il détecte un spot. Le taux de détection que nous obtenons est alors de 99 % des bons spots et 99,9 % des spots non hybridés ne sont pas segmentés. Ces résultats sont bien évidemment meilleurs que ceux des deux autres méthodes présentées.

Le comportement en présence d'artefacts ne peut se juger que visuellement car il n'existe pas de recensement exhaustif des défauts possibles dans ce type d'image. Les deux types les plus gênants sont liés au bruit de fond et aux arrachements de matière sur les sondes qui produisent des zones très brillantes après hybridation. Sur la figure 10, nous présentons le résultat de notre traitement dans le cas d'une image plus difficile (figure 8). Sur ces données, nous voyons plusieurs cas de pollution de sonde provenant d'arrachements d'autres spots avec différentes tailles et proximités de la sonde polluée. Nous constatons alors que le comportement de notre méthode est globalement satisfaisant, car les zones de pollution ne sont pas segmentées dès qu'elles atteignent une taille significative.

Pour ce qui est de juger de l'aptitude d'une segmentation à intégrer un minimum de pixels de fond dans les spots produits, la seule méthodologie valable consiste à tracer à l'aide d'un expert un grand nombre de spots à la main puis à calculer les matrices de confusion entre les segmentations et le masque manuel [12]. Cette opération est longue et fastidieuse si nous voulons traiter un grand nombre de cas. Nous avons donc procédé en tirant au sort 20 spots dans 32 lames différentes ce qui permet de balayer un grand nombre de cas sans dépenser trop de temps. Parmi ces spots, 15 par puce ont été

utilisés pour calculer la matrice de confusion et pour vérifier sa robustesse. Ces mesures montrent un gain de 15 à 18 % de bonne détection.

Notre méthode a donc les trois qualités qui correspondent à notre cahier des charges. Nous remarquons de plus que l'adaptation de notre segmentation aux petites déformations de grille, non compensées par les deux techniques de calages présentées précédemment, est excellente. Ceci montre bien la supériorité des méthodes à base de classification pour la détection de la forme des spots sur les images de biopuces.

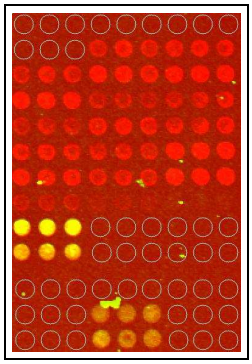


FIGURE 8. Image brut, les spots non remplit sont encerclés.

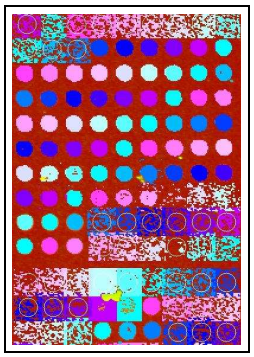


FIGURE 9. Résultat d'une classification par nuées dynamiques.

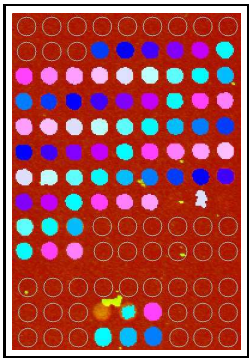


FIGURE 10. Résultat de notre traitement sur l'image de la figure 8.

6 Conclusions et perspectives

Le travail présenté ici atteint les buts que nous nous sommes fixés, car notre méthode permet de traiter des biopuces provenant de constructeurs quelconques tout en segmentant de façon robuste les forts et faibles contrastes. Notre méthode exploite au mieux les propriétés des différentes données disponibles (grille, images, ...) et elle utilise au minimum les interventions d'opérateurs.

Notre approche va être poursuivie suivant deux axes visant toujours l'amélioration de la qualité des mesures d'expression. Dans un autre aspect, nous travaillons sur les techniques de calibrage rouge vert afin d'améliorer cette fois la qualité de comparaison entre deux populations de gènes qui est souvent le but ultime de l'utilisation de biopuces.

Nous tenons à remercier le professeur Pierre Peyret de l'Université Blaise Pascal qui nous a fourni les images nécessaires à ce travail et Anne Moné qui à participé à l'expertise des résultat.

Références

1. G. Antoniol and M. Ceccarelli. Microarray image gridding with stochastic search based approaches. *Image and Vision Computing*, 25(2) :155 – 163, 2007.
2. Y. Balagurunathan, E. Dougherty, and Y. Chen. Simulation of cdna microarrays via a parameterized random signal mod el. *Joural of Biomediacl Optics*, 7(3) :507–523, July 2002.
3. V. Barra. Robust segmentation and analysis of dna microarray spots using an adaptative split and merge algorithm. *Computer Methods and Programs in Biomedicine*, 81(2) :174–180, Feb. 2006.
4. C. S. Brown, P. Goodwin, and P. K. Sorger. Image metrics in the statistical analysis of dna microarray data. *PNAS*, 96(16) :8944–8949, July 2001.
5. O. Demirkaya, M. H. Asyali, and M. M. Shoukri. Segmentation of cdna microarray spots using markov random field modeling. *Bioinformatics*, 21(13) :2994–3000, 2005.
6. X. Descombes. *Champs markoviens en analyse d'images*. 93 E 026, ENST, Paris-France, 1993.
7. M. B. Eisen and P. O. Brown. Dna arrays for analysis of gene expression. *Methodes in enzimology*, 303 :179–205, 1999.
8. J. W. Goodman. Some fundamental properties of speckle. *J. Opt. Soc. Am.*, 66(11) :1145–1150, 1976.
9. R. Hirata, J. Barrera, R. F. Hashimoto, D. O. Dantas, and G. H. Esteves. Segmentation of microarray images by mathematical. *Real-Time Imaging*, 8(6) :491–505, Dec. 2002.
10. M. Katzer and F. Kummert. A markov random field model of microarray gridding. In *in Proc. ACM Symposium on Applied Computing (SAC*, pages 72–77. ACM Press, 2003.
11. M. Katzer, F. Kummert, and G. Sagerer. Robust automatic microarray image analysis. In *In BREW Bioinformatics Research and Education Workshop*, 2002.
12. M. G. Kendall and A. Stuart. *The Advanced Theory Of Statistics*, volume 1. Charles Griffin and Compagny -Limited London and High Wycombe, third edition, 1952.
13. A. Kuklin, S. Shams, and S. Shah. Automation in microarray image analysis with autogene(tm). *Journal of the Association for Laboratory Automation*, 5(5) :67 – 70, 2000.
14. A. W.-C. Liew, H. Yan, and M. Yang. Robust adaptive spot segmentation of dna microarray images. *Pattern Recognition*, 36(5) :1251 – 1254, 2003.
15. M. McGoven and R. Fayek. Advantages of laser confocal microarray scanning. *Microarray Image Anlysis-Nuts and Bolts*, pages 51–68, 2002.
16. N. Pal and S. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26 :1277–1294, 1993.
17. A. Petrov and S. Shams. Microarray image processing and quality control. *Journal of VLSI Signal Processing Systems*, 38(3) :211–226, Nov. 2004.
18. A. Pretrov, S. Sha, S. Draghici, and S. Shams. Microarray image processing and quality control,. *Microarray Image Analysis-Nuts & Bolts*, (6) :99–130, 2002.
19. C. K. Wierling, M. Steinfath, T. Elge, S. Shculzen-Kremer, P. Aanstad, M. Clark, H. Lehrach, and R. Herwig. Simulation of dna hybridization experiments and evaluation of critical parameters during subsequent image and data analysis. *BMC bioinformatics*, 3(29) :17, Oct. 2002.
20. Y. H. Yang, M. J. Buckley, S. Dudoit, and T. P. Speed. Comparison of methods for image analysis on cdna microarray data. *Journal of Computational and Graphical Statistics*, 11 :108–136, 2002.
21. C. Yidong, R. Edward, and all. Ratio-based decisons and the quantitative analysis of cDNA microrarray images. *Journal of biomedical optics*, 2(4) :364–374, Oct. 1997.

Pseudo-CT basée sur l'IRM pour la correction d'atténuation

Chaibi Hassen[1], Nourine Rachid[2]

[1]: Labo IVARM – USTO Oran, chaibih@yahoo.fr.
[2]:Laboratoire d'Informatique et des Technologies de L'Information d'Oran, nourinerachid@lycos.com

Résumé : La correction d'atténuation (CA) est une étape primordiale pour la reconstruction des images PET. Ces dernières années de nouvelles recherches s'orientent vers l'exploitation de l'information IRM plutôt que CT pour établir cette carte d'atténuation, vu la richesse de l'information en IRM, ainsi que la sécurité de son utilisation. Néanmoins, l'utilisation de l'IRM pose le problème de la faible discrimination entre certains types de tissus. Dans cet article, nous proposons alors d'établir à partir d'une IRM une pseudo-CT, qui sera utilisée ultérieurement pour la CA. Nous considérons le problème comme une régression et nous proposons d'utiliser un perceptron à multicouches (PMC) pour le résoudre. Vu la complexité anatomique du corps humain, nous utilisant plusieurs PMC tel que chacun soit dédié à une zone spécifique du corps. Les résultats obtenus sont acceptables.

Mots clé: PET/IRM, Correction d'atténuation, Réseau de neurones

1 Introduction

L'imagerie multi-modalité combinant différents scanners, tels que PET/CT, SPECT/CT et plus récemment PET/IRM commence à révolutionner la médecine nucléaire. Ainsi, ces dernières années, les combinaisons PET/CT et SPECT/CT ont facilité la transition de la phase de recherches théorique à la pratique clinique [1].

Seulement, l'utilisation des images CT pose quelques inconvénients, tel que le faible contraste entre les tissus, l'ajout de radiation significative aux malades. Ainsi, de nombreuses recherches dans ce domaine se sont orientées récemment sur les systèmes hybrides PET/IRM [2,6]. Même si pour l'instant le développement de ces systèmes pose quelques problèmes, leur utilisation clinique n'est qu'une question de temps [10,11].

Le développement de ce type d'imagerie multimodal PET/IRM est motivé par divers facteurs. D'abord, les images IRM sont utilisées pour obtenir des images anatomiques et structurelles avec une meilleure résolution spatiale que celle offerte par l'imagerie CT. L'IRM offre aussi d'excellents contrastes entre les matières blanches et grises et tient compte de la formation d'image fonctionnelle dans des études du cerveau. Enfin, l'IRM n'utilise pas de radiation, ce qui permet son utilisation sans restriction dans des études périodiques, en pédiatrie par exemple et en d'autres situations où l'exposition à la radiation est un souci.

L'utilisation de l'imagerie PET nécessite, pour être quantitative, que soit pris en compte les effets d'atténuation dus au milieu traversé par les photons, avant leur détection. Ce problème a suscite de très nombreux travaux proposant différentes méthodes de correction d'atténuation (CA) [7, 12]. Les méthodes, proposant des cartes d'atténuation sont généralement regroupées en deux classes principales. La première classe inclut des techniques de correction sans transmission basées sur des méthodes calculées, et modèles statistiques pour l'évaluation des distributions d'atténuation. La deuxième classe inclut des méthodes basées sur les scanners de transmission comprenant une source extérieure de radionucléide ou un CT. D'autre part, une nouvelle classe de méthodes se développe actuellement, basée sur l'exploitation de l'imagerie IRM [1].

Dans le cas de la combinaison PET/CT, la correction d'atténuation est direct : les coefficients d'atténuations sont calculés à partir des images de transmission CT, où les différents tissus sont assez bien discriminés (bonne séparation entre les os et les autres tissus 'non-osseuse'). Malheureusement, ceci ne fonctionne pas aussi bien pour le PET/IRM puisque des structures anatomiques différentes telles que l'os et l'air ont les valeurs semblables sur les IRM. De plus, les images IRM seules n'apportent pas suffisamment d'information sur les coefficients d'atténuations des tissus [1]. Par conséquent, il est nécessaire d'utiliser d'autres méthodes pour estimer des CA à partir des images IRM.

Nous présentons dans la section suivante quelques unes de ces méthodes proposées par différents chercheurs. Nous d'écrirons ensuite l'approche que nous proposons et dont l'idée de base est d'estimer à partir d'une image IRM une image CT (que l'on appellera pseudo-CT), qui sera alors utilisée ultérieurement pour l'établissement de la CA. Différents problèmes sont soulevés dans cette approche que nous présenterons dans la section résultats et discutions.

2 Etat de l'art

Le signal IRM (grandeur et phase) d'un voxel individuel est lié à la densité de proton, et non pas la densité d'électron qui est nécessaire pour le calcule de la carte d'atténuation. Par exemple, dans la plupart des séquences standards IRM, l'air, l'os, le support du patient et les anneaux ne produisent aucun signal, tandis que leurs coefficients d'atténuation sont différents. Cela explique la difficulté à estimer

les coefficients d'atténuation directement à partir des IRM [3]. Néanmoins de nombreuses techniques sont proposées dans la littérature que l'on peut regrouper en deux classes : les méthodes qui se focalisent sur la partie crânienne (le cerveau), et les autres qui tentent d'étudier le corps entier [4].

L'approche par technique de segmentation du cerveau à été la première utilisée par Goff-Rougetet et al, qui a proposé une méthode pour calculer des coefficients de (CA) des images PET [4]. Elle est basée sur un alignement des images IRM aux images de transmission PET. Les images IRM alignées sont alors segmentées dans trois classes (tissu, cerveau, os). L'air est considéré seulement en dehors du patient. Les valeurs linéaires de coefficient d'atténuation (μ) à 511 kev sont alors assignées à ces classes de tissu. El Fakhri et al ont également mentionné une méthode de CA à partir d'image IRM, mais n'ont pas fourni d'autres détails de leur implémentation ou une évaluation des performances. Une méthode alternative a été suggérée par Zaidi et al [6]. Les auteurs ont utilisé une méthode de segmentation basée sur la logique floue, les images IRM sont segmentées dans cinq classes de tissu auxquelles on a assignées des coefficients d'atténuation à 511 kev .

L'approche Atlas est une alternative aux procédures de segmentation pour la CA basée sur l'IRM. Un atlas se compose typiquement d'une image IRM modèle ainsi qu'une image correspondante d'étiquette d'atténuation. L'image IRM modèle peut être obtenue comme une moyenne d'images alignées de plusieurs patients. L'image d'étiquette pourrait représenter une segmentation dans différentes classes de tissu (par exemple air, os et tissu mou) ou une carte d'atténuation alignée à partir d'un scanner de transmission PET ou un scanner CT. L'image IRM ATLAS est alignée à l'image IRM du patient. En appliquant la même transformation spatiale à l'image d'atténuation d'atlas une carte correspondante d'atténuation est produite (spécifique au patient).

Les approches basées sur un Atlas ont été présentées par Kops et Herzog [4] et Hofmann et autres [1]. Kops et le Herzog produisent un modèle des images de transmission PET des données de 10 patients qui sont recalées au modèle de transmission PET dans SPM2. Le modèle IRM dans SPM2 (qui est déjà aligné avec le modèle de transmission PET) est normalisée à l'image IRM du patient. La transformation obtenue est alors appliquée à l'image d'atténuation du modèle pour produire une image d'atténuation pour ce patient. Le même groupe a également présenté une étude basé sur la segmentation et le recalage de l'image IRM à l'image de mesure de transmission PET.

Hofmann et al ont suggéré une approche révisée basée sur un Atlas, où ils essayent de créer des pseudo-CT pour la correction d'atténuation [1]. Les auteurs utilisent un ensemble de volumes IRM-CT alignés de 17 patients. Chacun des 17 volumes IRM disponibles est aligné à l'image IRM du patient, par la suite les vecteurs d'alignements sont appliqués aux volumes correspondants d'image CT produisant 17 ensembles d'image de CT. Dans la 2eme étape un module de reconnaissance de forme est employé pour identifier l'image IRM de l'atlas la plus proche de l'image IRM du patient. L'image CT correspondante à cette image IRM trouvé est la pseudo-CT spécifique au patient.

En raison du manque de systèmes de simulation IRM/PET pour les études du corps entier, les applications extra-crâniennes sont rares. Beyer et al [5] installent une boîte à outils qui facilite la contre-vérification de carte d'atténuation basée sur l'IRM et celles basées sur une image CT. Ils ont étudié dix patients qui ont subi des balayages courants de torse avec les bras vers le haut sur un tomographe combiné au PET (CT/PET) et des balayages complémentaires IRM. D'abord, les images IRM étaient alignées aux images CT en utilisant un algorithme d'alignement non-linéaire. En second lieu, la distribution d'intensité de valeurs de voxels IRM a été assortie à celle de l'image CT correspondante. La transformation d'intensité IRM-CT a été exécutée dans un processus en trois étapes basé sur un algorithme de mise en correspondance d'histogramme.

Bien que principalement utilisées pour la formation d'image du cerveau, des méthodes basées sur atlas peuvent également être appliquées aux images du corps entier. Cependant, la variabilité anatomique est haute et il est peu probable qu'une transformation spatiale générale capture toute la variation entre un modèle et une anatomie spécifique (du patient).

Hofmann et al [1] estime que son approche pourrait être utilisée pour la carte d'atténuation des images extra-crâniennes. La validation de son approche a été effectuée sur deux ensembles de données du corps entier d'un lapin.

Z. Hu et al aborde dans leur étude [13] la CA sur les images du corps entier. Ils utilisent un algorithme de segmentation pour distinguer 4 classes biologiques (l'air, les poumons, les tissus mous et des os). Outre la segmentation d'image, cette étude présente une technique de compensation qui a été élaborée pour réduire les artefacts ou les erreurs de quantification découlant de troncature dans l'image IRM du corps entier (autour des bras).

3 Approche proposée

Le travail que nous présentons dans cet article entre dans le cadre globale de développement de système combiné PET/IRM comme alternative au system PET/CT, vu les propriétés importantes en termes de contraste et de résolution des images IRM. Néanmoins, il est établi que l'établissement d'une carte d'atténuation est plus simple en images CT qu'en IRM. Notre idée est donc de proposer un system permettant de prédire l'image CT correspondante à une IRM, pour ensuite établir la carte d'atténuation nécessaire à la correction de l'image PET.

Nous considérons donc l'établissement d'une image pseudo-CT à partir d'une IRM comme un problème de régression où nous voudrions expliquer les valeurs CT à travers celles de l'IRM. Seulement, plutôt que de considérer que cette régression est linéaire tel que proposé par Matthias Hofmann [1], nous supposons qu'elle est non linéaire. C'est pourquoi nous avons opté pour l'utilisation d'un réseau de neurone pour établir cette régression.

Choix du type de réseau de Neurone

Les réseaux de neurones possèdent d'indéniables qualités lorsque l'absence de linéarité et/ou le nombre de variables explicatives rendent les modèles statistiques traditionnelles inutilisables. La propriété d'approximation universelle de réseaux de neurones en fait des outils performants pour la régression non linéaire. On a montré que toute fonction continue d'un compact de $\mathbb{R}p$ dans $\mathbb{R}q$ peut être approchée avec une précision arbitraire par un réseau à une couche cachée en adaptant le nombre de neurones [8]. De ce fait on a utilisé un perceptron multicouche « feed-forward backpropagation network ».

Dans tout problème de régression il est nécessaire de préciser en plus de la variable à expliquer, les variables explicatives. D'autre part, estimer la valeur d'un pixel en CT à travers la valeur de son correspondant dans l'IRM étant impossible, nous avons opté pour l'exploitation du voisinage de ce dernier. De plus, plutôt que d'utiliser les valeurs du voisinage, nous avons choisi d'utiliser la variation d'intensité autour du pixel. Cette variation est résumée par deux paramètres, la moyenne et l'écart-type, calculés sur différents voisinages du pixel traité. Comme voisinages du pixel dans l'IRM, nous avons choisie 8 voisinages où le pixel est un coin et deux voisinages centrés autour de celui-ci, tel que le montre la figure 1.

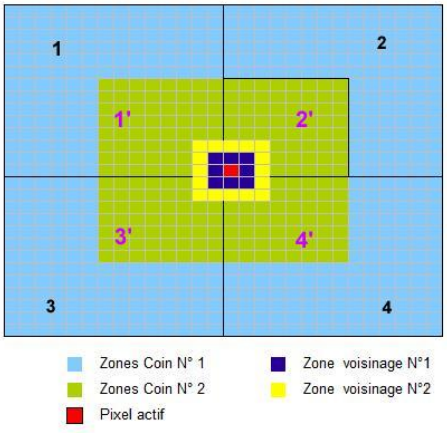


Figure 1 : Répartition des fenêtres de voisinage utilisées

Après avoir testé différents modèles de réseaux de neurones sur les différentes parties du corps, nous sommes arrivé à la conclusion suivante : la forte variation anatomiques entre les différentes zones du corps, rend difficile l'établissement d'un seul réseau de neurone, globale pour tout le corps. Pour y remédier, nous proposons de spécifier sur tout le corps des zones où la complexité anatomique est faible, c'est-à-dire qu'il y a une certaine homogénéité entre les coupes

de chaque zone. L'idée est alors d'utiliser des réseaux de neurones spécifiques sur chacune de ces zones. Ainsi, sur la base d'une étude préliminaire nous avons divisé horizontalement la tête en quatre zones que nous supposons homogènes, tel que le montre la figure 2. Le principe est le même pour l'ensemble du corps humain.

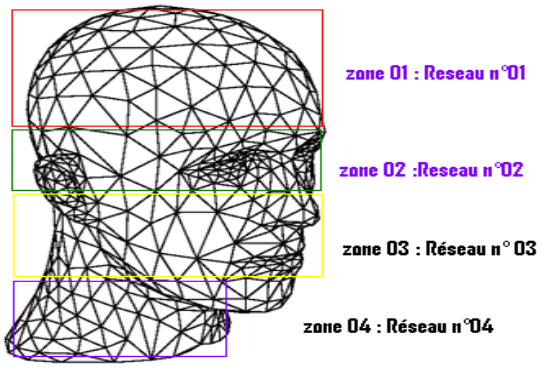


Figure 2 : Répartition des réseaux sur la tête.

D'autres part et afin d'augmenter la robustesse de prédiction des pseudo-valeurs CT, nous proposons de subdiviser chaque coupe (image) d'une zone anatomique en quatre sous images, auxquelles on dédiera des perceptrons spécifiques, comme l'indique la figure 3. Chaque zone homogène du corps sera alors traitée par quatre perceptrons identiques en architecture mais différents en termes d'apprentissages. De plus, ce choix de quatre zones permet de facilité l'utilisation du modèle, car il suffit de centrer les images IRM du patient, pour que chaque perceptron se voit confie la partie adéquate (sur laquelle il a fait l'apprentissage). Cette opération se limite à placer le centre du volume IRM du Patient sur les même coordonné que le centre du volume IRM d'apprentissage. Vus la variabilité anatomique inter-sujet un autre choix positionnerait mal les perceptrons, et on risque de donné à un perceptron une partie à traiter qu'il na pas étudié.

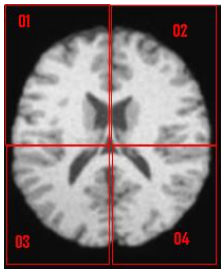


Figure 3 : Répartition des 4 perceptrons sur 4 sous images d'une coupe

Architecture du PMC

Le Réseau PMC que nous proposons doit estimer la valeur CT d'un pixel en se basant sur les variations des valeurs des voisinages autour du pixel correspondant dans l'IRM. Tel que nous l'avons cité précédemment nous avons sélectionné 10 voisinages autour du pixel traité. Ainsi, en calculant la variance et la moyenne de chaque voisinage, nous avons avec la valeur du pixel 21 entrées pour le PMC. Deux couches cachées s'en suivent l'une de 13 neurones, la seconde de 5 neurones et enfin un neurone en sortie, tel que le montre la figure 4.

Le problème avec les réseaux de neurones c'est le choix de l'architecture (nombre de couches et de neurones), Vue que dans notre problème il ya d'autre paramètres à optimiser (zone couverte, taille des fenêtres…), nous avons procédé comme suite :

- Premièrement on devra régler le problème de l'architecture du réseau et utiliser la configuration trouvé comme une configuration idéale pour tous les autres.
- Après quoi on cherche à optimiser les autres paramètres.

Pour résoudre le premier problème nous avons fait une étude préliminaire sur un nombre de coupe réduit, où nous avons testé plusieurs configurations pour choisir le nombre de couche et le nombre de neurones. Les critères étaient de minimiser le taux d'erreur d'apprentissage et la différence statistique entre l'image CT réel et pseudo-CT. Une fois le problème de l'architecture réglé, nous avons entamé une autre étude pour définir les zones couverts par chaque ensemble de perceptrons.

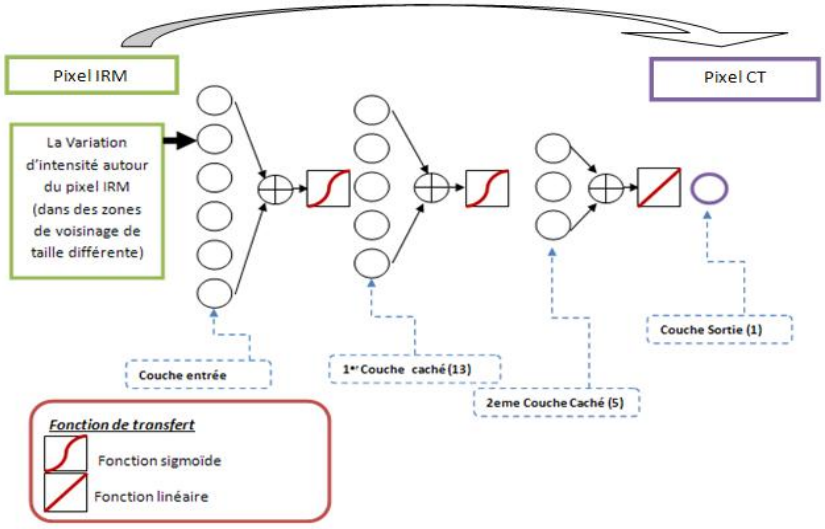


Figure 4 : Architecture du réseau PMC

L'utilisation d'un réseau de neurones passe par deux phase, la 1ere est l'apprentissage et la deuxième l'exploitation ou l'utilisation de ce réseaux.

A- Apprentissage du PMC :

L'apprentissage d'un réseau de neurones étant supervisé, il faut pouvoir définir des exemples d'apprentissage, ainsi que des contre-exemples (étape de collecte de données), suivi de l'étape d'apprentissage.

A.1 La Collecte des données :

Elle se présente en deux opérations :

- Extraction de formes : c'est un prétraitement qui consiste à ignorer l'arrière plan de l'image IRM et de ne traiter que les pixels qui forment la partie anatomique étudiée.
- Balayage et définition des vecteurs : le parcoure de la zone étudiée ce fait ligne par ligne, et pour tout pixel IRM on calcul la variation de son voisinage (selon la répartition déjà vue). Ainsi chaque pixel est représenté par un vecteur de 21 valeurs. Le parcoure de l'image CT correspondante s'effectue en même temps, sauf qu'on ne retient que la valeur du pixel CT.

A.2 L'Apprentissage :

La normalisation des données est la première étape d'apprentissage, elle consiste à définir les valeurs d'entrées et de sorties du perceptron dans l'intervalle [0 1]. La matrice des variations (Résultats des images IRM) représente l'entrée, et la matrice des valeurs CT représente la sortie du PMC. Comme algorithme d'apprentissage nous avons utilisé l'algorithme de Levenberg-Marquardt.

B- Exploitation Et utilisation du PMC

Pour utiliser des PMC il faut passer par le prétraitement précédant à savoir l'extraction de formes. Le balayage de l'image est réalisé ligne par ligne, et pour chaque pixel on calcule le vecteur des variations. Ce vecteur est passé comme entrée au PMC correspondent (qui couvre la zone du pixel IRM), et c'est ce dernier qui va donner comme résultat la valeur du pixel de l'image pseudo-CT.

4 Résultats et validation

Notre modèle a été testé en premier lieu sur deux volumes IRM et CT du même patient parfaitement alignées. Une partie des images est utilisée pour l'apprentissage et l'autre partie pour le teste. Pour jugé de la qualité des résultats, on a tracé le profil horizontal des images (réel et pseudo-CT) que nous juxtaposons. La différence entre les deux courbes n'est statistiquement pas très significative, ce que dénote d'une bonne estimation. Le modèle à pus prédire des zones d'Os et de l'Air dans

les images de la tète, ce qui était un grand défaut des modèles déjà testé (l'air et l'os on la même intensité du signal IRM) (figure 5 et 7).

Pour le test du modèle sur la partie hors-crânienne, et vue le manque de données (images IRM et CT) du corps entier on a utilisé des images démonstratives du projet « Visible Korean Human» [9]. Le projet est une base de données qui contienne des images IRM et CT d'un cadavre. Le modèle a était testé sur la partie abdominale (figure 9), et il a donné des résultats acceptable (vue la qualité des images IRM de la base d'apprentissage qui sont trop bruité). Les images qui suivent représentent les images originales (CT) et les images résultat du modèle (les pseudos CT) ainsi que les courbes du profil horizontal.

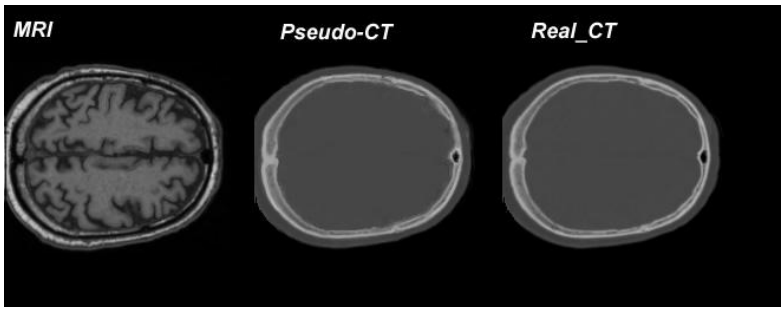


Figure 5 : (1) Image IRM, (2) Image Résultat (Pseudo-CT) correspondante, (3) Image CT Originale.

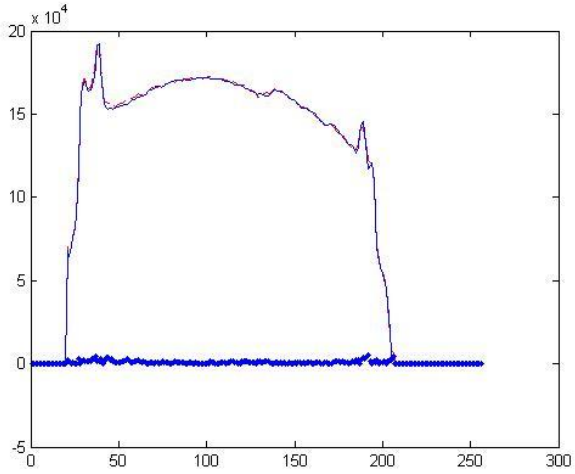


Figure 6: Profil horizontal des résultats de la figure 5

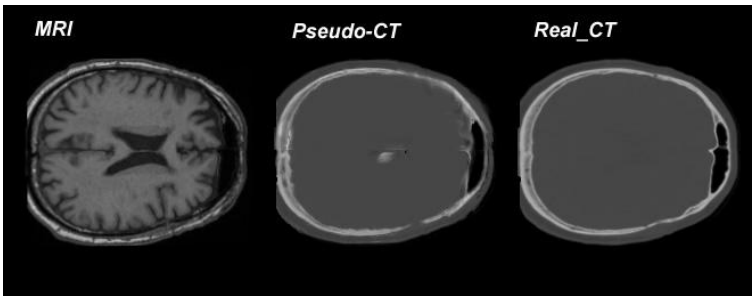


Figure 7 : (1) Image IRM, (2) Image Résultat (Pseudo-CT) correspondante, (3) Image CT Originale.

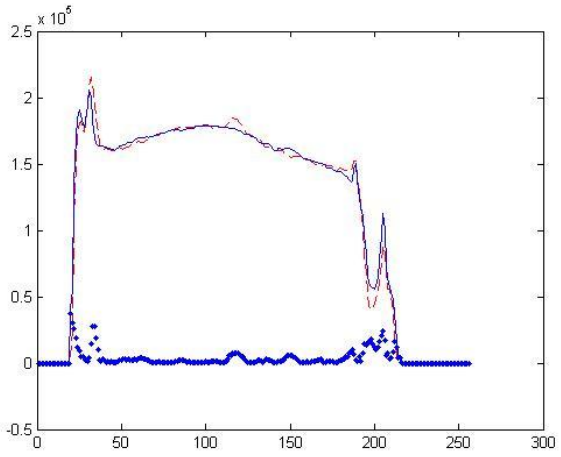


Figure 8: Profil horizontal des résultats de la figure 7.

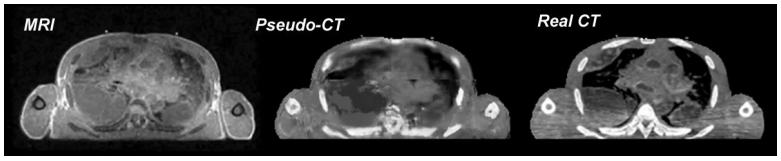


Figure 9 : (1) Image IRM, (2) Image Résultat (Pseudo-CT) correspondante, (3) Image CT Originale.

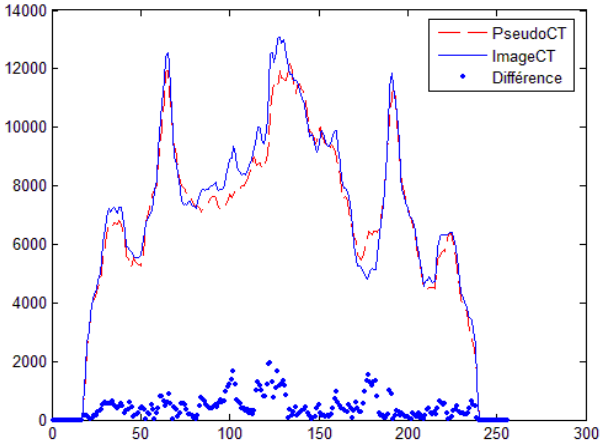


Figure 10: Profil horizontal des résultats de la figure 9.

Le profil horizontal d'une image c'est la somme des valeurs des pixels de la même colonne, ainsi le résultat est un vecteur qui représente une courbe. On trouve dans la littérature aussi la définition du profil horizontal comme étant le vecteur qui représente une ligne au milieu dans l'image. Dans notre cas nous avons opté pour la première définition.

On remarque sur les images de la partie crânienne que visuellement il n'existe pas une grande différence (figures 5 et 7). Les courbes de profils confirment cette constatation en indiquant une faire différence statistique (figures 6 et 8). On peut dire que les résultats sont acceptables.

Dans la partie hors-crânienne les résultats sont moins réussis (figure 10) On pense que ceci revient à la mauvaise qualité des images de la base d'apprentissage, mais malgré cela on peut observer que le modèle a pu prédire l'ensemble des tissus (fig. 9)

Il faut noter, qu'il a fallu configurer différemment les perceptrons selon la partie traitée (crânienne et hors-crânienne). Cette configuration passe par la définition de la taille des fenêtres de variation.

5 Perspective et conclusion:

Conscient qu'une évaluation de notre modèle sur des volumes d'image est plus que nécessaire, mais le problème principal c'est le manque de données du même patient (dans les 3 modalités IRM, CT et PET). Dans les prochains

travaux on va utiliser des volumes entiers, afin de faire une évaluation clinique qui passera par un expert en imagerie nucléaire.

Les premiers résultats montrent la puissance de notre modèle à prédire des Pseudo-CT. La méthode proposée ne se fonde pas sur l'information anatomique locale et semble donc prometteuse pour des applications du corps entier.

6 Référence

[1] Hofmann et al. MRI-Based Attenuation Correction for PET / MRI: A Novel Approach Combining Pattern Recognition and Atlas Registration .The Journal Of Nuclear Medicine. Volume 49 - No. 11. November 2008.

[2] Catana et al. Simultaneous Acquisition of Multislice PET and MR Images: Initial Results with a MR-Compatible PET Scanner. The Journal Of Nuclear Medicine . Volume 47- No. 12 .December 2006.

[3] Habib Zaidi. Is MR-guided Attenuation Correction a Viable Option for Dual-Modality PET / MR Imaging. Radiology: Volume 244 - No 3.September 2007.

[4] Hofmann et al. Towards quantitative PET/MRI; A Review of MR-based attenuation correction techniques. J Nucl Med Mol Imaging .93-104. Volume 36 (supplement 1).Mars 2009.

[5] T.Beyer et al. MR-based attenuation correction for torso-PET/MR imaging: pitfalls in mapping MR to CT data. J Nucl Med Mol Imaging. 1142-1146. Volume 35, Nummero 6. juin 2008.

[6] Habib Zaidi,Marie-Louise,Daniel O.Slosman. Magnetic resonance imaging-guided attenuation and scatter corrections in three-dimensional brain positron emission tomography . Med. Phys. 30-5. May 2003.

[7] I. Buvat et al. Comparison of different protocols for attenuation correction in cardiac SPECT imaging – Review. ACOMEN. Volume 4, No 2.1998 .

[8] Philippe Besse. Learning Statistics & Data mining. UMR CNRS C5583.Version July 2009

[9] Visible Korean Human. http://vkh3.kisti.re.kr/new/overview/index.htm

[10] Judenhofer MS, Catana C, Swann BK, et al. Simultaneous PET/MR images, acquired

with a compact MRI compatible PET detector in a 7 tesla magnet. Radiology. 244:807–814.2007.

[11] Judenhofer MS, Wehrl HF, Newport DF, et al. Simultaneous PET/MRI: a new perspective for functional and morphological imaging. Nat Med.14:459–465. 2008.

[12] O. de Dreuille et al, 'Principe et technique de la tomographie par émission de positons (TEP)', Elsevier SAS, EMC-Radiologie 1 (2004) 2–35, imaging. Nat Med.14:459–465.2008.

[13] Hu, Z. et al. MR-Based Attenuation Correction for a Whole-Body Sequential PET/MR System. •Nuclear Science Symposium and Medical Imaging Conference • 2009

Optimisation IV

Mapping Real Time Applications on NoC Architecture with Hybrid Multi-objective PSO Algorithm

A.Benyamina
Invited Searcher at INRIA/FUTUR France
Laboratoire d'Informatique d'ORAN-LIO
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie
Email: benyanabou@yahoo.fr

B.Beldjilali
Laboratoire d'Informatique d'ORAN-LIO
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie
Email: bouzianebeldjilali@yahoo.fr

S.Eltar and K.Dellal
Université d'ORAN ES-SENIA
BP 1524 EL mnaouer
ORAN Algérie

***Abstract*—Many tools will be required to develop a NoC architecture for a specific application. A tool which can map and schedule an application or a set of applications to a given NoC architecture will be essential and must be able to satisfy many relative trade-offs (real-time, performance, low power consumption, time to market, re-usability, cost, area, etc). This paper targets a very important sentence in the cycle development embedded systems, the design software. Our problem is related to the issue of solving the problem of placing application in a network on chip under the constraints of load balancing, bandwidth, available memory, size of the queue with the objective of minimizing execution time and therefore energy consumption. Our approach is based on a new approach PSO (Particle Swarm Optimization) to solve the problem of placement.**

Key words : **Keywords**: Mapping, Scheduling, PSO algorithm, Network-On-Chip, Multi-objective optimization, Dijkstra algorithm

I. Introduction

As silicon [6] technology keeps scaling, it is becoming technically feasible to integrate entire and complex systems on the same silicon die. As result of this Systems on Chip (SOC) are inherently heterogeneous and therefore complex, they are often formed multiple processors of different types (RISC, DSP; ASIC, for examples) features dedicated hardware or reconfiguration and peripheral.

To extend the integration and perform the design sweeps adopted the traditional concept of computer networking component interconnect-based routers, switches. well, a network on chip (NOC) or plus generally MPSoCs are a relatively aim new approach to integrated circuits on a platform SoC.[10]

Then MPSoCs and NoC are widely used in embedded systems (such as cellular phones, automotive control engines, etc..) where, once deployed in field, they always run the same set of applications.

In this paper we will focus on mesh-based NoC architectures, in which resources communicate with each other via mesh of switches that route and buffer messages. A ressource is generally any core : a general processor GP, a memory, an FPGA, DSP. A two dimensional mesh interconnection topology is simplest from a layout perspective and the local interconnection between resources and switches are independent of the size of the network.

Nevertheless, routing in a two dimensional mesh is easy, resulting in potentially small switches, high bandwidth, short clock cycles, and overall scalability [9]. One of the most onerous tasks in this context is the topological mapping of the resources on the mesh in such a way to optimize certain performances indexes (e.g power, performance). Mapping is, in fact, a problem of quadratic assignment that is known to be NP-hard.

The size space search of the problem increase exponentially with the system size depending on number of resources, tasks and communications. It is therefore of strategic importance to define methods to search a mapping that will optimize the desired performance indexes. In addition, the strategies have to handle a multi criteria exploration of the space of possible architectural mapping alternatives. The objectives to be optimized are, in fact, frequently multiple rather than single, and are almost always in contrast with each other. There is therefore non single solution to the problem of exploration (i.e single mapping) but a set of equivalent (i.e not dominated) possible architectural alternatives, featuring a different trade-off between the values of the objectives to be optimized (Pareto Set) [7].

Then a critical task for recent MPSoCs is the minimization of the energy consumed. We start from a well-characterized task graph, a directed acyclic graph representing a functional abstraction of the application that will run on the MPSoC. Each task is characterized by the number of clock cycles used for its execution. Clearly the duration of each task and the energy spent for running it depends on the clock frequency used during the task execution.

In addition, tasks connected by arcs in the task graph communicate if they are allocated to different processors.

Each edge is characterized by a number that corresponds to the total number of bytes exchanged between two tasks. Defining the optimal allocation, scheduling and voltage scaling for minimizing energy in MPSoCs is the aim of this paper. Energy is consumed during task execution and task communication.

The problem we face is very complex. Because we solve in the same time the allocation of tasks to the processors and find the optimal path allocation (or communication mapping) referred in literature as Network Assignment [8].

For this we have used two methods, one based on PSO algorithms and other on Dijkstra's algorithm. The solution must also verify some constraints such area, memory, load balancing, link speed, bandwidth and certainly hard real time constraints.

Our contribution differs from the above in that we use a new method in such research area and in the other hand we try to map tasks on processors and in the same time map optimally communications on the links of topology NoC.

The work presented in this paper is a contribution to solving a widespread problem in the field of system design, embedded the placement of a large application on an architecture (NOC). Our work in an application is represented by a set of tasks that communicate with each other by sending message via bus on a heterogeneous architecture.

Our role is to place the tiles (task) on different elements (core) of architecture with the objectives of minimizing time execution and the energy consumption under the constraints of load balancing, bandwidth, available memory and size of the queue waiting processors.

To solve this problem, we used in the context of our present work, a new meta-heuristic algorithm Particle Swarm. it has proved its effectiveness in many fields such as optimization of networks, image processing and even control of industrial systems but it was never applied in our domaine.

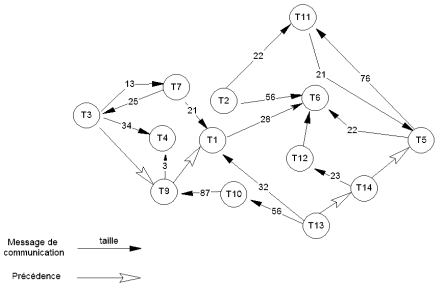


Fig. 1. Task graph

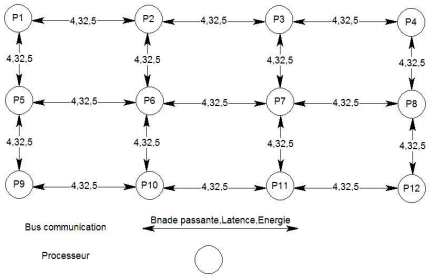


Fig. 2. NoC graph

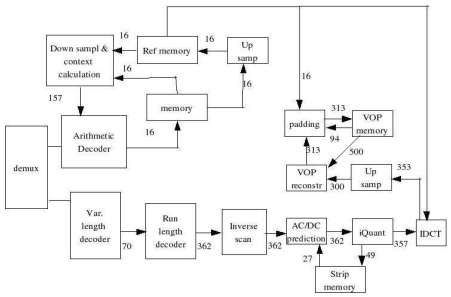


Fig. 3. block diagram of video object plane decoder with communication BW (in MB/s).

II. Problem definition and Formulation

The communication between the cores of SoC is represented by core graph.

Definition 1 : The Task Graph (CG)is a directed graph, $G(V, E)$ with each vertex $V_i \in V$ representing a task and the directed edge (v_i, v_j), denoted as $e_{ij} \in E$, representing the communication between the tasks v_i and v_j. The weight of the edge e_{ij}, denoted by Q_{ij} represents the bandwidth of the communication from v_i to v_j. The connectivity and link bandwidth of the NoC is represented by the NoC topology graph.

Definition 2 : The NOC Topology graph (NT) is a directed graph $P(U, F)$ with each vertex $u_i \in U$ representing a node in the topology and the directed edge (u_i,u_j) denoted as $f_{ij} \in F$representing a direct communication between the vertices's u_i and u_j. The weight of the edge f_{ij}, denoted by bw_{ij} represents the bandwidth available across the edge f_{ij} (see figure 1).

The mapping of the task graph $G(V, E)$ on to the processor graph $P(U, F)$ is defined by one to one mapping function map :

map : $V \rightarrow U$, s.t $map(v_i) = u_j \ \forall v_i \in V \ \exists u_j \in U$.

The mapping is defined when $\mid V \mid \leq \mid U \mid$ (see figure)

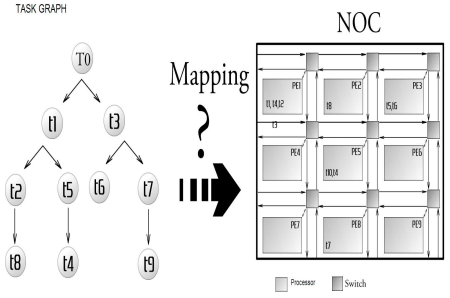


Fig. 4. Shows the mapping of task graph on NoC graph

A. Mathematical formulation

For an CG each node represent one task with its characteristics or property.

Let $T = t_1, t_2, ..., t_n$ be the set of all tasks represented by CG.

$P = p_1, p_2, ..., p_s$ is the set of processors represented by the nodes in NT.

We consider that each processor p can run in different modes m_1, m_2 or m_3.

We model the allocation problem with binary variables X_{ij}^m such that [6]:

$X_{ij}^m = 1$ if $task_i$ is mapped on the processor j and runs in mode m, o otherwise.

$d_{ij}^m =$ duration of execution task i on processor j running at mode m.

$d_{ij}^m = \frac{WCN_{ij}^m}{f_j^m}$ where WCN_{ij}^m is the number cycle needed by $task_i$ to be executed on processor j at mode m.

f_j^m is the frequency of the clock for processor j at mode m.

dl_i is deadline for $task_i$. The time at wish $task_i$ must be terminated.

There $dl_{final} = dl_n$ is deadline of last $task_n$ and of the application.

Q_{ij} is the size of data moved between $task_i$ and $task_j$.

dQ_{ijpq}^m is duration of communication between tasks i and j if they are assigned respectively to processor p and q at mode m (we see after how we compute dQ_{ijpq}^m).

q_{pq}^m is the duration of one unit (octet or bit) communication between p and q at mode m.

e_{pq}^m energy consumption for one unit from p to q at mode m.

If there is not a direct link between p and q let $\mu(p,q) = (pi, pj)$ be path form p to q.

Then duration using links is :

$dQ_{ijpq}^m = \sum Q_{ij} \times q_{plpk}^m$ where $(pl, pk) \in \mu(p,q)$ and $i \neq j$, $p \neq q$.
Let us note that if tasks i and j are mapped to the same processor the duration is negligible in comparison with case where they are mapped to different processors. therefore in the previous expression i differ of j and also p differ of q.
And consumption for communicating $task_i$ and $task_j$ if they are assigned to p and q at mode m towards the same path is :

$E_{ijpq}^m = \sum Q_{ij} \times e^m{}_{plpk}$ where $(pl, pk) \in \mu(p,q)$

Since we also assume take into account communication, we assume that two communicating tasks running on the same processor do not consume any energy and do not spend any time (indeed the communication time and energy spent are included in the execution time and energy), when if they are allocated on two different processors, they both consume energy and spend time. Each path contain some switch and router that need power consumption and duration.[5]
Ascia and al [7] address this problem but not explain how compute this. In This work we will consider an average value of consumption (Csw) and duration (dsw); we can estimate theses considering communications, input and output to router as stochastic.

Then if $\mid \mu(p,q) \mid$ is the length of path $\mu(p,q)$, the total consumption of switch and router on this path is :

$$\mid \mu(p,q) \mid +1) \times C_{sw}$$

And the total duration is :

$$\mid \mu(p,q) \mid +1) \times d_{sw}$$

We can now done equation of total consumption due to communication between $task_i$ and $task_j$ assigned respectively at processor p and q at mode m :

$$E_{ijpq}^m = E_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times C_{sw} \quad (1)$$

and duration due to communication as :

$$dQ_{ijpq}^m = dQ_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times d_{sw} \quad (2)$$

We explicit now total consumption and duration of processors.
For one $task_i$ mapped at $processor_p$ the duration is the sum of it's start time and duration of all its communications with other tasks mapped to other processors. The strategies scheduling is LS with ASAP (As Soon As Possible). Than for the $task_i$ mapped on $processor_p$ its duration D_{ip} is done by the following equation :

$$D_{ip} = d_{ip}^m + dQ_{ijpq}^m + (\mid \mu(p,q) \mid +1) \times d_{sw} \text{with} i \neq j, p \neq q \quad (3)$$

$Dstart_i$ is the time at wish $task_i$ begin execution. it is equal at the time of the end of the last task which precedes it. The duration of the application mapped on many processors it is equal at time end of the last task. Let D be total duration including time execution and communication over links of all

tasks.

$$D = Dstart_n + D_n \quad (4)$$

Were D_n it is the duration of $task_n$ that is the last task. $Dstart_n$ is the time at wish $task_n$ begin execution.

The total consumption including processor, link and switch consumption is done by the flowed equation :

$$E_{pr} = \sum_{i=1}^{N} \sum_{p=i+1}^{s} \sum_{m=ml}^{mn} x_{ip}^{m} \times ET_{ip}^{m} \quad (5)$$

Where ET_{ip}^{m} is the consumption of $task_i$ if it is assigned on processor p at mode m.
Then E_{pr} is total energy consumed by all processors in NoC.

$$E_{com} = \sum_{i=1}^{N} \sum_{j=i+1}^{N} \sum_{p=1}^{s} \sum_{q=p+1}^{s} \sum_{m=ml}^{mn} x_{ip}^{m} \times x_{jp}^{m} \times E_{ijpq}^{m} \quad (6)$$

E_{com} is total energy consumed by network (links, routers or switch)to assure all communications between all tasks overall the NoC.

B. Our multi-objective Model

Our method is based on evolutionary computing method and an optimizer path. We have to search set of solutions under $multi - objective$. We consider here two objective total duration and consumption.

First objective is duration D (4)
The second objective is the total power consumption done by E such that :

$$E = E_{pr} + E_{com} \quad (7)$$

Note date during computing D and E we look for the shortest path between two cores which satisfy the constraints (such bandwidth and buffer). To obtain this we used dijkstra. we refer to this problem with AAS (Assignation Affectation and Scheduling.)

III. AAS Problem resolution

The rationale behind our approach is the minimization of total power consummation in the order to augment the autonomy of system Nevertheless try to reach this objective increase time computing. Or embedded applications are generally real time and of course the deadline must not be exceed. minimizing power consumption increase time computing and minimizing time computing increase power consumption. We have here two contradictory objectives, what returns this very complex problem. Multi-objective problems have a set of Pareto-optimal solutions. each solution represents a different optimal trade-off between the objectives and is said "non-dominated" since it is not possible to improve one criterion without worsening another. We propose a multi-objective approach based on Particle Swarm optimization technique to solve our AAS problem.

A. PSO: Particle Swarm Optimization

A.Capone and M.Cesana proposed [3] an evolutionary population-based heuristic for optimization problems. It models the dynamic movement or behavior of the particles in a search space. By sharing information across the environment over generations, the search process is accelerated and is more likely to visit potential optimal or near-optimal solutions. PSO has been extended to cope with multi-objective problems which mainly consist of determining a local best and global best position of a particle in order to obtain a front of optimal solutions. One of the welle-known multi-objective techniques based on PSO algorithm is MOPSO [4]. It is able to generate almost the best set of non dominated solutions close to the true Pareto front. The main algorithm is given below.

$$E_{ijpq}^{m} = E_{ijpq}^{m} + (\mid \mu(p,q) \mid +1) \times C_{sw} \quad (8)$$

Algorithm 1 MOPSO-Main

input : Swarm at iteration t S^t, MaxArchiveSize, MaxIteration
Output : Repository REP
$Step0$:Initialization of Swarm
Initialize S at iteration $t = 0$
for each i $\in S^0$ **do**

  for each dimension d **do**
    Initialize $position_i$, save $pBest_i$, initialize velocity
    Specify $lowerbound_i$ and $upperbound_i$
  end for
end for
$Step1$: Evaluation of particles S
$Step2$: Update REP
for each i $\in S^t$ **do**
  compVector(i,REP)search-insert(S,REP)
end for
$STEP3$: Generate Mapping(associative grid): makeCost(Mincost)
$Step4$: Update Swarm :
for each i $\in S^t$ **do**

  for each dimension d **do**
    $Update - velocity_i$, $Update_position_i$
  end for
end for
$Step5$: Boundary chek
$Step6$: Update pBest
$Step7$:
if $t > MaxIteration$ **then**
  Stop
end if
$t = t + 1$ and GO TO $Step1$

The following are the phases involved in the resolution of the proposed algorithm. In continuous optimization problems, getting the initial position and velocity is more straightforward because random initialization can be used. However, since the mapping problem is a constrained optimization problem, the initial positions must represent feasible solutions. Thus, they need to be designed carefully.[2]
A position in the search space represents a set of assignments that is a solution to the problem. In our case, each position provides information about how processor in the NoC will execute each task. Than, for each position in the swarm, we assign a Boolean value to the variables X_{ij}^{m}. We consider a feasible solution, a solution that satisfies all hard and soft constraints. During the search, only non-feasible solutions that violate some soft constraints can be included in the population. This increases the likelihood of a non-feasible solution to mutate and provide a feasible one in later generation.

B. Algorithm description

Given the rapidly changing and increasingly complex systems on chip (SoC - System on Chip) to systems on chip multi-processors (MPSoC - multiprocessor SoCs), interconnection of communication modules or cores (IP – Intellectual Property) constituting these systems, has undergone a change both in topology of the structure. This responds to the constraints of performance and cost related to the complexity and the increasing of interconnected modules or IPs. Currently this process is moving towards the integration of a communication network on chip, implementing the transmission of data packets to nodes interconnected network corresponding to modules or IPs (processors, memory, peripheral controllers connected, etc..). This transmission is done through routers forming the network and implementing rules of referral and routing packets across the network.
In the design flow of an embedded system, the stage of investment and is directly related to the implementation of the application on an architecture specialist. The entries in this phase are:

- An application model
- A model of target architecture
- Constraints of performance and energy
- The objective functions to optimize

The output of this phase is an allocation of tasks and communications in natural resources, according to various tasks on these resources.

To run a distributed application, it is necessary to determine the best placement of spots that compose the target NoC architecture while reducing the execution time and energy constrained load balancing, memory, and the size of the queue. Our proposal for solving the problem of mapping is defined as follows:
The particle is a representation of the solution of the problem in this case describes the investment. If you have a NoC Mesh M processors running at M modes and an application

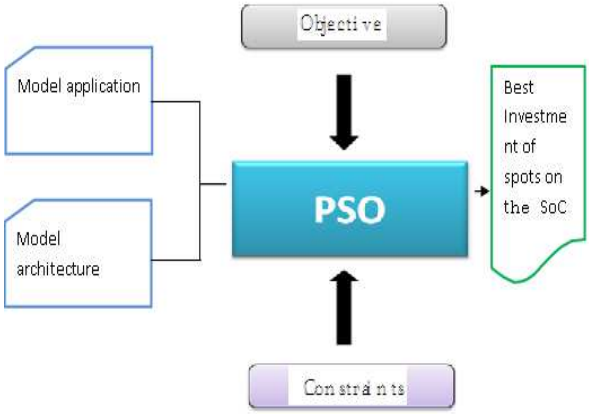


Fig. 5. PSO Algorithm

with N tasks when the particle is a matrix of N and S line column.
As an example take 2 processors and 3 tasks, then the particle is the mapping of 3 tasks: $Tasks2$ and 3 are assigned to P_1, $task1$ is assigned to P_2.

P_1	P_2
0	1
1	0
1	0

TABLE I
ONE PARTICLE

Get a set of points describing the Pareto front :

- Estimated front by iterative algorithms generate points near the front
- eliminating dominated points
- Problems of convergence:

1) approach the front
2) cover the entire front
3) Concept archive: Keep each iteration all the points not dominated.

Then The problem of placement of tasks in an application on a NoC to minimize cost objective functions. It can be formulated as follows:

- Given the graph of application (size of the task type of Soc, runtime memory required by SoC bandwidth required for a message and message size).
- Given the architectural graph (speed performance by mode, power consumption by mode, load balancing (load minimum and maximum), available memory in SoC size of the queue and bus latency and energy consumption due to transmission
- From the placement of tasks on the SoC's by different modes. This is equivalent to Minimize F (X): (f (time) f (energy)) The determination of the fitness function (or

adaptive function evaluation) involves several steps. Each time a swarm is generated according to the fitness of each particle must be evaluated.

- A particle represents a distribution of tasks of the application in the target NoC architecture.
- For a particle, move the communication costs of all messages for each message eliminating paths whose bandwidth is less than that required by the message (using the algorithm of the shortest path), then: We calculate the execution time of each task by mode which has been allocated within the SoC.

The Pareto Front is generated from existing solutions in the archive.

IV. Experimentations and results analysis

It's the first time who the PSO is applied in such area research. For now our first objective is to adapt this method to mapping real Time Applications on NoC by defining it's variables and parameters. We can study in the following works the performance of this algorithm in this area. The algorithm is coded in JAVA programming language and all the experiments were carried out on a Pentium 3.2 GHz. We have varied

]

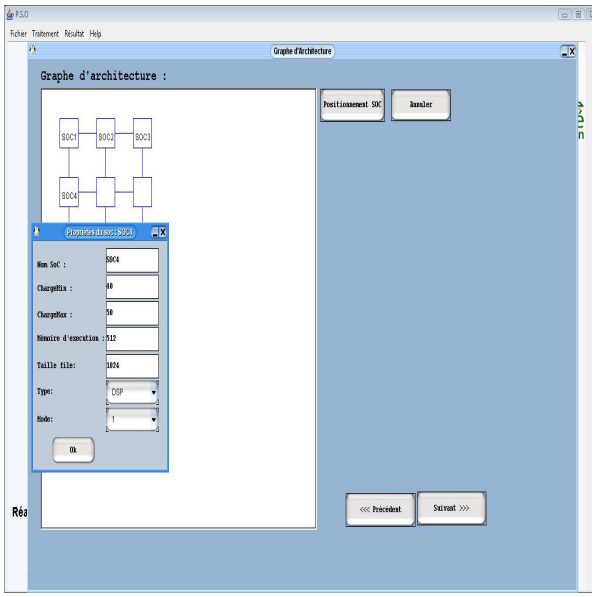


Fig. 6. Application interface

nombre of generations for a same exemple to know wath is the best interval of generations for similaire size of applications. In general study the searchers fixe this nombre at 20 and our experiments shows that not necessarily to take this nombre bigger than 20 nevertheless our exemple is not taken with a big size(see figure 7).

The other parameter of the algorithm is the size of swarm. It is also important because it influences the search convergence. Our experiment show that is important to take Swarm size near 100.(see figure 8)

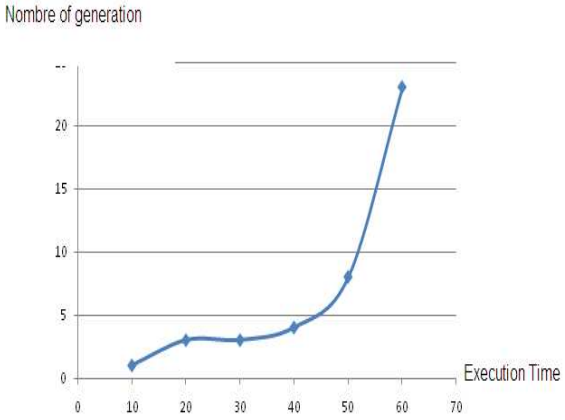


Fig. 7. influence of nombre generation on Time computing

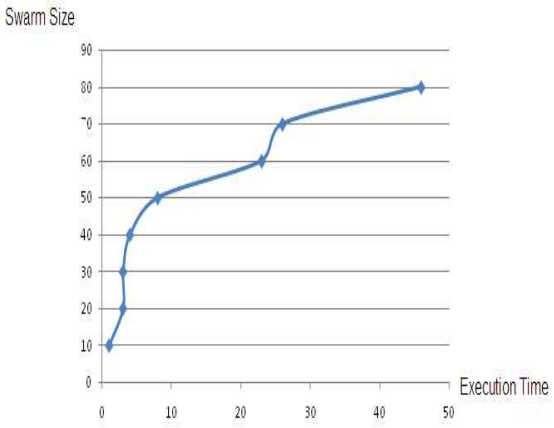


Fig. 8. influence of size swarm on Time computing

Figure (9) shows the results of different experiments for applications of average size that we have obtained. The achieved results and performances indicate that the proposed algorithm has a polynomial complexity and that it is adapted for computationally hard problems.

V. Conclusion

In this paper an algorithm for allocation and scheduling has been proposed exploiting the method of PSO and Disjkstra's shortest path algorithm. Under that we can't assert that this algorithm gives exact results for all kinds of problems. The proposed framework searches a good mapping for heterogeneous distributed embedded systems and of course this approach can easily be applied to regular architectures and first experimental results show that the global time for research is reasonable. Nevertheless for asserting that this method can be used for solving problems with large sizes we must experiment the algorithm using examples with many hundreds of PE and tasks. Then we consider for now this work a first step in our research for a good meta-heuristic which can be crossed with other exact method to solve th global problem known as GILR.

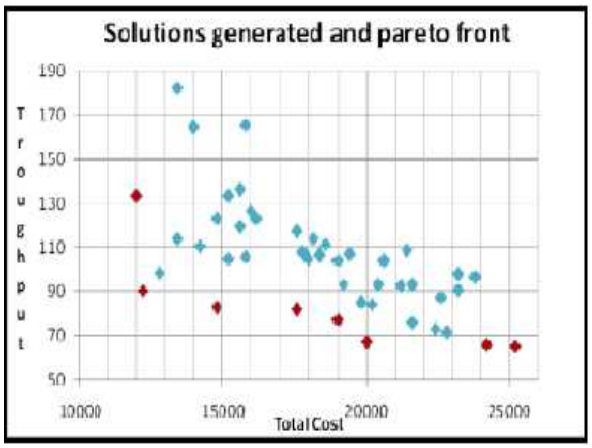


Fig. 9. Feasible solutions generated with the same initial parameters

This will be the continuation for this work and our perspective.

REFERENCES

[1] IR.Quadri, P.Boulet, S.Meftali and JL.Dekeyser, *Using an MDE Approach for Modeling of Interconnection Networks*, Proceedings of the The International Symposium on Parallel Architectures, Algorithms, and Networks. ISPAN, pp 289-294, 2008.

[2] HJ.Escalante, M.Montes and LE.Sucar *Particle Swarm Model Selection*, The Journal of Machine Learning Research. pp.405-440, vol 10, 2009.

[3] A.Capone, M.Cesana, I.Filippini and F.Malucelli, *Optimization models and methods for planning wireless mesh networks* , Computer Network. accepted for publication January 2008.

[4] TDL.Truong, *Shared protection for multi-domain networks*, Ph.D.thesis. University of Montreal. Sept.2007

[5] AH.Benyamina, P.Boulet and B.Beldjilali , *An Hybrid algorithm for Mapping on NoC Architectures*, META'08 . Hammamat,Tunisia,October, 2008.

[6] A.Benymina and P.Boulet,*Multi-objective Mapping for NoC Architectures* , Journal of Digital Information Management (JDIM). ,pages 378-384, volume 5, number 6, 2007.

[7] G.Ascia, V.Catania and M.Palesi, *A Multi-objective Genetic Approach to Mapping Problem on Network-on-Chip* Journal of Universal Computer Science. , pages 370-394, vol 2, (2006)

[8] D.Shin, J.Kim,*Power-Aware Communication Optimisation for Networks-on-Chips with Voltage Scalable Links* , in Proc. CODES+ISSS'04, , pp.170-175, Sep.2004.

[9] L.Benini, D.Bertozzi, A.Guerri and M.Milano, *Allocation, Scheduling and Voltage Scaling on Energy Aware MPSoCs*, CPAIOR pp.44-58, 2006.

[10] T.Bjerregaard and S.Mahadevan, *survey of research and pratices of network-on-chip*, ACM Comput Surv. 38(1):1, 2006.

Multiobjective programming under generalized V-type I invexity

Hachem Slimani[1] and Mohammed Said Radjef[2]

Laboratory of Modeling and Optimization of Systems (LAMOS)
Operational Research Department, University of Bejaia, 06000 Bejaia, Algeria,
haslimani@gmail.com [1], radjefms@gmail.com [2]

Abstract. In this paper, we are concerned with a differentiable multiobjective programming problem with inequality constraints. We introduce new concepts of generalized V-type I invexity problems in which each component of the objective and constraint functions is considered with respect to its own function η_i or θ_j. In the setting of these definitions, we establish new Karush-Kuhn-Tucker type necessary and sufficient optimality conditions for a feasible point to be efficient or properly efficient. Furthermore, we show, with examples, that the obtained results allow to prove that a feasible point is an efficient or properly efficient solution even if it is not an usual vector Karush-Kuhn-Tucker point for a multiobjective programming problem.
Keywords: Multiobjective programming; Generalized V-type I problem; Generalized Karush-Kuhn-Tucker condition; Optimality; (Properly) efficient point

1 Introduction

In optimization theory, convexity plays a very important role especially in the construction of sufficient conditions of optimality and duality theory see, for example, Mangasarian [22] and Bazaraa et al. [6]. Several generalizations were introduced in the literature in order to weaken the hypothesis of convexity in mathematical programming and multiobjective problems. Hanson [14] introduced the concept of invexity for the differentiable functions, generalizing the difference $(x - x_0)$ in the definition of convex function to any function $\eta(x, x_0)$. He proved that if, in a mathematical programming problem, instead of the convexity assumption, the objective and constraint functions are invex with respect to a same vector function η, then both the sufficiency of Karush-Kuhn-Tucker conditions and weak and strong Wolfe duality still hold. Later, Hanson and Mond [15] introduced two new classes of functions called type I and type II functions, which are not only sufficient but are also necessary for optimality in primal and dual problems, respectively. In [28], Rueda and Hanson extended type I functions to pseudo-type I and quasi-type I functions and have obtained sufficient optimality criteria for a nonlinear programming problem involving these functions. Rueda et al. [29] obtained optimality and duality results for several mathematical programs by combining the concepts of type I and univex functions defined

by Bector et al. [7]. For other generalizations of invexity, see [2, 8, 11, 19, 23, 27, 30] and the references cited therein.

On the other hand, Kaul et al. [18] considered a multiobjective problem involving generalized type-I functions, with scalarization, and obtained some results on optimality and duality, where the Wolfe and Mond-Weir duals are considered. Mishra [24] considered a multiple objective nonlinear programming problem and obtained optimality, duality and saddle point results of a vector valued Lagrangian by combining the concepts of generalized type-I and univex functions. Aghezzaf and Hachimi [1] introduced new classes of generalized type-I vector-valued functions and, without scalarization, derived various duality results for a nonlinear multiobjective programming problem. Following Jeyakumar and Mond [17] and Kaul et al. [18], Hanson et al. [16] introduced the V-type I problem with respect to η, including positive real-valued functions α_i and β_j in their definition, and they obtained optimality conditions and duality results under various types of generalized V-type I requirements. For other optimality conditions and approaches to duality for multiobjective optimization problems, the reader can refer to the references [3–5, 9, 10, 12, 13, 20, 25, 26, 33].

However, in the literature, the type I functions and the V-type I problems (the invex problems in general) are considered with respect to a same function η. Jeyakumar and Mond [17] have observed that one major difficulty in all of these extensions of convexity is that invex problems require a same function η for the objective and constraint functions. This requirement turns out to be a major restriction in applications. In [31], a nonlinear programming is considered and KT-invex, weakly KT-pseudo-invex and type I problems with respect to different η_i are defined. A new Karush-Kuhn-Tucker type necessary condition is introduced and duality results are obtained, for Wolfe and Mond-Weir type dual programs, under generalized invexity assumptions. In [32], the invexity with respect to different η_i is used in the nondifferentiable case.

Motivated and inspired by work in [31, 32], in this paper, we define new classes of generalized V-type I invexity problems in which each component of the objective and constraint functions is considered with respect to its own function η_i or θ_j. These multiobjective programming problems preserve the sufficient optimality conditions under a generalized Karush-Kuhn-Tucker condition, and avoid the major difficulty of verifying that the inequality holds for a same function η for invex functions. This relaxation widens the area of application and allows to get results which are applicable to prove that a feasible point is an efficient or properly efficient solution even if it is not an usual vector Karush-Kuhn-Tucker point for a multiobjective programming problem. Further, we illustrate the obtained results by some examples where we have a large choice to take the different functions η_i and θ_j with respect to which the objective and constraint functions are considered.

2 Preliminaries and definitions

The following conventions for inequalities will be used. If $x = (x_1, ..., x_n)$, $y = (y_1, ..., y_n) \in \mathbb{R}^n$, then: $x \leqq y \Leftrightarrow x_i \leqq y_i,\ \forall\ i = 1, ..., n$; $x \leq y \Leftrightarrow x \leqq y$ and $x \neq y$; $x < y \Leftrightarrow x_i < y_i,\ \forall\ i = 1, ..., n$. We also note $\mathbb{R}^q_{\leqq}$ (resp. $\mathbb{R}^q_{\geq}$ or $\mathbb{R}^q_{>}$) the set of vectors $y \in \mathbb{R}^q$ with $y \geqq 0$ (resp. $y \geq 0$ or $y > 0$).

We consider the following multiobjective optimization problem

$$(VP) \qquad \begin{array}{l} \text{Minimize } f(x) = (f_1(x), ..., f_N(x)), \\ \text{subject to } \ g(x) \leqq 0, \end{array}$$

where $f : D \to \mathbb{R}^N$ and $g : D \to \mathbb{R}^k$ are differentiable functions on the open set $D \subseteq \mathbb{R}^n$. Let $X = \{x \in D : g(x) \leqq 0\}$ the set of feasible solutions of (VP). For $x_0 \in D$, we denote by $J(x_0)$ the set $\{j \in \{1, ..., k\} : g_j(x_0) = 0\}$, $J = |J(x_0)|$ and by $\tilde{J}(x_0)$ (resp. $\bar{J}(x_0)$) the set $\{j \in \{1, ..., k\} : g_j(x_0) < 0$ (resp. $g_j(x_0) > 0)\}$. We have $J(x_0) \cup \tilde{J}(x_0) \cup \bar{J}(x_0) = \{1, ..., k\}$ and if $x_0 \in X$, $\bar{J}(x_0) = \varnothing$.

We recall some optimality concepts, the most often studied in the literature, for the problem (VP). For other notions and their connections, see [34].

Definition 1. *A point $x_0 \in X$ is said to be a weakly efficient (an efficient) solution of the problem (VP), if there exists no $x \in X$ such that*

$$f(x) < f(x_0)\ (f(x) \leq f(x_0)). \tag{1}$$

Definition 2. *An efficient solution $x_0 \in X$ of (VP) is said to be properly efficient, if there exists a positive real number M such that the inequality*

$$f_i(x_0) - f_i(x) \leqq M[f_j(x) - f_j(x_0)], \tag{2}$$

is verified for all $i \in \{1, ..., N\}$ and $x \in X$ such that $f_i(x) < f_i(x_0)$, and for a certain $j \in \{1, ..., N\}$ such that $f_j(x) > f_j(x_0)$.

Kaul et al. [18] and Hanson et al. [16] defined the invex type I functions and the invex V-type I problem respectively, by taking a same η for the objective and constraint functions. In what follows, we define vector type I problems, where each component of the objective and constraint functions is considered with respect to its own function η_i or θ_j.

Definition 3. *We say that the problem (VP) is of V-type I at $x_0 \in D$ with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$, if there exist $(N + k)$ vector functions $\eta_i : X \times D \to \mathbb{R}^n,\ i = \overline{1, N}$ and $\theta_j : X \times D \to \mathbb{R}^n,\ j = \overline{1, k}$ such that for all $x \in X$:*

$$f_i(x) - f_i(x_0) \geqq [\nabla f_i(x_0)]^t \eta_i(x, x_0),\ \forall\ i = 1, ..., N, \tag{3}$$

$$-g_j(x_0) \geqq [\nabla g_j(x_0)]^t \theta_j(x, x_0),\ \forall\ j = 1, ..., k. \tag{4}$$

If the inequalities in (3) are strict (whenever $x \neq x_0$), we say that (VP) is of semi strictly V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$.

Example 1. We consider the following multiobjective optimization problem

$$\begin{array}{l}\text{Minimize } f(x) = (x + \sin x, \cos x),\\ \text{subject to } \ g(x) = x - \frac{\pi}{6} \leqq 0,\end{array}$$

where $f :]0, \frac{\pi}{2}[\to \mathbb{R}^2$ and $g :]0, \frac{\pi}{2}[\to \mathbb{R}$. The set of feasible solutions of problem is $X = \{x \in]0, \frac{\pi}{2}[\ : \ g(x) \leqq 0\} =]0, \frac{\pi}{6}]$. The problem is V-type I at $x_0 = \frac{\pi}{6} \in X$ with respect to $(\eta_i)_{i=1,2}$ and θ defined as follows: $\eta_1(x, x_0) = (\sin x - \sin x_0)/\cos x_0$, $\eta_2(x, x_0) = (\cos x_0 - \cos x)/\sin x_0$ and $\theta(x, x_0) = \sin(x - \frac{\pi}{6})$ ($\theta(x, x_0)$ may be any negative scalar function on X).

Definition 4. *We say that the problem (VP) is of quasi V-type I at $x_0 \in D$ with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$, if there exist $(N + k)$ vector functions $\eta_i : X \times D \to \mathbb{R}^n$, $i = \overline{1, N}$ and $\theta_j : X \times D \to \mathbb{R}^n$, $j = \overline{1, k}$ such that for some vectors $\mu \in \mathbb{R}^N_{\leqq}$ and $\lambda \in \mathbb{R}^k_{\leqq}$:*

$$\sum_{i=1}^{N} \mu_i[f_i(x) - f_i(x_0)] \leqq 0 \ \Rightarrow \ \sum_{i=1}^{N} \mu_i[\nabla f_i(x_0)]^t \eta_i(x, x_0) \leqq 0, \ \forall \ x \in X, \quad (5)$$

$$\sum_{j=1}^{k} \lambda_j g_j(x_0) \geqq 0 \ \Rightarrow \ \sum_{j=1}^{k} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) \leqq 0, \ \forall \ x \in X. \quad (6)$$

If the second (implied) inequality in (5) is strict ($x \neq x_0$), we say that (VP) is of semi strictly-quasi V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$.

Example 2. We consider the following multiobjective optimization problem

$$\begin{array}{l}\text{Minimize } f(x) = (\sin x, \cos x),\\ \text{subject to } \ g(x) = x - \frac{\pi}{3} \leqq 0,\end{array}$$

where $f :]0, \frac{\pi}{2}[\to \mathbb{R}^2$ and $g :]0, \frac{\pi}{2}[\to \mathbb{R}$. The set of feasible solutions of problem is $X =]0, \frac{\pi}{3}]$. The problem is semi strictly-quasi V-type I at $x_0 = \frac{\pi}{3} \in X$ with respect to $(\eta_i)_{i=1,2}$ and θ defined as follows: $\eta_1(x, x_0) = x_0 - x$, $\eta_2(x, x_0) = \sin(x - x_0)$ and $\theta(x, x_0) = -\cos(x + \frac{\pi}{6})$ (as it can be seen by taking $\mu_1 = \frac{1}{4}$, $\mu_2 = \frac{3}{4}$ and $\lambda = \frac{1}{2}$).

Definition 5. *We say that the problem (VP) is of pseudo V-type I at $x_0 \in D$ with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$, if there exist $(N + k)$ vector functions $\eta_i : X \times D \to \mathbb{R}^n$, $i = \overline{1, N}$ and $\theta_j : X \times D \to \mathbb{R}^n$, $j = \overline{1, k}$ such that for some vectors $\mu \in \mathbb{R}^N_{\leqq}$ and $\lambda \in \mathbb{R}^k_{\leqq}$:*

$$\sum_{i=1}^{N} \mu_i[\nabla f_i(x_0)]^t \eta_i(x, x_0) \geqq 0 \ \Rightarrow \ \sum_{i=1}^{N} \mu_i[f_i(x) - f_i(x_0)] \geqq 0, \ \forall \ x \in X, \quad (7)$$

$$\sum_{j=1}^{k} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) \geqq 0 \ \Rightarrow \ \sum_{j=1}^{k} \lambda_j g_j(x_0) \leqq 0, \ \forall \ x \in X. \quad (8)$$

If the second (implied) inequality in (7) (resp. (8)) is strict ($x \neq x_0$), we say that (VP) is of semi strictly-pseudo V-type I in f (resp. in g) at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$. If the second (implied) inequalities in (7) and (8) are both strict ($x \neq x_0$), we say that (VP) is of strictly-pseudo V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$.

Example 3. We consider the following multiobjective optimization problem

$$\begin{array}{c} \text{Minimize } f(x) = (-x, -\cos^2 x), \\ \text{subject to } \ g(x) = x - \frac{\pi}{3} \leqq 0, \end{array}$$

where $f :]0, \frac{\pi}{2}[\to \mathbb{R}^2$ and $g :]0, \frac{\pi}{2}[\to \mathbb{R}$. The set of feasible solutions of problem is $X =]0, \frac{\pi}{3}]$. The problem is strictly-pseudo V-type I at $x_0 = \frac{\pi}{3} \in X$ with respect to $(\eta_i)_{i=1,2}$ and θ defined as follows: $\eta_1(x, x_0) = x - x_0$, $\eta_2(x, x_0) = \sin x_0(\cos x_0 - \cos x)$ and $\theta(x, x_0) = \sin(x - x_0)$ (as it can be seen by taking $\mu_1 = \frac{3}{4}$ and $\mu_2 = \lambda = \frac{1}{4}$), but the problem is not V-type I at x_0 with respect to the same $(\eta_i)_{i=1,2}$ and θ because f_2 is not invex at $x_0 = \frac{\pi}{3}$ with respect to η_2 (take $x = \frac{\pi}{6}$).

Definition 6. *We say that the problem (VP) is of quasi pseudo V-type I at $x_0 \in D$ with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$, if there exist $(N + k)$ vector functions $\eta_i : X \times D \to \mathbb{R}^n, i = \overline{1, N}$ and $\theta_j : X \times D \to \mathbb{R}^n$, $j = \overline{1, k}$ such that for some vectors $\mu \in \mathbb{R}^N_{\geqq}$ and $\lambda \in \mathbb{R}^k_{\geqq}$ the relations (5) and (8) are satisfied.*
If the second (implied) inequality in (8) is strict ($x \neq x_0$), we say that (VP) is of quasi strictly-pseudo V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$.

Example 4. We consider the following multiobjective optimization problem

$$\begin{array}{c} \text{Minimize } f(x) = (\frac{1}{x}, x), \\ \text{subject to } \ g(x) = x - 1 \leqq 0, \end{array}$$

where $f :]0, +\infty[\to \mathbb{R}^2$ and $g :]0, +\infty[\to \mathbb{R}$. The set of feasible solutions of problem is $X =]0, 1]$. The problem is quasi strictly-pseudo V-type I at $x_0 = 1 \in X$ with respect to $(\eta_i)_{i=1,2}$ and θ defined as follows: $\eta_1(x, x_0) = x^2 - x_0^2$, $\eta_2(x, x_0) = x_0 - x$, and $\theta(x, x_0) = x - x_0$ (as it can be seen by taking $\mu_1 = \frac{3}{4}$ and $\mu_2 = \lambda = \frac{1}{4}$).

Definition 7. *We say that the problem (VP) is of pseudo quasi V-type I at $x_0 \in D$ with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$, if there exist $(N + k)$ vector functions $\eta_i : X \times D \to \mathbb{R}^n, i = \overline{1, N}$ and $\theta_j : X \times D \to \mathbb{R}^n$, $j = \overline{1, k}$ such that for some vectors $\mu \in \mathbb{R}^N_{\geqq}$ and $\lambda \in \mathbb{R}^k_{\geqq}$ the relations (7) and (6) are satisfied.*
If the second (implied) inequality in (7) is strict ($x \neq x_0$), we say that (VP) is of strictly-pseudo quasi V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j=\overline{1,k}}$.

Example 5. We consider the following multiobjective optimization problem

$$\begin{array}{c} \text{Minimize } f(x) = (-x^2, -x^4), \\ \text{subject to } \ g(x) = (x - 1)^3 \leqq 0, \end{array}$$

where $f :]0, +\infty[\to \mathbb{R}^2$ and $g :]0, +\infty[\to \mathbb{R}$. The set of feasible solutions of problem is $X =]0, 1]$. The problem is pseudo quasi V-type I at $x_0 = 1 \in X$ with respect to $(\eta_i)_{i=1,2}$ and θ defined as follows: $\eta_1(x, x_0) = x^2 - x_0^2$, $\eta_2(x, x_0) = x^4 - x_0^4$ and $\theta(x, x_0) = x^2$ ($\theta(x, x_0)$ may be any scalar function). The problem is not V-type I at x_0 with respect to the same $(\eta_i)_{i=1,2}$ and θ but it is with respect to other functions $\eta_1'(x, x_0) = \frac{1}{2}\eta_1(x, x_0)$, $\eta_2'(x, x_0) = \frac{1}{4}\eta_2(x, x_0)$ and $\theta'(x, x_0) = \theta(x, x_0)$.

Figure 1. summarizes the interconnection between the different concepts of problems defined above.

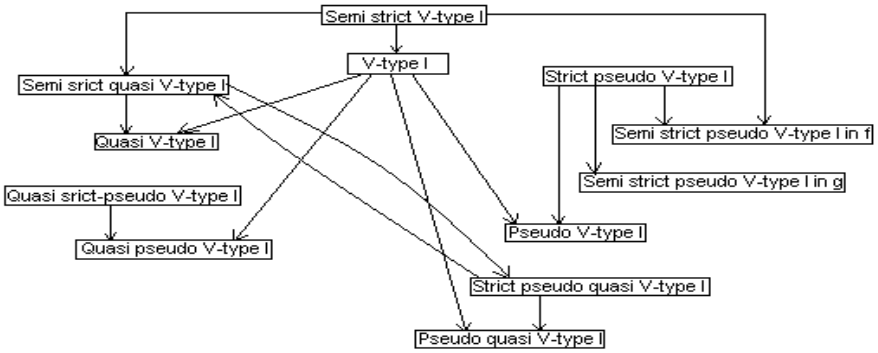


Fig.1. Connections between the different concepts of problems

In the figure 1., "concept $c_1 \to$ concept c_2" means that: if the problem (VP) is of "concept c_1" at x_0 with respect to $(\eta_i)_i$ and $(\theta_j)_j$, then (VP) is of "concept c_2" at x_0 with respect to the same functions $(\eta_i)_i$ and $(\theta_j)_j$.
However, the problem (VP) can be, furthermore, of "concept c_2" at x_0 with respect to other functions $(\bar{\eta}_i)_i$ and $(\bar{\theta}_j)_j$ without it be of "concept c_1" at x_0 with respect to the same functions $(\bar{\eta}_i)_i$ and $(\bar{\theta}_j)_j$.
For example: "V-type I $\to$ pseudo quasi V-type I" means that if the problem (VP) is V-type I at x_0 with respect to $(\eta_i)_i$ and $(\theta_j)_j$, then it is pseudo quasi V-type I at x_0 with respect to the same functions $(\eta_i)_i$ and $(\theta_j)_j$ but the converse is not true in general, see the example 5.

3 Optimality conditions

Weir [33], Kaul et al. [18] and Hanson et al. [16] have given Karush-Kuhn-Tucker type necessary conditions for x_0 to be properly efficient for (VP). Maeda [20, 21] has given the same necessary conditions for x_0 to be efficient for (VP). Osuna-Gómez et al. [26] (resp. Arana-Jiménez et al. [5]) have given Fritz-John and Karush-Kuhn-Tucker type necessary conditions for x_0 to be weakly efficient (resp. efficient) for (VP). Now, in the setting of the new concepts of generalized invexity with respect to different η_i, we give a new Karush-Kuhn-Tucker type necessary optimality condition for x_0 to be efficient for (VP) and then we establish sufficient conditions for a feasible solution to be efficient or properly efficient for (VP).

In the following theorem, we extend the Karush-Kuhn-Tucker type necessary condition established for nonlinear programming programs in [31], to the case of multiobjective programs.

Theorem 1. *(Karush-Kuhn-Tucker type necessary optimality condition) Suppose that x_0 is an efficient solution for (VP) and the functions $f_i,\ i = \overline{1,N}$, $g_j,\ j \in J(x_0)$ are differentiable at x_0. Then there exist vector functions $\eta_i : X \times D \to \mathbb{R}^n,\ i = \overline{1,N},\ \theta_j : X \times D \to \mathbb{R}^n,\ j \in J(x_0),\ (\eta_i \not\equiv 0,\ \forall\ i = \overline{1,N},\ \theta_j \not\equiv 0,\ \forall\ j \in J(x_0))$ and vectors $\mu \in \mathbb{R}^N_>$ and $\lambda \in \mathbb{R}^J_>$ such that $(x_0, \mu, \lambda, (\eta_i)_{i=\overline{1,N}}, (\theta_j)_{j\in J(x_0)})$ satisfies the following generalized Karush-Kuhn-Tucker condition*

$$\sum_{i=1}^{N} \mu_i [\nabla f_i(x_0)]^t \eta_i(x, x_0) + \sum_{j\in J(x_0)} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) \geqq 0,\ \forall\ x \in X. \qquad (9)$$

Proof. It suffices to take $\eta_i,\ i = 1, ..., N;\ \theta_j,\ j \in J(x_0);\ \mu$ and λ as follows:

- If f is a constant on X, then $\eta_i(x, x_0)$ can be any nonzero function.
  If f is not a constant on X, then there exists $\bar{x} \in X,\ f(\bar{x}) \neq f(x_0)$, it follows that there exists $i_0 \in \{1, ..., N\},\ f_{i_0}(\bar{x}) > f_{i_0}(x_0)$ because x_0 is efficient for (VP1). For all $x \in X$, consider the set $I_x = \{i \in \{1, ..., N\} : f_i(x) - f_i(x_0) > 0\}$. Note that I_x can be empty. Thus, $\eta_i(x, x_0) = \phi_i(x, x_0)[\nabla f_i(x_0)]$ with $\phi_i(x, x_0) = \begin{cases} f_{i_x}(x) - f_{i_x}(x_0), \text{ if } I_x \neq \emptyset \text{ (with } i_x = \min I_x); \\ f_{i_0}(\bar{x}) - f_{i_0}(x_0), \text{ otherwise.} \end{cases}$
- $\theta_j(x, x_0) = -g_j(x)[\nabla g_j(x_0)]$;
- $\mu_i = \frac{1}{N}$, for all $i = 1, ..., N$; $\lambda_j = \frac{1}{J}$, for all $j \in J(x_0)$.

Now, we present some Karush-Kuhn-Tucker type sufficient optimality conditions for (VP) under various types of generalized V-type I assumptions. We give several examples to illustrate the obtained results.

Theorem 2. *(Karush-Kuhn-Tucker type sufficient optimality conditions) Let x_0 be a feasible solution for (VP) and suppose that there exist $(N + J)$ vector functions $\eta_i : X \times X \to \mathbb{R}^n,\ i = \overline{1,N},\ \theta_j : X \times X \to \mathbb{R}^n,\ j \in J(x_0)$ and scalars $\mu_i \geqq 0,\ i = \overline{1,N},\ \sum_{i=1}^{N} \mu_i = 1,\ \lambda_j \geqq 0,\ j \in J(x_0)$ such that the generalized Karush-Kuhn-Tucker condition (9) is satisfied. Moreover, assume that one of the following conditions is verified:*

(a) the problem (VP) is quasi strictly-pseudo V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$, $(\theta_j)_{j\in J(x_0)}$ and for μ and λ;
(b) the problem (VP) is semi strictly-quasi V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$, $(\theta_j)_{j\in J(x_0)}$ and for μ and λ;
(c) the problem (VP) is strictly-pseudo V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$, $(\theta_j)_{j\in J(x_0)}$ and for μ and λ.

Then x_0 is an efficient solution for (VP).

Proof. Suppose that x_0 is not an efficient solution of (VP). Then there exists a feasible solution $x \in X$ such that $f(x) \leq f(x_0)$, which implies that

$$\sum_{i=1}^{N} \mu_i [f_i(x) - f_i(x_0)] \leqq 0. \tag{10}$$

From the above inequality and the condition (a), we obtain

$$\sum_{i=1}^{N} \mu_i [\nabla f_i(x_0)]^t \eta_i(x, x_0) \leqq 0. \tag{11}$$

By using the condition (9), we deduce that

$$\sum_{j \in J(x_0)} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) \geqq 0, \tag{12}$$

which implies, from the condition (a) (in view of definition 6), that

$$\sum_{j \in J(x_0)} \lambda_j g_j(x_0) < 0.$$

The last inequality contradicts the fact that $g_j(x_0) = 0,\ \forall\ j \in J(x_0)$ and hence the conclusion follows.
The proof of the part (b) is very similar to the proof of part (a), except that for this case the inequality (11) becomes strict ($<$), it follows that the inequality (12) becomes strict ($>$) and, using the reverse implication in (6), we get the contradiction again.
By condition (c), from $g_j(x_0) = 0,\ \lambda_j \geqq 0,\ \forall\ j \in J(x_0)$, in view of the reverse implication in (8), we obtain $\sum_{j \in J(x_0)} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) < 0,\ \forall\ x \in X \setminus \{x_0\}$.
By using (9), we deduce $\sum_{i=1}^{N} \mu_i [\nabla f_i(x_0)]^t \eta_i(x, x_0) > 0,\ \forall\ x \in X \setminus \{x_0\}$,
which implies, according to the relation (7) (for strictly-pseudo V-type I problem), that

$$\sum_{i=1}^{N} \mu_i [f_i(x) - f_i(x_0)] > 0,\ \forall\ x \in X \setminus \{x_0\}. \tag{13}$$

Thus (10) and (13) contradict each other, hence x_0 is an efficient solution of (VP). This completes the proof.

In order to illustrate the obtained result, we shall give an example of multiobjective optimization problem in which an efficient solution will be obtained by the application of theorem 2, whereas it will be impossible to apply for this purpose the sufficient optimality conditions using the usual Karush-Kuhn-Tucker condition.

Example 6. We consider the following multiobjective optimization problem

$$\begin{array}{ll} \text{Minimize} & f(x) = (x_1^3 - x_3, x_2^2 - x_1 - x_3), \\ \text{subject to} & g_1(x) = x_2 \leqq 0 \\ & g_2(x) = x_3^3 - x_2 \leqq 0, \\ & g_3(x) = x_1 \leqq 0, \end{array} \tag{14}$$

where $f : \mathbb{R}^3 \to \mathbb{R}^2$ and $g = (g_1, g_2, g_3) : \mathbb{R}^3 \to \mathbb{R}^3$. The set of feasible solutions of problem is $X = \{x = (x_1, x_2, x_3) \in \mathbb{R}^3 : x_2 \leqq 0,\ x_3^3 - x_2 \leqq 0 \text{ and } x_1 \leqq 0\}$.

- We have $x_0 = (0, 0, 0) \in X$ is not a vector Karush-Kuhn-Tucker point of problem (14), because the condition of Karush-Kuhn-Tucker at x_0 takes a form $\mu_1 \nabla f_1(x_0) + \mu_2 \nabla f_2(x_0) + \lambda_1 \nabla g_1(x_0) + \lambda_2 \nabla g_2(x_0) + \lambda_3 \nabla g_3(x_0) = (-\mu_2 + \lambda_3, \lambda_1 - \lambda_2, -\mu_1 - \mu_2) \neq (0, 0, 0),\ \forall\ (\mu_1, \mu_2) \geq 0,\ \forall\ (\lambda_1, \lambda_2, \lambda_3) \geqq 0$, then the known sufficient optimality conditions using this concept, for example from [4, 5, 13, 16–18, 24–26, 33] are not applicable.
- However, using the theorem 2, we have: there exist vector functions $\eta_1(x, x_0) = (x_1, x_2, x_3),\ \eta_2(x, x_0) = (x_1 + x_2, x_3, x_3),\ \theta_1(x, x_0) = (x_1, x_1, x_1),\ \theta_2(x, x_0) = (x_2, -x_2, x_2),\ \theta_3(x, x_0) = (x_3, x_3, x_3)$ and scalars $\mu_1 = 0,\ \mu_2 = \frac{1}{2},\ \lambda_1 = \lambda_2 = \lambda_3 = \frac{1}{3}$ such that the generalized Karush-Kuhn-Tucker condition (9) is satisfied and the problem (14) is strictly-pseudo V-type I at x_0 with respect to $(\eta_i)_{i=1,2}$, $(\theta_j)_{j=1,2,3}$, $\mu = (\mu_1, \mu_2)$ and $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ (the problem (14) is, in fact, quasi strictly-pseudo V-type I and semi strictly-quasi V-type I at x_0 with respect to the same $(\eta_i)_{i=1,2}$, $(\theta_j)_{j=1,2,3}$, $\mu = (\mu_1, \mu_2)$ and $\lambda = (\lambda_1, \lambda_2, \lambda_3)$). It follows that, by theorem 2, x_0 is an efficient solution for the given multiobjective optimization problem.

In the above example, we show that the hypothesis of x_0 to be a vector Karush-Kuhn-Tucker point is sometimes a strong sufficient condition and it is not indispensable to prove that x_0 is an efficient solution of (VP). In this way, the obtained optimality conditions may be considered as an extension of previously known results.

Example 7. The hypothesis of theorem 2 (with the condition (a)) are satisfied for the problem given in the example 4 at $x_0 = 1$ with respect to the same functions $(\eta_i)_{i=1,2}$, θ and for $\mu_1 = \frac{3}{4}$ and $\mu_2 = \lambda = \frac{1}{4}$. Then x_0 is an efficient solution for this problem.

Example 8. The hypothesis of theorem 2 (with the condition (b)) are satisfied for the problem given in the example 2 at $x_0 = \frac{\pi}{3}$ with respect to the same functions $(\eta_i)_{i=1,2}$, θ and for $\mu_1 = \frac{1}{4},\ \mu_2 = \frac{3}{4}$ and $\lambda = \frac{1}{2}$. Then x_0 is an efficient solution for this problem.

Remark 1. As particular cases of theorem 2, if the functions $\eta_i,\ i = \overline{1, N}$ and $\theta_j,\ j \in J(x_0)$ are equal to a same function η and by using the usual Karush-Kuhn-Tucker condition:

(i) with the condition (a), we obtain the theorem 3.6 of Kaul et al. [18]. If further there exist $(N + k)$ positive real-valued functions $\alpha_i,\ i = \overline{1, N},\ \beta_j,\ j = \overline{1, k}$

defined on $X \times D$ such that in the definition 6 the implication (5) remains true when multiplying $\mu_i[f_i(x) - f_i(x_0)]$ by $\alpha_i(x, x_0)$ and the implication (8) remains true when multiplying $\lambda_j g_j(x_0)$ by $\beta_j(x, x_0)$, we obtain the theorem 3.1 of Hanson et al. [16].

(ii) with the condition (b), we obtain the theorem 3.4 of Kaul et al. [18]. If further there exist $(N + k)$ positive real-valued functions $\alpha_i,\ i = \overline{1, N},\ \beta_j,\ j = \overline{1, k}$ defined on $X \times D$ such that in the definition 4 the implication (5) remains true when multiplying $\mu_i[f_i(x) - f_i(x_0)]$ by $\alpha_i(x, x_0)$ and the implication (6) remains true when multiplying $\lambda_j g_j(x_0)$ by $\beta_j(x, x_0)$, we obtain the theorem 3.3 of Hanson et al. [16].

(iii) with the condition (c), and if there exist $(N + k)$ positive real-valued functions $\alpha_i,\ i = \overline{1, N},\ \beta_j,\ j = \overline{1, k}$ defined on $X \times D$ such that in the definition 5 the implication (7) remains true when multiplying $\mu_i[f_i(x) - f_i(x_0)]$ by $\alpha_i(x, x_0)$ and the implication (8) remains true when multiplying $\lambda_j g_j(x_0)$ by $\beta_j(x, x_0)$, we obtain the first case of theorem 3.4 of Hanson et al. [16].

Theorem 3. *(Karush-Kuhn-Tucker type sufficient optimality conditions) Let x_0 be a feasible solution for (VP) and suppose that there exist $(N + J)$ vector functions $\eta_i : X \times X \to \mathbb{R}^n,\ i = \overline{1, N},\ \theta_j : X \times X \to \mathbb{R}^n,\ j \in J(x_0)$ and scalars $\mu_i > 0,\ i = \overline{1, N},\ \lambda_j \geqq 0,\ j \in J(x_0)$ such that the generalized Karush-Kuhn-Tucker condition (9) is satisfied. Moreover, assume that one of the following conditions is verified:*

(a) the problem (VP) is V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$ and $(\theta_j)_{j \in J(x_0)}$;
(b) the problem (VP) is pseudo quasi V-type I at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$, $(\theta_j)_{j \in J(x_0)}$ and for μ and λ;
(c) the problem (VP) is semi strictly-pseudo V-type I in g at x_0 with respect to $(\eta_i)_{i=\overline{1,N}}$, $(\theta_j)_{j \in J(x_0)}$ and for μ and λ.

Then x_0 is a properly efficient solution for (VP).

Proof. By condition (a), for all $x \in X$, we have

$$\sum_{i=1}^{N} \mu_i f_i(x) - \sum_{i=1}^{N} \mu_i f_i(x_0) \overbrace{\geqq}^{(3)} \sum_{i=1}^{N} \mu_i [\nabla f_i(x_0)]^t \eta_i(x, x_0) \overbrace{\geqq}^{(9)} - \sum_{j \in J(x_0)} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0)$$
$$\overbrace{\geqq}^{(4)} \sum_{j \in J(x_0)} \lambda_j g_j(x_0) = 0.$$

Thus $\sum_{i=1}^{N} \mu_i f_i(x) \geqq \sum_{i=1}^{N} \mu_i f_i(x_0)$ for all $x \in X$ with $\mu > 0$. Hence, from theorem 1 of Geoffrion [12], x_0 is a properly efficient solution for (VP).
By condition (b), from $g_j(x_0) = 0,\ \lambda_j \geqq 0,\ \forall\ j \in J(x_0)$ (in view of definition 7), we obtain

$$\sum_{j \in J(x_0)} \lambda_j [\nabla g_j(x_0)]^t \theta_j(x, x_0) \leqq 0,\ \forall\ x \in X.$$

From the above inequality and the condition (9), it follows that

$$\sum_{i=1}^{N}\mu_i[\nabla f_i(x_0)]^t\eta_i(x,x_0) \geqq 0,\ \forall\ x \in X.$$

By using the relation (7) (in view of definition 7), we deduce that $\sum_{i=1}^{N}\mu_i f_i(x) \geqq \sum_{i=1}^{N}\mu_i f_i(x_0),\ \forall\ x \in X$, and the conclusion follows.

For the proof of part (c), we proceed as in part (b) and using the reverse implication in (8), we get $\sum_{j\in J(x_0)} \lambda_j[\nabla g_j(x_0)]^t\theta_j(x,x_0) < 0,\ \forall\ x \in X \setminus \{x_0\}$. In the same way as in (b), we get $\sum_{i=1}^{N}\mu_i f_i(x) \geqq \sum_{i=1}^{N}\mu_i f_i(x_0),\ \forall\ x \in X$ and it follows that x_0 is properly efficient for (VP). This completes the proof.

In order to illustrate the obtained result, we shall give an example of multiobjective optimization problem in which the properly efficient solution will be obtained by the application of theorem 3, whereas it will be impossible to apply for this purpose the theorem 3.1 of Kaul et al. [18].

Example 9. We reconsider the multiobjective optimization problem given in example 1.

- We have: the problem is not V-type I at $x_0 = \frac{\pi}{6}$ with respect to a same function η because there exists no a function $\eta :]0, \frac{\pi}{6}]\times\ ]0, \frac{\pi}{6}] \to \mathbb{R}$ for which the functions f_1 and f_2 are both invex at x_0, as it can be seen by taking $x = \frac{\pi}{12}$, then the theorem 3.1 of Kaul et al. [18] is not applicable.
- However, the hypothesis of theorem 3 are verified. In fact: the condition (9) is satisfied for $(\eta_i)_{i=1,2}$ and θ given in the example 1 and $\mu_1 = \frac{1}{10},\ \mu_2 = \frac{9}{10},\ \lambda = \frac{1}{50}$; the problem is V-type I at $x_0 = \frac{\pi}{6} \in X$ with respect to the same $(\eta_i)_{i=1,2}$ and θ. It follows that, x_0 is a properly efficient solution for the given multiobjective optimization problem.

Now, we shall give example of multiobjective optimization problem in which a properly efficient solution will be obtained by the application of theorem 3, whereas it will be impossible to apply for this purpose the sufficient optimality conditions using the usual Karush-Kuhn-Tucker condition.

Example 10. We consider the following multiobjective optimization problem

$$\begin{array}{l} \text{Minimize } f(x) = (-x_1, x_2^2 - x_1), \\ \text{subject to } \ g_1(x) = x_1^3 - x_2 \leqq 0 \\ \qquad\qquad\ g_2(x) = x_2 \leqq 0, \end{array} \tag{15}$$

where $f : \mathbb{R}^2 \to \mathbb{R}^2$ and $g = (g_1, g_2) : \mathbb{R}^2 \to \mathbb{R}^2$. The set of feasible solutions of problem is $X = \{x = (x_1, x_2) \in \mathbb{R}^2 : x_1^3 - x_2 \leqq 0 \text{ and } x_2 \leqq 0\}$.

- We have $x_0 = (0,0) \in X$ is not a vector Karush-Kuhn-Tucker point of problem (15), because the condition of Karush-Kuhn-Tucker at x_0 takes the form $\mu_1 \nabla f_1(x_0) + \mu_2 \nabla f_2(x_0) + \lambda_1 \nabla g_1(x_0) + \lambda_2 \nabla g_2(x_0) = (-\mu_1 - \mu_2, -\lambda + \lambda) \neq (0,0),\ \forall\ (\mu_1, \mu_2) \geq 0,\ \forall\ (\lambda_1, \lambda_2) \geqq 0$, then the known sufficient optimality conditions using this concept, for example from [4, 5, 13, 16–18, 24–26, 33] are not applicable.
- However, using the theorem 3, we have: there exist vector functions $\eta_1(x, x_0) = (x_1, x_1),\ \eta_2(x, x_0) = (x_1, x_2),\ \theta_1(x, x_0) = (x_1, -x_2),\ \theta_2(x, x_0) = (x_2, x_1)$ and scalars $\mu_1 = \mu_2 = \lambda_2 = \frac{1}{2},\ \lambda_1 = 0$ such that the generalized Karush-Kuhn-Tucker condition (9) is satisfied and the problem (15) is V-type I at x_0 with respect to $(\eta_i)_{i=1,2}$, $(\theta_j)_{j=1,2}$, $\mu = (\mu_1, \mu_2)$ and $\lambda = (\lambda_1, \lambda_2)$ (the problem (15) is, in fact, pseudo quasi V-type I and (semi) strictly-pseudo V-type I (in g) at x_0 with respect to the same $(\eta_i)_{i=1,2}$, $(\theta_j)_{j=1,2}$, μ and λ). It follows that, by theorem 3, x_0 is a properly efficient solution for the given multiobjective optimization problem.

In the example 10,we show that the hypothesis of x_0 to be a vector Karush-Kuhn-Tucker point is sometimes a strong sufficient condition and it is not indispensable to prove that x_0 is a properly efficient solution of (VP). In this way, the obtained optimality conditions may be considered as an extension of previously known results.

Example 11. The hypothesis of theorem 3 (with the condition (c)) are satisfied for the problem given in the example 3 at $x_0 = \frac{\pi}{3}$ with respect to the same functions $(\eta_i)_{i=1,2}$, θ and for $\mu_1 = \frac{3}{4}$ and $\mu_2 = \lambda = \frac{1}{4}$. Then x_0 is a properly efficient solution for problem.

Remark 2. As particular cases of theorem 3, if the functions $\eta_i,\ i = \overline{1,N}$ and $\theta_j,\ j \in J(x_0)$ are equal to a same function η and by using the usual Karush-Kuhn-Tucker condition:

(i) with the condition (a), we obtain the theorem 3.1 of Kaul et al. [18].
(ii) with the condition (b), we obtain the theorem 3.5 of Kaul et al. [18]. If further there exist $(N+k)$ positive real-valued functions $\alpha_i,\ i = \overline{1,N},\ \beta_j,\ j = \overline{1,k}$ defined on $X \times D$ such that in the definition 7 the implication (7) remains true when multiplying $\mu_i[f_i(x) - f_i(x_0)]$ by $\alpha_i(x, x_0)$ and the implication (6) remains true when multiplying $\lambda_j g_j(x_0)$ by $\beta_j(x, x_0)$ and if also there exist positive real numbers n_i and m_i such that $n_i < \alpha_i(x, x_0) < m_i$ for each $x \in X$ and for all $i = \overline{1,N}$, we obtain the second case of theorem 3.2 of Hanson et al. [16].
(iii) with the condition (c), if there exist $(N+k)$ positive real-valued functions $\alpha_i,\ i = \overline{1,N},\ \beta_j,\ j = \overline{1,k}$ defined on $X \times D$ such that in the definition 5 the implication (7) remains true when multiplying $\mu_i[f_i(x) - f_i(x_0)]$ by $\alpha_i(x, x_0)$ and the implication (8) remains true when multiplying $\lambda_j g_j(x_0)$ by $\beta_j(x, x_0)$ and if also there exist positive real numbers n_i and m_i such that $n_i < \alpha_i(x, x_0) < m_i$ for each $x \in X$ and for all $i = \overline{1,N}$, we obtain the second case of theorem 3.4 of Hanson et al. [16].

4 Conclusion

In this paper, we have defined new classes of problems called V-type I, quasi-, pseudo-, pseudo quasi-, quasi pseudo- V-type I with respect to $(\eta_i)_i$ and $(\theta_j)_j$, as a generalization of invex problems with respect to a same function η. In the setting of these definitions, we have established new Karush-Kuhn-Tucker type necessary and sufficient optimality conditions for a feasible point to be efficient or properly efficient. We illustrated these optimality results with some examples and we have shown that the obtained results allow to prove that a feasible point is an efficient or properly efficient solution even if it is not an usual vector Karush-Kuhn-Tucker point for a multiobjective programming problem. Known results in the literature (Hanson et al. 2001; Kaul et al. 1994) can be deduced as particular cases from the obtained results, when the functions $(\eta_i)_i$ and $(\theta_j)_j$ are equal to a same function η. However, the concept of invexity with respect to different functions η_i may be extended in different directions of the field of multiobjective programming. It may be used, with and without differentiability assumption, in the framework of fractional programming, variational problems, symmetric duality, game theory, etc.

References

1. B. Aghezzaf and A. Hachimi (2000). Generalized Invexity and Duality in Multiobjective Programming Problems. J. Global Optim. **18**: 91-101.
2. T. Antczak (2003). A class of B-(p,r)-invex functions and mathematical programming. J. Math. Anal. Appl. **286**: 187-206.
3. T. Antczak (2002). Multiobjective programming under d-invexity. Eur. J. Oper. Res. **137**: 28-36.
4. T. Antczak (2001). Multiobjective programming with (p,r)-invexity. Zeszyty Naukowe Politechniki Rzeszowskiej Nr 190, Matematyka, **z.25**: 5-30.
5. M. Arana-Jiménez, A. Rufián-Lizana, R. Osuna-Gómez, and G. Ruiz-Garzón (2008). Pseudoinvexity, optimality conditions and efficiency in multiobjective problems; duality. Nonlinear Anal-Theor **68(1)**: 24-34.
6. M.S. Bazaraa, H.D. Sherali, and C.M. Shetty (2006). Nonlinear Programming: Theory and Algorithms. Wiley, New York, Third Edition.
7. C.R. Bector, S.K. Sunedja, and S. Gupta (1992). Univex functions and univex nonlinear programming. In "Proceedings of the Administrative Sciences Association of Canada" 115-124.
8. C.R. Bector, S.K. Suneja, and C.S. Lalitha (1993). Generalized B-vex functions and generalized B-vex programming. J. Optim. Theory Appl. **76**: 561-576.
9. A. Chinchuluun and P.M. Pardalos (2007). A survey of recent developments in multiobjective optimization. Ann. Oper. Res. **154**: 29-50.
10. A. Chinchuluun, D. Yuan, and P.M. Pardalos (2007). Optimality conditions and duality for nondifferentiable multiobjective fractional programming with generalized convexity. Ann. Oper. Res. **154**: 133-147.
11. C. Fulga and V. Preda (2009). Nonlinear programming with E-preinvex and local E-preinvex functions. Eur. J. Oper. Res. **192**: 737-743.
12. A.M. Geoffrion (1968). Proper efficiency and the theory of vector maximisation. J. Math. Anal. Appl.J **22**: 618-630.

13. M. Hachimi and B. Aghezzaf (2004). Sufficiency and duality in differentiable multiobjective programming involving generalized type I functions. J. Math. Anal. Appl. **296**: 382-392.
14. M.A. Hanson (1981). On sufficiency of the Kuhn-Tuker conditions. J. Math. Anal. Appl. **80**: 445-550.
15. M.A. Hanson and B. Mond (1987). Necessary and sufficient conditions in constrained optimization. Math. Programming **37**: 51-58.
16. M.A. Hanson, R. Pini and C. Singh (2001). Multiobjective programming under generalized type I invexity. J. Math. Anal. Appl. **261**: 562-577.
17. V. Jeyakumar and B. Mond (1992). On generalized convex mathematical programming. J. Austral. Math. Soc. Ser. **B 34**: 43-53.
18. R.N. Kaul, S.K. Suneja, and M.K. Srivastava (1994). Optimality criteria and duality in multiple-objective optimization involving generalized invexity. J. Optim. Theory Appl. **80 (3)**: 465-482.
19. J.C. Lee and H.C. Lai (2005). Parameter-free dual models for fractional programming with generalized invexity. Ann. Oper. Res. **133**: 47-61.
20. T. Maeda (1996). Multiobjective decision making theory and economic analysis. Makino-Syoten.
21. T. Maeda(2004). Second-order conditions for efficiency in nonsmooth multiobjective optimization problems. J. Optimiz. Theory App. **122(3)**: 521-538.
22. O.L. Mangasarian (1969). Nonlinear Programming. McGraw–Hill, New York.
23. D.H. Martin (1985). The essence of invexity. J. Optim. Theory Appl. **47**: 65-76.
24. S.K. Mishra (1998). On multiple objective optimization with generalized univexity. J. Math. Anal. Appl. **224**: 131-148.
25. S.K. Mishra, S.Y. Wang, and K.K. Lai (2005). Optimality and duality for multiple-objective optimization under generalized type I univexity. J. Math. Anal. Appl. **303**: 315-326.
26. R. Osuna-Gómez, A. Beato-Morero, and A. Rufián-Lizana (1999). Generalized convexity in multiobjective programming. J. Math. Anal. Appl. **233**: 205-220.
27. R. Pini and C. Singh (1997). A survey of recent [1985-1995] advances in generalized convexity with applications to duality theory and optimality conditions. Optim. **39 (4)**: 311-360.
28. N.G. Rueda and M.A. Hanson (1988). Optimality criteria in mathematical programming involving generalized invexity. J. Math. Anal. Appl. **130**: 375-385.
29. N.G. Rueda, M.A. Hanson and C. Singh (1995). Optimality and duality with generalized convexity. J. Optimiz. Theory App. **86 (2)**: 491-500.
30. IM. Stancu-Minasian (2006). Optimality and duality in nonlinear programming involving semilocally B-preinvex and related functions. Eur. J. Oper. Res. **173**: 47-58.
31. H. Slimani and M.S. Radjef (2009). Duality for nonlinear programming under generalized Kuhn-Tucker condition. Journal of Optimization: Theory, Methods and Applications (IJOTMA) **1 (1)**: 75-86.
32. H. Slimani and M.S. Radjef (2010). Nondifferentiable multiobjective programming under generalized d_I-invexity. Eur. J. Oper. Res. **202**: 32-41, doi:10.1016/j.ejor.2009.04.018.
33. T. Weir (1988). A note on invex functions and duality in multiple objective optimization. Opsearch, 25, 98-104.
34. P.L. Yu (1985). Multicriteria decision making: Concepts, Techniques and Extensions. Plenum Press, New York.

Une Approche pour l'accélération de la génération de colonnes appliquées au problème de rotations d'équipages

Abdelkader LAMAMRI[(1)], Hacène AIT HADDADENE[(2)], Anass NAGIH[(3)]

[(1)]Université de Blida, Algérie.

[(2)]Laboratoire LAID3, USTHB, Algérie.

[(3)]Laboratoire LITA, Université Paul Verlaine – Metz, France.

[(1)]prslamamri_inch@yahoo.fr , [(2)]aithaddadenehacene@yahoo.fr, [(3)]anass.nagih@univ-metz.fr

Abstract: We are interested in problems from combinatorial optimization, more precisely, the problem of construction crew rotations with resource constraints. The problem is to cover the cost of all flights of the company. Given the large size of the problems encountered in industry, these models are solved by an approach based on column generation that can handle implicitly all feasible solutions and a master problem determining the best solution. We propose in this paper an approach to improve the acceleration of the method of column generation for solving the problem of construction crew rotations, it is projected in each arc, the resources a vector of size smaller by using a Lagrangean relaxation algorithm to determine the coefficients of the projection arc combined with an algorithm for re-optimization, then generates a sub-set of complementary solutions to the master problem. The preliminary experiments of our technique gave good results on instances of random rotation of crews.

Keywords: Combinatorial optimization, mathematical programming, column generation,

Résumé :

Nous nous sommes intéressés aux problèmes issus de l'optimisation combinatoire, plus précisément, le problème de construction des rotations d'équipages avec contraintes de ressources. Le problème consiste à couvrir au moindre coût l'ensemble des vols de la compagnie. Etant donné la grande taille des problèmes rencontrés dans l'industrie, ces modèles sont résolus par une approche basée sur la génération de colonnes qui permet de gérer implicitement l'ensemble des solutions réalisables et un problème maître déterminant la meilleure solution. Nous proposons dans cet article une approche permettant d'améliorer l'accélération de la méthode de la génération de colonnes pour la résolution de problème de construction des rotations d'équipages, il s'agit de projeter, en chaque arc, les ressources sur un vecteur de dimension inférieure en utilisant un algorithme de relaxation lagrangienne pour déterminer les coefficients de la projection par arc combinée à un algorithme de ré-optimisation, puis génère un sous-ensemble des solutions complémentaires vers le problème maître. Les expérimentations préliminaires de notre technique ont donné de bons résultats sur des instances aléatoires de rotations d'équipages.

Mots clés : Optimisation combinatoire, programmation mathématique, Génération de colonnes,

1. Introduction :

Dans l'industrie du transport aérien, l'optimisation et l'automatisation de la construction des rotations d'équipages est un enjeu financier et organisationnel majeur. Le problème consiste à couvrir au moindre coût l'ensemble des vols de la compagnie, programmés sur un horizon de temps donné, par des équipages formés de personnel de cockpit (pilotes, co-pilotes) et de personnel de cabine (hôtesses, stewards). À périodicité de plusieurs jours (de l'ordre de la semaine), chaque équipage part de la base à laquelle il est affecté, enchaîne un certain nombre de vols et revient à la base. Cette séquence de vols avec retour à la base est appelé *rotation*. La construction des rotations d'une compagnie aérienne est extrêmement contrainte par les réglementations internationale, nationale et interne du travail, et par la disponibilité limitée des ressources.
Ces contraintes rendent le problème particulièrement difficile à résoudre. L'utilisation de modèles et des logiciels d'optimisation pour ce problème permet aux grandes compagnies d'effectuer des gains financiers substantiels. Il n'est pas rare qu'une réduction d'un pourcent sur le coût total des rotations se traduise par plusieurs dizaines de millions de dollars d'économie pour les grandes compagnies [1], d'où une recherche, fondamentale et appliquée, foisonnante sur le sujet. Le problème général de Construction des Rotations avec Contraintes des Ressources (PCR-CR) peut se formuler comme un problème de flot réalisable à coût minimum dans un réseau multiple, avec ajout des variables et des contraintes de ressources. Notons enfin que les contraintes de ressources rendent le problème (PCR-CR) NP-difficile. Les modalités de construction du réseau des rotations admissibles, de calcul du coût des rotations ainsi que le Programme Mathématique associé sont présentées en section 2. La section 3 donne un aperçu sur les techniques classiques de résolution, notamment la méthode de génération de colonnes, dont le sous-problème associé est traité dans cette section, en plus de notre contribution. La section 4 présente notre Algorithme. L'application fera sur des instances de rotations d'équipages qui seront présenté dans la section 5. La section 6 étude le cas au la solution obtenue est fractionnaire. Afin la section 7 conclut le document.

2. Présentation du problème

L'ensemble des vols à couvrir par les équipages est noté $V = \{1, \dots, n\}$. Le programme de vols et les horaires associés sont établis de façon quasi-certaine sur la période considérée, de l'ordre du mois ou de la semaine selon la taille de la compagnie.
Le terme de *vol* associé a chaque élément $i \in V$est dans certains cas abusif, dans la mesure où i peut en réalité représenter une séquence de vols agrégée et insécable, i.e., ne pouvant être couverte que par un seul et même équipage dans sa totalité. Souvent également, la tâche à couvrir par un équipage n'est pas le seul vol mais un service de vol pouvant débuter avant et terminer après le vol proprement dit, pour pouvoir inclure le temps de préparation de l'avion et le temps d'accompagnement des passagers, par exemple. Cependant, nous maintiendrons cette terminologie de *vol*, par souci de lisibilité. On connaît pour chaque vol $i \in V$:

(i) l'horaire de départ $t^{\nearrow}(i)$,
(ii) l'horaire d'arrivée $t^{\searrow}(i)$,
(iii) l'aéroport de départ $a^{\nearrow}(i)$,
(iv) l'aéroport d'arrivée $a^{\searrow}(i)$.

Une rotation doit débuter et terminer à l'une des bases de la compagnie. L'ensemble $\mathcal{B}$ des bases est généralement composé de grandes plates-formes d'interconnexion appelées *hubs*. Le problème de construction de rotations dans le transport aérien comporte le plus souvent des

contraintes de ressources sur les rotations. Afin de prendre en compte ces contraintes, valables pour chaque rotation prise individuellement, une modélisation classique associe à chaque équipage un sous-réseau construit de la façon suivante.

2.1 Construction des sous-réseaux

L'ensemble des équipages susceptibles d'être utilisés pour couvrir les vols de la compagnie est indicé par $k \in \mathcal{K} = \{1, \ldots, K\}$. Pour $k \in \mathcal{K}$, $b^k \in \mathcal{B}$ désigne la base de départ et d'arrivée de l'équipage k. Un graphe $G^k = (X^k, A^k)$est alors associé à l'équipage $k \in \mathcal{K}$, où X^kdésigne l'ensemble des nœuds du réseau et A^k l'ensemble des arcs. L'ensemble X^k se décompose en trois sous-ensembles :

$$X^k = \{o^k\} \cup V^k \cup \{d^k\}$$

où, pour $k \in \mathcal{K}$:
- l'origine o^k (resp. la destination d^k) désigne la source (resp. le puits) du réseau G^k,
- $V^k \subseteq V$ désigne l'ensemble des vols programmés par la compagnie pouvant être couverts par l'équipage k.

L'ensemble A^k des arcs du réseau se décompose de la façon suivante :

$$A^k = \mathcal{O}V^k \cup VV^k \cup V\mathcal{D}^k \cup \{(o^k, d^k)\}$$

Ou

$$\mathcal{O}V^k = \{(o^k, i) : i \in V^k, a^{\nearrow}(i) = b^k\}$$
$$VV^k \subseteq \mathcal{U}^k = \{(i,j) \in V^k \times V^k : a^{\searrow}(i) = a^{\nearrow}(j), t^{\searrow}(j) \geq t^{\nearrow}(i) + t_{min}(i,j)\}$$
$$V\mathcal{D}^k = \{(i, d^k) : i \in V^k, a^{\searrow}(i) = b^k\}$$

Le passage par l'arc (o^k, d^k) signifiera que l'équipage k ne sera pas utilisé. $\mathcal{U}^k$ est l'ensemble des couples de vols (i,j) remplissant les conditions nécessaires évidentes rendant possible l'enchaînement de deux vols par le même équipage :
(i) l'aéroport d'arrivée du vol i est l'aéroport de départ du vol j,
(ii) l'heure de départ du vol j est postérieure à l'heure d'arrivée du vol i, d'un écart supérieur ou égal à une valeur $t_{min}(i,j)$, fixée par la compagnie ou par les contraintes de transit de l'aéroport.
Généralement, $VV^k \neq \mathcal{U}^k$ car la réglementation du travail de la compagnie impose un certain nombre de contraintes supplémentaires (créneaux repas, pauses, découchés) restreignant les possibilités de connexion entre vols. De plus, des contraintes globales, variant d'une compagnie à l'autre, sont généralement associées à chaque rotation. Citant parmi l'ensemble des contraintes possibles les contraintes de borne suivantes :
- borne inférieure et/ou supérieure sur l'amplitude totale de la rotation,
- borne inférieure et/ou supérieure sur le temps de travail total (temps de travail total = temps de vol total + transferts + pauses légales)
- borne inférieure sur le nombre de jours de repos
- borne inférieure sur le nombre d'heures de repos quotidien

- borne supérieure sur le nombre de vols

- borne supérieure sur le nombre d'heures de travail consécutives.

Ces contraintes peuvent se modéliser par la donnée :
- d'un ensemble $\mathcal{Q} = \{1, \ldots, Q\}$ de ressources,

- de consommations des ressources $t_{i,j}^{k,q}$ associées à chaque arc $(i,j) \in A^k$ et chaque ressource $q \in \mathcal{Q}$,
- de seuils minimaux $a_i^{k,q}$ et maximaux $b_i^{k,q}$ de consommation de ressource $q \in \mathcal{Q}$, à respecter pour chaque équipage $k \in \mathcal{K}$ et en chaque nœud i du réseau ; si les contraintes de ressources portent sur l'ensemble de la rotation, i.e., sur le seul nœud destination d et pas sur les nœuds intermédiaires, dans ce cas $a_i^{k,q} = 0 \; et \; b_i^{k,q} = \infty$ pour tout nœud $i \neq d$.

2.2 Coût des rotations

Le mode de calcul du coût d'une rotation est généralement complexe et varie selon les compagnies. Ce coût peut être une fonction non linéaire de plusieurs paramètres tels que la consommation de ressources, l'amplitude totale et le temps de vol total de la rotation [1][2]. Afin d'établir un modèle générique pour ce chapitre, nous considérerons que la fonction de coût est une approximation linéaire décomposable par équipage $k = 1, \ldots, K$ et par arcs $(i,j) \in A^k$. Le coût de la rotation effectuée par l'équipage $k \in \mathcal{K}$ sera alors la somme des coûts c_{ij}^k associés aux arcs $(i,j) \in A^k$ qui composent cette rotation.

2.3 Modélisation mathématique

Le Problème de Construction des Rotations avec Contraintes des Ressources (PCRCR) peut se modéliser, si la fonction de coût est linéaire, par la Programmation Linéaire en variables mixtes. Nous avons un problème de flot réalisable à coût minimal sur l'ensemble des sous-réseaux, avec variables binaires de flot et variables continues de ressources (PCR-CR) :

$$(PCRCR) \equiv \begin{cases} \min \sum_{k=1}^{K} \sum_{(i,j)\in A^k} c_{ij}^k x_{ij}^k & (1) \\ \sum_{k=1}^{K} \sum_{j:(i,j)\in A^k} x_{ij}^k \geq 1 \;\; pour \;\; i \in V = \{1, \ldots, n\} & (2) \\ \sum_{i:(o^k,i)\in A^k} x_{o^k i}^k = 1 \;\; pour \;\; k \in \mathcal{K} & (3) \\ \sum_{i:(i,d^k)\in A^k} x_{i d^k}^k = 1 \;\; pour \;\; k \in \mathcal{K} & (4) \\ \sum_{i:(i,j)\in A^k} x_{ij}^k = \sum_{l:(j,l)\in A^k} x_{jl}^k \;\; pour \;\; j \in V^k & (5) \\ T_i^{k,q} + t_i^{k,q} - T_j^{k,q} \leq M\left(1 - x_{ij}^k\right) \; pour (i,j) \in A^k \, , k \in \mathcal{K} \, , q \in \mathcal{Q} & (6) \\ a_i^{k,q} \leq T_i^{k,q} \leq b_i^{k,q} \;\; pour \;\; i \in V^k , k \in \mathcal{K} \, , q \in \mathcal{Q} & (7) \\ x_{ij}^k \in \{0,1\} \, , T_i^{k,q} \geq 0 \;\; pour \; (i,j) \in A^k \, , k \in \mathcal{K} \, , q \in \mathcal{Q} & (8) \end{cases}$$

Les variables binaires x_{ij}^k indiquent si la rotation emprunte l'arc $(i,j) \in A^k$
(et donc enchaîne les vols i et j si $(i,j) \in VV^k$), tandis que les variables $T_i^{k,q}$ indiquent la consommation cumulée de chaque ressource q à chaque nœud i d'un réseau G^k.

L'objectif (1) minimise le coût total des rotations. Les contraintes (2) expriment la couverture de chaque vol par au moins un équipage si un seul équipage est autorisé par vol la contrainte est à l'égalité. Les contraintes $(3-5)$ définissent une structure de chemin dans le sous-réseau G^k : passage d'un flux d'une unité $(3\ ou\ 4)$ et conservation du flux aux sommets (5). Les contraintes $(6-7)$ sont les contraintes de ressources associées à chaque rotation. La contrainte (6), dans laquelle $M > 0$ est un paramètre très grand, peut aussi se trouver sous la forme non linéaire suivante :

$$x_{ij}^k \left(T_i^{k,q} + t_i^{k,q} - T_j^{k,q}\right) \leq 0 \quad (i,j) \in A^k\ , k \in \mathcal{K}\ , q \in \mathcal{Q} \qquad (9)$$

L'inégalité dans (6) ou (9) stipule que l'attente est permise pour l'équipage, dans le cas contraire la contrainte s'écrit à l'égalité. Cette contrainte permet d'obtenir la consommation cumulée de ressource q au nœud j, puisqu'on a :

$$T_j^{k,q} = \max(a_i^{k,q}, T_i^{k,q} + t_i^{k,q})$$

Les contraintes (7) sont des contraintes de bornes aux noeuds du réseau (fenêtres de temps par exemple). Remarquons que les contraintes $(3-7)$ sont des contraintes locales valables pour le seul sous-réseau G^k.

Seules les contraintes de couverture (2) sont des contraintes globales liant les K sous-réseaux. La relaxation de ces contraintes liantes et la décomposition du problème initial par sous-réseau sera donc une option de résolution intéressante. Notons enfin que les contraintes de ressources $(6-7)$ rendent le problème (PCR-CR) NP-difficile. Même le problème de réalisabilité associé est NP-complet [10].

3. Approches de résolution

3.1 Principes de décomposition

On distingue deux types de contraintes dans le système $(2)-(7)$:

(i) les contraintes de couverture (2), dites liantes ou globales, qui lient l'ensemble des équipages $k = 1, \ldots, K$,

(ii) les contraintes $(3)-(7)$ propres à chaque équipage $k \in \{1, \ldots, K\}$ et définissant un itinéraire légal.

La matrice associée aux contraintes $(3)-(7)$ étant bloc-diagonale, et l'objectif (1) étant séparable (car linéaire), la résolution de la relaxation continue de ce modèle peut être basée sur la décomposition de Dantzig-Wolfe. Dans ce type de décomposition, les contraintes $(3)-(7)$ définissent K sous-problèmes indépendants et les contraintes globales (2) sont conservées dans le problème maître. Dans un schéma de type génération de colonnes, il s'agit de résoudre alternativement le problème maître et les K sous-problèmes. Pour obtenir une solution entière, ce schéma peut être appliqué au niveau de chaque nœud de l'arbre de recherche. La difficulté majeure réside dans la résolution des sous-problèmes dont les espaces des états peuvent augmenter de façon exponentielle avec le nombre de ressources Q, rendant incontournable l'utilisation d'heuristiques. D'autre part, la convergence du schéma de génération de colonnes étant sensible à la qualité des solutions fournies par la résolution de ces sous-problèmes, la résolution effective d'instances réelles issues de l'industrie nécessite de trouver un bon compromis entre la qualité des solutions et le temps de résolution des sous-problèmes. Dans ce qui suit, nous détaillons le principe général de la génération de colonnes pour le problème$(PCRCR)$.

3.2 Génération de colonnes, problème maître et sous-problème

Les méthodes de génération de colonnes [3] ont été appliquées avec succès aux problèmes de construction de rotations [2], [5]. Dans cette approche, le problème maître est reformulé par un Problème de Couverture (PC) (*Set Covering* ou *Set Partitioning* selon que la contrainte de couverture des vols est une inégalité ou à l'égalité) :

$$(PC) \equiv \begin{cases} \min \sum_{r \in \mathcal{R}} c_r x_r & (10) \\ \sum_{r \in \mathcal{R}} a_{ir} x_r \geq 1 \; pour \;\; i \in V = \{1, \ldots\ldots, n\} & (11) \\ x_r \in \{0,1\} \quad pour \quad r \in \mathcal{R} & (12) \end{cases}$$

Où $\mathcal{R}$ désigne l'ensemble des rotations admissibles satisfaisant les contraintes de ressources et d'enchaînement entre vols, c_r représente le coût de la rotation $r \in \mathcal{R}$, $a_{ir} = 1$ si et seulement si la rotation r couvre le vol i, et la variable binaire x_r indique le choix ou non de la rotation r dans la solution.

On note $(\overline{PC})$ la relaxation continue du problème (PC) où les contraintes d'intégrité (12) sont remplacées par $x_r \geq 0$ pour $r \in \mathcal{R}$. Le nombre total de rotations admissibles $|\mathcal{R}|$ étant généralement une fonction exponentielle du nombre $n = |V|$ de vols à couvrir, l'énumération complète de $\mathcal{R}$ est à proscrire. Pour autant, il est possible de trouver en un temps raisonnable une solution optimale de $(\overline{PC})$ en ne générant qu'un sous-ensemble restreint de rotations (i.e., de colonnes de la matrice de contraintes). Le principe est le suivant : Soit $\mathcal{R}^0$ une solution réalisable pour (PC),comprenant un nombre restreint de rotations de $\mathcal{R}$, et généré par une heuristique quelconque. Nous pouvons résoudre par la Programmation Linéaire (par exemple, par l'algorithme du Simplexe) le programme $(\overline{PC}^0)$, qui est la restriction de $(\overline{PC})$ au sous-ensemble de rotations $\mathcal{R}^0$. Cette résolution fournit également un vecteur de multiplicateurs ou de variables duales $(\delta_1^0, \ldots\ldots, \delta_n^0)$ associé aux n vols à couvrir. Le critère d'optimalité selon lequel toutes les rotations sont de coût réduit positif à l'optimum, conduit à rechercher la rotation de plus faible coût réduit négatif, soit

$$r^0 = arg \min_{r \in \mathcal{R}} \left(c_r - \sum_{i=1}^{n} \delta_1^0 a_{ir} \right) \qquad (13)$$

Si l'on parvient à trouver en temps raisonnable cette rotation r^0 , on peut alors relancer la résolution du programme de couverture $(\overline{PC})$ sur l'ensemble $\mathcal{R}^1 = \mathcal{R}^0 \cup \{r^0\}$, en ajoutant la colonne a_{r^0} à la matrice des contraintes. De façon générale, on résout à chaque itération t le problème maître restreint $(\overline{PC}^t)$:

$$(\overline{PC}^t) \equiv \begin{cases} \min \sum_{r \in \mathcal{R}^t} c_r x_r & (14) \\ \sum_{r \in \mathcal{R}^t} a_{ir} x_r \geq 1 \; pour \;\; i \in V = \{1, \ldots\ldots, n\} & (15) \\ x_r \geq 0 \quad pour \quad r \in \mathcal{R}^t & (16) \end{cases}$$

tel que $\mathcal{R}^t = \mathcal{R}^{t-1} \cup \{r^{t-1}\}$

où, si δ^{t-1} désigne le vecteur de multiplicateurs associé aux n vols dans la résolution de $(\overline{PC}^{t-1})$, la rotation r^{t-1} de plus faible coût réduit négatif est définie par

$$r^{t-1} = arg\min_{r\in\mathcal{R}}\left(c_r - \sum_{i=1}^{n}\delta_i^{t-1}a_{ir}\right) \qquad (17)$$

Le terme de *génération de colonnes* provient de l'ajout de la colonne a_{r^t} à la matrice des contraintes du problème maître, à chaque itération t. Ce processus de résolution itérative du problème maître $(14-16)$ et du sous-problème (17) est stoppé dés que toutes les rotations sont de coût réduit positif dans la résolution du sous problème , signe que l'optimum continu est atteint,

- soit à l'itération s telle que :

$$\min_{r\in\mathcal{R}}\left(c_r - \sum_{i=1}^{n}\delta_i^{s}a_{ir}\right) \geq 0$$

Une variante de cette méthode, permettant d'accélérer le processus en pratique [6], consiste à ajouter à chaque itération un sous-ensemble de rotations complémentaires de coût réduit négatif au lieu de la seule meilleure rotation du sous-problème (17)(voir [7]). La taille maximale souhaitée de ce sous-ensemble de colonnes entrantes pourra être paramétrée de manière à évoluer au cours de l'algorithme. La complexité globale de la méthode est fortement dépendante de la complexité du sous-problème, que les contraintes de ressources rendent NP-difficile. Il est souvent possible cependant de le résoudre en un temps raisonnable grâce à une énumération *implicite* de $\mathcal{R}$, en exploitant la structure de graphe du sous-problème et en appliquant des variantes d'algorithmes de plus court chemin.

3.3 Résolution du sous-problème pour la génération de colonnes

Notant que dans le cas de plusieurs sous réseaux $k = 1, \ldots\ldots, K$, la résolution du sous problème(17)étant décomposable par sous réseaux, on omettra l'indice k et le graphe du sous problème sera noté $G = (\{o\} \cup V \cup \{d\}, A)$.

Le problème de plus court chemin avec contraintes des ressources $(PCC - CR)$, se formule comme suit :

$$(PCC-CR) \equiv \begin{cases} \min \sum_{(i,j)\in A} c_{ij}x_{ij} & (18)\\ \sum_{i:(o,i)\in A} x_{oi} = 1 & (19)\\ \sum_{i:(i,d)\in A} x_{id} = 1 & (20)\\ \sum_{i:(i,j)\in A} x_{ij} = \sum_{i:(i,j)\in A} x_{ji} \quad pour \quad j \in V = \{1, \ldots\ldots, n\} & (21)\\ x_{ij}\left(T_i^q + t_i^q - T_j^q\right) \leq 0 \quad pour \quad (i,j) \in A\,, q \in Q & (22)\\ a_i^q \leq T_i^q \leq b_i^q \; pour \;\; i \in V, q \in Q & (23)\\ x_{ij} \in \{0,1\}\,, T_i^q \geq 0 \; pour \; (i,j) \in A\,, q \in Q & (24) \end{cases}$$

Pour résoudre ce problème, Desrochers et Soumis [9] proposent un algorithme de programmation dynamique du type pulling.

Définition 1 A chaque chemin de l'origine o au nœud j, on associe une étiquette $E(\mathrm{C_j}, \mathrm{T_j}) = E(\mathrm{C_j}, \mathrm{T_j^1}, \ldots., \mathrm{T_j^Q})$ représentant l'état de ses ressources et son coût.

Définition 2 Soient $E(\mathrm{C_j}, \mathrm{T_j})$ et $E'(\mathrm{C'_j}, \mathrm{T'_j})$deux étiquettes associées à deux chemins réalisables Pet P'de o à j. On dit que $E(\mathrm{C_j}, \mathrm{T_j})$ domine $E'(\mathrm{C'_j}, \mathrm{T'_j})$, (resp. Pdomine P'), et on note $E(\mathrm{C_j}, \mathrm{T_j}) \leq E(\mathrm{C'_j}, \mathrm{T'_j})$, (resp. $P \leq P'$) si et seulement si $\mathrm{C_j} \leq \mathrm{C'_j}$ et $T_j^q \leq T'^q_j$, $\forall q \in \mathcal{Q}$.

Définition 3 Une étiquette associée a un chemin réalisable de o à j, est dite efficace si elle est minimale au sens de la relation d'ordre $\leq$. Un chemin est dit efficace s'il est associe à une étiquette efficace.

L'algorithme de programmation dynamique (APD) procède en trois grandes étapes. En chaque nœud $j \in V$, il effectue les opérations suivantes :
1. Prolongation des chemins (génération des étiquettes),
2. Filtrage (test de réalisabilité),
3. Dominance (élimination des étiquettes non efficaces).

Pour un nœud j donne, des étiquettes sont créées en prolongeant celles présentes aux nœuds i, tels que $(i,j) \in A$. Ainsi, une nouvelle étiquette $E(\mathrm{C_j}, \mathrm{T_j})$ est donnée par

$$\mathrm{C}_j = \mathrm{C}_i + c_{ij}$$
$$T_j^q = \max\{a_j^q, T_i^q + t_{ij}^q\}, q \in \mathcal{Q}$$

En considérant que tous les prédécesseurs du nœud $j \in V$ sont déjà traités, la dominance au nœud j peut être interprétée comme la détermination des Pareto optimaux du problème multicritère a $|\mathcal{Q}| + 1$ fonctions :

$$\begin{cases} \min\limits_{i;(i,j)\in A} \left(C_i + c_{ij}\ ; \max\{a_j^q, (T_i^q + t_{ij}^q)\}, q \in \mathcal{Q}\right) \\ T_i^q + t_{ij}^q \leq b_j^q \quad , \qquad q \in \mathcal{Q} \end{cases}$$

Soit v^* sa valeur optimale.

La relation de dominance $\leq$ étant une relation d'ordre partiel, le nombre d'étiquettes efficaces à traiter augmente de façon exponentielle en fonction du nombre de ressources, ce qui rend la procédure de prolongation très ardue.
Dans un récent travail Nagih et Soumis [4] proposent une méthode d'agrégation des ressources pour les PCC-CR par projection, en chaque nœud en utilisant simultanément un algorithme de programmation dynamique et une relaxation lagrangienne. Cependant, dans le cas général, cette approche ne donne pas toujours l'optimalité comme le montre l'exemple1 suivant.

Exemple1

Considérant le problème de plus court chemin avec une contrainte de ressource, représenté dans la figure 1. La solution optimale globale correspond au l'étiquette $\mathrm{E_3}$ (10,6).

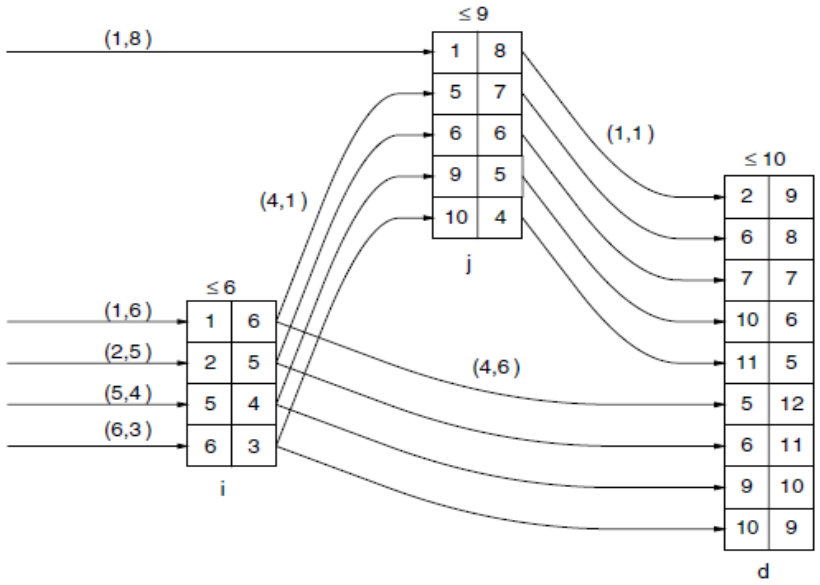


Figure 1 : Exemple 1-ensemble des étiquettes.

Comme on peut le voir sur la figure 2,

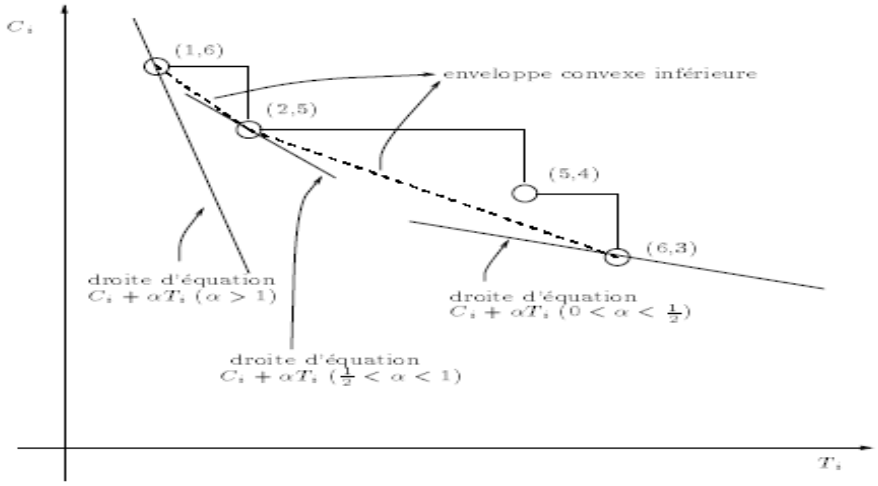


Figure 2 : Exemple 1-représentation des étiquettes au nœud 1 dans le plan (coût, ressource).

La dominance par rapport à la combinaison $(coût + u_j \times ressource)$ au nœud 1, conduit aux étiquettes $\begin{cases} E_4(6,3)\ si\ u_1 \in \left[0, \frac{1}{2}\right] \\ E_2(2,5)\ si\ u_1 \in \left[\frac{1}{2}, 1\right] \\ E_1(1,6)\ si\ u_1 \in [1, +\infty[\end{cases}$

Il n'existe donc pas de valeur de u_1 permettant de récupérer l'étiquette $E_3(5,4)$ au nœud 1, et par suite la solution optimale globale.

Par contre, en considérant un coefficient de projection qui dépend, en plus du nœud traité, du nœud prédécesseur, il existe toujours une instanciation de ces coefficients qui permet de récupérer la solution optimale globale, comme le montre le théorème 1.

Soit v_{noeud} la meilleure valeur obtenue par l'algorithme de Nagih et Soumis, et v_{arc} la meilleure valeur fournie par la deuxième technique.

Théorème 1

$$\boldsymbol{v^* = v_{arc} \leq v_{noeud}}$$

Preuve : Soit (o, j_1) le premier arc du chemin optimal global. En prenant u_{ij_1} suffisamment grand pour tout $(i, j_1) \in A \; tel \; que \; i \neq o \; et \; u_{oj_1} = 0$, la solution au nœud j_1 correspondra au chemin (o, j_1). Puis on réitère le même procédé jusqu'au nœud d. ⊡

Comme le nombre de coefficients à ajuster sera plus important pour l'approche de projection par arcs, trouver les multiplicateurs optimaux u_{ij}^* nécessiterai plusieurs itérations successives de APD-L (voir l'annexe 1), cette méthode peut s'avérer coûteuse. Afin d'obtenir rapidement de bonnes solutions heuristique (réalisables), notre approche appliqué une seule fois APD-L puis appliquer APD-LND (voir l'annexe 1) ,en utilisant les multiplicateurs u_{ij} trouves afin de produire des colonnes réalisables et de coût marginal négatif. Plus précisément, on choisit d'abord une suite de pas (p_k) telle que le série $(\sum p_k)$ est divergente et $\lim_{k\to\infty} p_k = 0$, autrement dit les conditions qui assurent la convergence de l'algorithme du sous-gradient aussi appelées conditions de Polyak [8]. On applique en premier lieu APD-L en utilisant les multiplicateurs u_{ij}^{k-1}de l'itération précédente, on trouve les sous-gradients $Sg_{ij}^k = t_{ij} - b_j$ correspondants à l'arc (i, j) ensuite on calcule les nouveaux multiplicateurs de Lagrange $u_{ij}^k = u_{ij}^{k-1} + p_k \times Sg_{ij}^k$. Cette heuristique est certainement basée sur le fait que lorsque k est grand, le vecteur C_k des coûts réduits sur les arcs du réseau ne change pas beaucoup d'une itération à une autre de l'algorithme de génération de colonnes. Ainsi pour k grand, on peut espérer voir u_{ij}^k converger vers une valeur optimale.

4. Algorithme (Méthode proposée « ALG.P.arc.CC »)

Les étapes principales de notre approche sont résumées ci-dessous :

Etap0 initialisation PMR^0

Etap1 (Résoudre PMR^k) par la méthode de simplex → $(z_{PM}^k, x^k, \delta^k)$

Etap2 (Résoudre SP^k)

- mètre à jour les coûts : $c_{ij} = c_{ij} - \delta_j$.
- calculer les multiplicateurs de Lagrange u_{ij}^k .
- calculer la solution $maxL(u_{ij}^k)$, en utilise APD-L
- calculer les solutions réalisables $\Phi(u_{ij}^k)$, en utilise APD-LND
- tester, **Si** min $\Phi(u_{ij}^k) \geq 0$ **alors** stop, aller à l'étape 5. **Sinon** aller à l'étape 3.

Etap3 généré la solution de meilleur coût négative et un sous-ensemble des solutions complémentaires qui peut être calculé par deux techniques (sélection, ou, résolution) → X^k.

Etap4 posé $PMR^k = PMR^k \cup \{X^k\}$.
Retournée à l'étape 1.

Etap5 → (z_{PM}^k, x^k) solution optimale.

---

5. Expérimentations

Cette section présente l'évaluation préliminaire de « ALG.P.arc.CC » . Jusqu'à maintenant, notre méthode n'a été testée que sur des problèmes de petite taille [voir l'annexe 2].

Les résultats de plusieurs tests sont présentés dans le tableau 1. Chaque ligne donne des informations sur la valeur optimale (la meilleure valeur) du problème trouvé par « ALG.P.s.M »,« ALG.P.s.CC », « ALG.P.arc.M » (voir l'annexe1), et notre méthode « ALG.P.arc.CC », le nombre d'itérations pour la méthode de génération de colonne.

Tableau. 1. Quelques résultats

problèmes	ALG.P.s.M		ALG.P.s.CC		ALG.P.arc.M		ALG.P.arc.CC	
	V	Nb.IT	V	Nb.IT	V	Nb.IT	V	Nb.IT
P1(initialisation 1)	7	1	7	1	6*	1	6*	1
P1(initialisation 2)	7	2	7	1	6*	2	6*	1
P2(initialisation 1)	15.5	10	15.5	6	15.5*	10	15.5*	4
P2(initialisation 2)	17	5	17	4	15.5*	4	15.5*	3
P3(initialisation 1)	23	15	23	7	23*	14	23*	4
P3(initialisation 2)	24	10	24	6	23*	10	23*	3

ALG.P.s.M : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par sommet) avec meilleurs coûts réduits [4].
ALG.P.s.CC : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par sommet) avec colonnes complémentaires.
ALG.P.arc.M : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par arc) avec meilleurs coûts réduits.
ALG.P.arc.CC : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par arc) avec colonnes complémentaires.
V: valeur optimal (la meilleure valeur) du problème trouver par cette technique ; Nb.IT : nombre d'itérations, (*) : valeur optimal.

Dans ce tableau, on considère scellement deux critères, la meilleure valeur du problème trouvé par cette technique et le nombre d'itérations globale.

Remarque

La comparaison entre les quatre techniques ont permis de constater que « ALG.P.arc.CC » a fourni de bons résultats. Ceux-ci sont meilleurs quand certaines conditions sont réunies : l'initialisation de l'algorithme, le choix des multiplicateurs de Lagrange et le pas de déplacement.

6. Méthode de séparation

La méthode de génération de colonnes est utilisée pour résoudre le problème relaxé au nœud U. Elle hybride l'algorithme de simplexe (méthode existant dans la librairie ILOG) avec une méthode nommée Pricing. Si la solution obtenue est fractionnaire alors une méthode de séparation est appliquée au problème P^u. Elle consiste à subdiviser l'ensemble des solutions entières Su en deux sous ensembles disjoints, ceci a pour conséquence d'éliminer la réalisabilité de la solution fractionnaire pour les deux nouveaux problèmes qui sont des fils de P^u.

7. Conclusion et perspectives

Dans cet article consacré à la résolution du Problème de Construction des Rotations avec Contraintes de Ressources (PCR-CR), nous avons principalement développé les approches de génération de colonnes et de décomposition en problème maître et sous-problème. La

difficulté de la résolution du sous-problème étant directement liée au nombre de ressources, nous avons particulièrement étudié les techniques de réduction de l'espace des ressources, cette notion de réduction étant un élément-clé de l'efficacité de la résolution globale du problème. Les testes préliminaires effectués sur plusieurs instances ont montré l'efficacité de « ALG.P.arc.CC». En effet, si dans une perspective de planification stratégique le temps de calcul peut s'avérer moins critique que le coût global du programme de rotations, en revanche dans un contexte opérationnel le gain de temps sur la résolution du sous-problème devient un enjeu majeur. Nos perspectives de recherche sur ce problème sont nombreuses. Parmi elles, citant les problèmes d'évalues la méthode « ALG.P.arc.CC » sur des instances de grande taille et sur des problèmes réels, aussi les problèmes de reconstruction d'une solution robuste suite à la perturbation, par un quelconque aléa, du planning initialement construit. Ces problématiques de ré-optimisation suscitent un intérêt croissant chez les ingénieurs chargés de la planification dans les grandes entreprises de transport et ouvrent des voies de recherche particulièrement intéressantes et prometteuses.

Références

[1] DESAULNIERS G., DESROSIERS J., DUMAS Y., MARC S., RIOUX B., SOLOMON M., SOUMIS F., «Crew Pairing at Air France », European Journal of Operational Research, vol. 97, p. 245–259, 1997.

[2] LAVOIE S., MINOUX M., ODIER E., « A new approach for crew pairing problems by column generation with an application to air transportation », European Journal of Operational Research, vol. 35, p. 45–58, 1988.

[3] LOISEAU I., CESELLI A., N.MACULAN, SALANI M., « Génération de colonnes en programmation linéaire en nombres entiers », PASCHOS V. T., Ed., Optimisation combinatoire : Concepts fondamentaux, Hermès, Paris, 2005.

[4] NAGIH A., SOUMIS F., « Nodal aggregation of resource constraints in a shortest path problem », European Journal of Operational Research, 2005.

[5] CRAINIC T., ROUSSEAU J., « The column generation principle and the airline crew scheduling problem », INFOR, vol. 25, p. 136–151, 1987.

[6] N. Touati, L. Létocart, and A. Nagih. Solutions diversification in a column generation scheme. En soumission à Discrete Optimization, 2008.

[7] N. Touati, L. Létocart, and A. Nagih. Reoptimization in a column generation scheme. En soumission à Computers and Operations Research, 2008.

[8] B.T. Polyak (1967). A General Method of Solving Extremum Problems, Soviet Math. Doklady, 8(3), 593–597.

[9] M. Desrochers et F. Soumis (1988a), A Generalized Permanent Labelling Algorithm for the Shortest Path Problem with Time Windows, INFOR 26, 191–212.

[10] Vangelis Paschos (2005), Livre, optimisation combinatoire 3 : applications, Hermès Science. ch 10.

Annexe 1

1-APD-L : Algorithme de programmation dynamique lagrangienne.

La prolongation se fait comme en (APD), la dominance au nœud j peut être interprétée comme la détermination des Pareto optimaux du problème multicritère a $|Q_2|+1$ fonctions, ou $Q = Q_1 \cup Q_2$

$$\begin{cases} \min\limits_{i;(i,j)\in A} \left(C_i + c_{ij} + \sum\limits_{q_1 \in Q_1} \boldsymbol{u_{ij}^{q_1}} \left(\max\left\{a_j^{q_1}, \left(T_i^{q_1} + t_{ij}^{q_1}\right)\right\} - b_j^{q_1}\right) ; \max\left\{a_j^{q_2}, \left(T_i^{q_2} + t_{ij}^{q_2}\right)\right\} , q_1 \in Q_1, q_2 \in Q_2 \right) \\ T_i^{q_2} + t_{ij}^{q_2} \le b_j^{q_2} \quad , \quad q_2 \in Q_2 \end{cases}$$

2-APD-LND : Algorithme de programmation dynamique lagrangienne avec ressources non dominé. Cet algorithme est comme APD-L mais utilise la ré-optimisation pour construire des solutions réalisables.

3-ALG.P.s.M : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par sommet) avec meilleurs coûts réduits [4].

Problème maitre.
Sour problème
- calculer les multiplicateurs de Lagrange u_j^k (Projection par sommet).
- calculer la solution $maxL(u_j^k)$, en utilise APD − L
- calculer les solutions réalisables $\Phi(u_j^k)$, en utilise APD − LND

Généré la solution de meilleur coût négative.

4-ALG.P.s.CC : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par sommet) avec colonnes complémentaires.

Problème maitre.
Sour problème
- calculer les multiplicateurs de Lagrange u_j^k (Projection par sommet).
- calculer la solution $maxL(u_j^k)$, en utilise APD − L
- calculer les solutions réalisables $\Phi(u_j^k)$, en utilise APD − LND

Généré la solution de meilleur coût négative et un sous-ensemble des solutions complémentaires.

5-ALG.P.arc.M : algorithme Agrégation des ressources par relaxation lagrangienne (Projection par arc) avec meilleurs coûts réduits.

Problème maitre.
Sour problème
- calculer les multiplicateurs de Lagrange u_{ij}^k (Projection par arc).
- calculer la solution $maxL(u_{ij}^k)$, en utilise APD − L
- calculer les solutions réalisables $\Phi(u_{ij}^k)$, en utilise APD − LND

Généré la solution de meilleur coût négative.

Annexe 2

Problèmes	Nombre de vols	Arcs	Le nombre total de rotations
P1	4	15	16
P2	10	30	40
P3	20	70	102

Mathematical Integer Programming for a One Machine Scheduling Problem

Samia Ourari[1,2] and Cyril Briand[2], and Brahim Bouzouia[1]

[1]Centre de Développement des Technologies Avances, Alger, Algrie
[2]Laboratoire d'Architecture et d'Analyses des Systmes-CNRS; Toulouse, France
sourari@cdta.dz;briand@laas.fr;bbouzouia@cdta.dz

Abstract. This paper considers the problem of scheduling n jobs on a single machine. A fixed processing time and an execution interval are associated with each job. Preemption is not allowed. The objective is to find a feasible job sequence that minimizes the number of tardy jobs. On the basis of an original mathematical integer programming formulation, this paper shows how both good-quality lower and upper bounds can be computed. Numerical experiments on Baptiste et al.'s instances are provided, which demonstrate the efficiency of the approach.

1 Introduction

A single machine scheduling problem (SMSP) consists of a set V of n jobs to be sequenced on a single disjunctive resource. The interval $[r_j, d_j]$ defines the execution window of each job j, where r_j is the release date of j and d_j, its due-date. The processing time p_j of j is known and preemption is not allowed. A job sequence σ is said feasible if, for any job $j \in V$, $s_j \geq r_j$ and $s_j + p_j \leq d_j$, s_j being the earliest starting time of Job j in σ.

In this paper, we take an interest in finding a job sequence that minimizes the number of late jobs, problem referred to as $1|r_j|\sum U_j$ in the litterature, where U_j is set to 1 if job j is late, *i.e.*, $U_j \leftarrow (s_j + p_j > d_j)$. In the sequel, we review some important papers that deal with this problem. Nevertheless, the litterature is also rich of papers that consider other important variants by considering various additional assumptions, such as: jobs are weighted (problem $1|r_j|\sum w_j U_j$), setup times are considered, there exists several identical machines, *etc.* For a more extended review, the reader is referred to [5].

Determining whether it exists a feasible sequence, *i.e.*, all jobs meet their due dates, is NP-complete [14]. The problem of minimizing the number of late jobs is also NP-hard [10]. Efficient branch-and-bound procedures are reported in [2, 8, 5] that solve problem instances with up to 200 jobs.

When additional assumptions are made, $1|r_j|\sum U_j$ problems can become solvable in polynomial time. For instance, when release dates are equal, $1||\sum U_j$ problems can be solved in $O(n \log(n))$ using Moore's well known algorithm [15]. Considering the case where release and due dates of jobs are similarly ordered,

i.e., $r_i < r_j \Rightarrow d_i \leq d_j$), Kise, Ibaraki and Mine proposed a dynamic programming algorithm that runs in $O(n^2)$ [11]. Under this same assumption, an $O(n \log(n))$ algorithm was later proposed by Lawler in 1982 [12]. Lawler [13] also described an $O(n \log(n))$ algorithm that works on preemptive nested problems $1|r_j, nested, pmtn|\sum U_j$, *i.e.*, job preemption is allowed and job execution windows are nested: $r_i < r_j \Rightarrow d_i \geq d_j$ or $d_i > d_j \Rightarrow r_i \leq r_j$. More recently, considering the general preemptive problem $1|r_j, pmtn|\sum U_j$, Baptiste designed an algorithm that runs in $O(n^4)$ [1]. When processing times are equal, Carlier [6] proposed in the early eighties a $O(n^3 \log(n))$ procedure. Nevertheless, this procedure has recently been proved non-optimal by Chrobak et al. [7] who exhibit a new optimal $O(n^5)$ algorithm.

This paper presents how, using an original Mathematical Integer Programming (MIP) formulation, both good-quality lower and upper bounds can be computed for the $1|r_j|\sum U_j$ problem. The proposed approach mainly differs from the branch-and-bound approaches described in [2], [8] and [5] in the fact that, since MIP is used, it is more generic: new constraints can easily be added to the model. Moreover, from the best of our knowledge, there does not exist other MIP approach for the $1|r_j|\sum U_j$ problem that can be compared in terms of efficiency with the already existing branch-and-bound methods.

The paper is structured as follows. First, a dominance theorem is recalled that stands for the problem of finding a feasible sequence to the SMSP. In Section 3, a MIP formulation for searching a feasible job sequence is described. The fourth section discusses the validity of the dominance theorem of Section 2 when the $\sum U_j$ criterion is considered. Section 5 and 6 show how the MIP of Section 3 can easily be adapted for computing an upper bound and a lower bound to the $1|r_j|\sum U_j$ problem. The last section is devoted to the synthesis of the numerical experiments and discusses the efficiency of our MIP approach.

2 A general dominance theorem for the SMSP

In this section, some analytical dominance conditions are recalled for the SMSP. They have been originally proposed in the early eighties by Erschler et al. [9] within a theorem that is stated in the sequel. This theorem uses the notions of a *top* and a *pyramid* that are defined below. It defines a set S_{dom} of dominant job sequences, with respect to the feasibility problem, for the SMSP. Let us recall that a job sequence σ_1 dominates another job sequence σ_2 if σ_2 feasible $\Rightarrow$ σ_1 feasible. By extension, a set of job sequences S_{dom} is said dominant if, for any job sequence $\sigma_2 \notin S_{\text{dom}}$, it exists $\sigma_1 \in S_{\text{dom}}$ such that σ_2 feasible $\Rightarrow \sigma_1$ feasible.

Characterizing a set of dominant job sequences is of interest since, when searching for a feasible job sequence, only the set of dominant sequences need to be explored. Indeed, when there does not exist any feasible sequence in the dominant set, it can be asserted that the original problem does not admit any feasible solution. This allows a significant reduction of the search space.

Definition 1 *A job $t \in V$ is a top if there does not exist any other job $j \in V$ such that $r_j > r_t \wedge d_j < d_t$ (i.e., the execution window of a top does not strictly include any other execution window).*

The tops are indexed in ascending order with respect to their release dates or, in case of tie, in ascending order with respect to their due dates. When both their release dates and due dates are equal, they can be indexed in an arbitrary order. Thus, if t_a and t_a are two tops then $a < b$ if and only if $(r_{t_a} \leq r_{t_b}) \wedge (d_{t_a} \leq d_{t_b})$. Let m refers to as the total number of tops.

Definition 2 *Given a top t_k, a pyramid P_k related to t_k is the set of jobs $j \in V$ such that $r_j < r_{t_k} \wedge d_j > d_{t_k}$ (i.e., the set of jobs so that their execution window strictly includes the execution window of the top).*

Considering the previous definition, we stress that a non-top job can belong to several pyramids. Let $u(j)$ ($v(j)$ resp.) refers to as the index of the first pyramid (the last pyramid resp.) to which Job j can be assigned.

The following theorem can now be stated. The reader is referred to [9] for its proof.

Theorem 1 *The set S_{dom} of job sequences in the form:*

$$\alpha_1 \prec t_1 \prec \beta_1 \prec \cdots \prec \alpha_k \prec t_k \prec \beta_k \prec \cdots \prec \alpha_m \prec t_m \prec \beta_m$$

where:

- *t_k is the top of Pyramid P_k, $\forall k = 1 \ldots m$;*
- *α_k and β_k are two job subsequences located at the left and the right of Top t_k respectively, such that jobs belonging to subsequence α_k are sequenced with respect to the increasing order of their r_j, and jobs belonging to β_k, are sequenced with respect to the increasing order of their d_j ;*
- *any non-top j is located either in subsequence α_k or β_k, for a given k such that $u(j) \leq k \leq v(j)$.*

is dominant for the problem of finding a feasible job sequence.

3 A MIP formulation for finding a feasible job sequence

In this section, the problem of searching a feasible job sequence is considered and a MIP is described that has been originally introduced in [4]. It aims at determining the most dominant job sequence among the set S_{dom} in the form $\alpha_1 \prec t_1 \prec \beta_1 \prec \cdots \prec \alpha_m \prec t_m \prec \beta_m$. It is formulated below.

In the MIP, the binary variable x^+_{ki} (x^-_{ki} resp.) is set to 1 if the job i, assigned to Pyramid P_k, is sequenced in α_k (in β_k resp.), provided that (see constraint (3.7)):

$$\max \; z = \min_{k=1,\dots,m}(D_k - R_k - p_{t_k})$$

$$\text{s.t.} \quad \begin{cases} R_k \geq r_{t_k} & , \; \forall k \in [1\ m] & (3.1) \\ R_k \geq r_i + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x^+_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (3.2) \\ R_k \geq R_{k-1} + \sum_{\{j \in P_{k-1}\}} p_j x^-_{(k-1)j} + p_{t_{k-1}} & & \\ \qquad + \sum_{\{j \in P_k\}} p_j x^+_{kj} & , \; \forall k \in [2\ m] & (3.3) \\ D_k \leq d_{t_k} & , \; \forall k \in [1\ m] & (3.4) \\ D_k \leq d_i - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x^-_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (3.5) \\ D_k \leq D_{k+1} - \sum_{\{j \in P_{k+1}\}} p_j x^+_{(k+1)j} - p_{t_{k+1}} & & \\ \qquad - \sum_{\{j \in P_k\}} p_j x^-_{kj} & , \; \forall k \in [1\ (m-1)] & (3.6) \\ \sum_{k=u(i)}^{v(i)} (x^-_{ki} + x^+_{ki}) = 1 & , \; \forall i \in P_k & (3.7) \\ x^-_{ki} \ , \ x^+_{ki} \in \{0,1\} & , \; \forall k \in [1\ m], \forall i \in P_k & \\ D_k \ , \ R_k \in \mathbb{Z} & , \; \forall k \in [1\ m] & \end{cases}$$

- $u(i) \leq k \leq v(i)$;
- i cannot be sequenced both in α_k and β_k;
- i cannot be assigned to several pyramids.

The integer variable R_k corresponds to the earliest starting time of Job t_k. By definition:

$$R_k = \max(r_{t_k}, eft_{k-1} + \sum_{\{j \in \alpha_k\}} p_j, \max_{i \in \alpha_k}(r_i + p_i + \sum_{\{j \in \alpha_k | i \prec j\}} p_j)) \tag{3.8}$$

where eft_{k-1} is the earliest completion time of the job subsequence β_{k-1}. As the variable R_{k-1} corresponds to the earliest starting time of Job t_{k-1}, it comes that $eft_{k-1} = R_{k-1} + p_{t_{k-1}} + \sum_{j \in \beta_{k-1}} p_j$. Therefore, the constraints (3.1), (3.2) and (3.3), according to Equation (3.8), allow to determine the value of R_k.

Symmetrically, the integer variable D_k corresponds to the latest finishing time of Job t_k. By definition:

$$D_k = \min(d_{t_k}, lst_{k+1} - \sum_{\{j \in \beta_k\}} p_j, \min_{i \in \beta_k}(d_i - p_i - \sum_{\{j \in \beta_k | j \prec i\}} p_j)) \tag{3.9}$$

where lst_{k+1} is the latest starting time of the job subsequence α_{k+1}. As the variable D_{k+1} corresponds to the latest finishing time of Job t_{k+1}, it comes that $lst_{k+1} = D_{k+1} - p_{t_{k+1}} - \sum_{j \in \alpha_{k+1}} p_j$. Therefore, the constraints (3.4), (3.5) and (3.6), according to Equation (3.9), give to D_k its value.

Obviously, it can be observed that the values of the R_k and D_k variables of the MIP can directly be deduced from the values of the x^+_{ki} and x^-_{ki} binary variables. In [4], it is shown that if $z = \min_{k=1,\dots,m}(D_k - R_k - p_{t_k})$ is maximized

while respecting the constraints, then the obtained sequence dominates all the others. Indeed, the authors give the proof that, for any feasible combination of the x^+_{ki} and x^-_{ki} variables respecting the MIP constraints, a job sequence is obtained having its maximum lateness $L_{\max}$ strictly equals to $-z$. Therefore, maximizing z is strictly equivalent to minimize the maximum lateness $L_{\max}$ and it can be asserted that any sequence $\alpha_1 \prec t_1 \prec \beta_1 \prec \cdots \prec \alpha_m \prec t_m \prec \beta_m$ is feasible if and only if $z = \min_{k=1,\ldots,m}(D_k - R_k - p_{t_k}) \geq 0$. In the case where $z^* < 0$, there obviously does not exist any feasible sequence of n jobs for the considered problem.

4 Dominance condition for the $1|r_j|\sum U_j$ problem

In this section, the $\sum U_j$ criterion is considered. Searching optimal solution for $1|r_j|\sum U_j$ problem amounts to determine a feasible sequence for the largest selection of jobs $E \subseteq V$. Let E^* be this selection. The jobs of E^* are on time while others are late. The late jobs can be scheduled after the jobs of E^* in any order. So they do not need to be considered when searching a feasible job sequence for on-time jobs. Consequently, Theorem 1 can be applied only to the jobs belonging to E^*. There are $\sum_{k=1\ldots n} C^k_n$ possible different selections of jobs. Regarding the $\sum U_j$ criterion, the following corollary is proved.

Corollary 1 *The union of all the dominant sequences that Theorem 1 characterizes for each possible selection of jobs is dominant for the $\sum U_j$ criterion.*

Proof. The proof is obvious since the union of all the sequences that Theorem 1 characterizes for any possible selection necessarily includes the dominant sequences associated with E^*, hence an optimal solution. □

As already pointed out, the number of possible job selections is quite large. Nevertheless, as explained in [16], it is not necessary to enumerate all the possible job selections to get the dominant sequences. Indeed, they can be characterized using one or more *master-pyramid sequences*. The notion of a master-pyramid sequence is somewhat close to the notion of a master sequence that Dauzères-Pérès and Sevaux proposed in [8]. It allows to easily verify if a job sequence belongs to the set of dominant sequences that Theorem 1 characterizes. For building up a master-pyramid-sequence associated with a job selection $E \subseteq V$, the m_E tops and pyramids have first to be determined. Then, knowing that the set of dominant sequences is in the form $\alpha_1(E) \prec t_1(E) \prec \beta_1(E) \prec \cdots \prec \alpha_k(E) \prec t_k(E) \prec \beta_k(E) \prec \cdots \prec \alpha_{m_E}(E) \prec t_{m_E}(E) \prec \beta_{m_E}(E)$, it is assumed that any non-top job j is sequenced both in $\alpha_k(E)$ and $\beta_k(E)$ (these subsequences being ordered as described in Theorem 1), $\forall k$ such that $u(j) \leq k \leq v(j)$. For illustration, let us consider a problem instance with 7 jobs such that the relative order among the release and due dates of the jobs is $r_6 < r_1 < r_3 < r_2 < r_4 < d_2 < d_3 < d_4 < (r_5 = r_7) < d_6 < d_5 < d_1 < d_7$. For this example, the-master-pyramid-sequence associated with the selection $E = V$ is (tops are in bold):

$$\sigma_\Delta(V) = (6, 1, 3, \mathbf{2}, 3, 6, 1, 6, 1, \mathbf{4}, 6, 1, 1, \mathbf{5}, 1, \mathbf{7})$$

Any job sequence of n jobs *compatible* with $\sigma_\Delta(V)$ belongs to the set of dominant sequences. A sequence s is said compatible with the master-pyramid sequence $\sigma_\Delta(V)$ if the order of the jobs in s does not contradict the possible orders defined by $\sigma_\Delta(V)$, this will be denoted as $s \Subset \sigma_\Delta(V)$. Under the hypothesis that all tops are on-time, it is obvious that $\sigma_\Delta(V)$ also characterizes the set of dominant sequences (according to the $\sum U_j$ criterion) of any job selection E such that $\{t_1, \ldots, t_m\} \subseteq E$. Indeed, the master-pyramid sequence $\sigma_\Delta(E)$ associated with such a selection is necessarily compatible with the master-pyramid sequence $\sigma_\Delta(V)$, *i.e.*, if s is a job sequence such that $s \Subset \sigma_\Delta(E)$ then $s \Subset \sigma_\Delta(V)$.

Nevertheless, $\sigma_\Delta(V)$ does not necessarily characterize all the job sequences being dominant for the $\sum U_j$ criterion. This assertion can easily be illustrated considering a problem V with 4 jobs having the total order: $r_d < r_b < r_c < r_a < d_a < d_b < d_c < d_d$. Job a is the top of the corresponding structure and the master-pyramid sequence $\sigma_\Delta(V)$ is $(d, b, c, \mathbf{a}, b, c, d)$. Now, let us imagine that a is not selected (it is late and its interval can be ignored). In this case, there are two tops b and c and the master-pyramid sequence $\sigma_\Delta(V \setminus \{a\})$ is $(d, \mathbf{b}, d, \mathbf{c}, d)$. As it can be observed, $\sigma_\Delta(V \setminus \{a\})$ is not compatible with $\sigma_\Delta(V)$ since, in the former, Job d cannot be sequenced between b and c, while it is possible in the latter. This simple example shows that the complete characterization of the set of dominant sequences requires to enumerate all the non-compatible master-pyramid sequences, their number being possibly exponential in the worst case.

From now the focus is on a particular SMSP where any pyramid P_k, $\forall k = 1, \ldots, m$ is said *perfect*, *i.e.*, $\forall (i, j) \in P_k \times P_k$, $(r_i \geq r_j) \Leftrightarrow (d_i \leq d_j)$, *i.e.*, the execution intervals of the jobs belonging to P_k are included each inside the other. By extension, when all the pyramids are perfect, the corresponding SMSP will be said perfect. For this special case, the following theorem is proved:

Theorem 2 *Given a perfect SMSP V, the master-pyramid sequence $\sigma_\Delta(V)$ characterizes the complete set of sequences being dominant for the $\sum U_j$ criterion.*

Proof. Obviously, removing a job j from a perfect SMSP V produces a new perfect SMSP $V \setminus \{j\}$. The proof goes by showing that the master pyramid sequence $\sigma_\Delta(V \setminus \{j\})$ is compatible with $\sigma_\Delta(V)$ (in other words, all the sequences that are compatible with $\sigma_\Delta(V \setminus \{j\})$ are also compatible with $\sigma_\Delta(V)$). Let us assume first that the removed job j is a non-top job. Since $\sigma_\Delta(V)$ is built up according to the position of the tops, removing j from V induces to remove j from all the subsequences α_k, β_k of $\sigma_\Delta(V)$ such that $u(j) \leq k \leq v(j)$ and, in this case, the relation $\sigma_\Delta(V \setminus \{j\}) \Subset \sigma_\Delta(V)$ necessarily holds.
Let us assume now that j is a top having the index x (*i.e.*, $\sigma_\Delta(V) = (\alpha_1, t_1, \ldots, t_{x-1}, \beta_{x-1}, \alpha_x, j, \beta_x, \alpha_{x+1}, t_{x+1}, \ldots, t_m, \beta_m)$). Two cases have to be considered: if j is a top such that $\forall i \in P_x \Rightarrow i \in P_{x-1}$ or $i \in P_{x+1}$, then $\sigma_\Delta(V \setminus \{j\}) = (\alpha_1, t_1, \ldots, t_{x-1}, \beta_{x-1}, \alpha_{x+1}, t_{x+1}, \ldots, t_m, \beta_m)$ and in this case, $\sigma_\Delta(V \setminus \{j\})$ is obviously compatible with $\sigma_\Delta(V)$. Otherwise, let k be the last job of α_x (*i.e.*, $\alpha_x = (\alpha'_x, k)$). Since the execution intervals of the jobs belonging to P_x are included each inside the other, the order of the jobs in α_x matches the reverse

order of the jobs of β_x, therefore k is also the first job of β_x (*i.e.*, $\beta_x = (k, \beta'_x)$). Then $\sigma_\Delta(V \setminus \{j\}) = (\alpha_1, t_1 \dots t_{x-1}, \beta_{x-1}, \alpha'_x, k, \beta'_x, \alpha_{x+1}, t_{x+1} \dots t_m, \beta_m)$ and in this case, $\sigma_\Delta(V \setminus \{j\})$ is obviously compatible with $\sigma_\Delta(V)$. □

5 Computing an upper bound for the $1|r_j|\sum U_j$ problem

In this section, we take an interest in finding a feasible job sequence in the form $\alpha_1 \prec t_1 \prec \beta_1 \cdots \prec \alpha_m \prec t_m \prec \beta_m$ that minimizes the $\sum U_j$ criterion, knowing that a non-top j is not necessarily sequenced, that is the major difference with Section 3. As discussed in Section 4, the sequences in this form are dominant for the $\sum U_j$ criterion only under the hypothesis that the top jobs are scheduled on time. As it does not necessarily exist any optimal sequence satisfying this assumption, finding an optimal sequence in this desired form only gives an upper bound to the general $1|r_j|\sum U_j$ problem. Nevertheless, as shown by the experiments (see Section 7), this upper bound is good and even often optimal.

The MIP formulation of Section 3 can easily be adapted for solving the previous problem:

$$\min \; z = \sum_{\{j \in V \setminus \{t_1 \dots t_m\}\}} (1 - \sum_{k=u(j)}^{v(j)} (x^-_{jk} + x^+_{jk})) + \sum_{k=1}^{m} y_{t_k}$$

$$\text{s.t.} \begin{cases}
R_k \geq r_{t_k} & , \; \forall k \in [1\ m] & (5.1)\\
R_k \geq r_i + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x^+_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (5.2)\\
R_k \geq R_{k-1} + \sum_{\{j \in P_{k-1}\}} p_j x^-_{(k-1)j} + p_{t_{k-1}} & & \\
\qquad + \sum_{\{j \in P_k\}} p_j x^+_{kj} & , \; \forall k \in [2\ m] & (5.3)\\
D_k \leq d_{t_k} & , \; \forall k \in [1\ m] & (5.4)\\
D_k \leq d_i - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x^-_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (5.5)\\
D_k \leq D_{k+1} - \sum_{\{j \in P_{k+1}\}} p_j x^+_{(k+1)j} - p_{t_{k+1}} & & \\
\qquad - \sum_{\{j \in P_k\}} p_j x^-_{kj} & , \; \forall k \in [1\ (m-1)] & (5.6)\\
\sum_{k=u(i)}^{v(i)} (x^-_{ki} + x^+_{ki}) \leq 1 & , \; \forall i \in P_k & (5.7)\\
D_k - R_k \geq p_{t_k}(1 - y_{t_k}) & , \; \forall k \in [1\ m] & (5.8)\\
y_{t_k} \;,\; x^-_{ki} \;,\; x^+_{ki} \in \{0,1\} & , \; \forall k \in [1\ m], \forall i \in P_k & \\
D_k \;,\; R_k \in \mathbb{Z} & , \; \forall k \in [1\ m] &
\end{cases}$$

Let us comment this MIP. First, constraints (5.1)-(5.6) are identical to constraints (3.1)-(3.6) since integer variables R_k and D_k are determined in the same way. Allowing a non-top job to be late is easy by relaxing constraint (3.7), replacing it by constraint (5.7). As the feasibility of the obtained sequence is required, the constraint $D_k - R_k \geq p_{t_k}$ is set, $\forall k = 1, \dots, m$. Nevertheless, we observe that this constraint is a bit too strong since, when two consecutive tops t_k and t_{k+1} are such that $d_{t_{k+1}} - r_{t_k} < p_{t_{k+1}} + p_{t_k}$, the MIP is unfeasible (*i.e.*, there does not exist any feasible sequence that keeps both t_k and t_{k+1} on time). For avoiding this unfeasibility, the binary variable y_{t_k} is introduced (see constraint

(5.8)): y_{t_k} equals 1 if the processing time of t_k is ignored, 0 otherwise. Therefore, the $\sum U_j$ criterion can easily be expressed using the binary variables y_{t_k}, x^+_{ki} and x^-_{ki} since, if $y_{t_k} = 1$, Top t_k is late and, if $\sum_{k=u(j)}^{v(j)}(x^-_{jk} + x^+_{jk}) = 0$, non-top job j is late.

6 Lower-bound for the $1|r_j|\sum U_j$ problem

In a minimization problem, a lower bound is obtained by relaxing some constraints and optimally solving the relaxed problem. According to Theorem 2, when all the pyramids are perfect, the set of sequences in the form $\alpha_1 \prec t_1 \prec \beta_1 \cdots \prec \alpha_m \prec t_m \prec \beta_m$ is dominant for the $\sum U_j$ criterion. Moreover, for any problem, we know that it is always possible to decrease the r_j values (or increase the d_j values) of some jobs in order to make the pyramids perfect, *i.e.*, such that $\forall (i,j) \in P_k \times P_k$, $(r_i \geq r_j) \Leftrightarrow (d_i \leq d_j)$, $\forall k = 1, \ldots, m$ (see Figure 1). Doing so, a relaxed problem is obtained that can be optimally solved by the following MIP:

$$\min \; z = \sum_{\{j \in V \setminus \{t_1 \ldots t_m\}\}} (1 - \sum_{k=u(j)}^{v(j)} (x^-_{jk} + x^+_{jk})) + \sum_{k=1}^{m} y_{t_k}$$

$$\text{s.t.} \quad \begin{cases}
R_k \geq r_{t_k} + y_{t_k}(r_{n_k} - r_{t_k}) & , \; \forall k \in [1\ m] & (6.1) \\
R_k \geq r_i + (1 - x^+_{ik})(r_{n_k} - r_i) & & \\
\qquad + \sum_{\{j \in P_k | r_j \geq r_i\}} p_j x^+_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (6.2) \\
R_k \geq R_{k-1} + \sum_{\{j \in P_{k-1}\}} p_j x^-_{(k-1)j} + p_{t_{k-1}} & & \\
\qquad + \sum_{\{j \in P_k\}} p_j x^+_{kj} & , \; \forall k \in [2\ m] & (6.3) \\
D_k \leq d_{t_k} + y_{t_k}(d_{n_k} - d_{t_k}) & , \; \forall k \in [1\ m] & (6.4) \\
D_k \leq d_i + (1 - x^-_{ik})(d_{n_k} - d_i) & & \\
\qquad - \sum_{\{j \in P_k | d_j \leq d_i\}} p_j x^-_{kj} & , \; \forall k \in [1\ m], \forall i \in P_k & (6.5) \\
D_k \leq D_{k+1} - \sum_{\{j \in P_{k+1}\}} p_j x^+_{(k+1)j} - p_{t_{k+1}} & & \\
\qquad - \sum_{\{j \in P_k\}} p_j x^-_{kj} & , \; \forall k \in [1\ (m-1)] & (6.6) \\
\sum_{k=u(i)}^{v(i)} (x^-_{ki} + x^+_{ki}) \leq 1 & , \; \forall i \in P_k & (6.7) \\
D_k - R_k \geq p_{t_k}(1 - y_{t_k}) & , \; \forall k \in [1\ m] & (6.8) \\
y_{t_k} \;,\; x^-_{ki} \;,\; x^+_{ki} \in \{0,1\} & , \; \forall k \in [1\ m], \forall i \in P_k & \\
D_k \;,\; R_k \in \mathbb{Z} & , \; \forall k \in [1\ m] &
\end{cases}$$

with:

- $r_{n_k} = \min_{\{j \in P_k\}} r_j$;
- $d_{n_k} = \max_{\{j \in P_k\}} d_j$;

This MIP differs from the one of Section 5 only by the addition of the terms in bold that allow to deactivate the constraints of type (6.1), (6.2), (6.4) or

(6.5) when some jobs are late. For instance, if t_k is late, *i.e.*, $y_{t_k} = 1$, the term $y_{t_k}(r_{n_k} - r_{t_k})$ ($y_{t_k}(d_{n_k} - d_{t_k})$ resp.) deactivates the constraint (6.1) (the constraint (6.4) resp.). Indeed, we know that the inequality $R_k \geq r_{n_k}$ (resp. $D_k \leq d_{n_k}$) is obviously always verified. Similarly, in the case where $i \notin \alpha_k$ ($i \notin \beta_k$ resp.), the term $(1 - x_{ik}^{+})(r_{n_k} - r_i)$ ($(1 - x_{ik}^{-})(d_{n_k} - d_i)$ resp.) deactivates the constraint (6.2) (the constraint (6.5) resp.). Note that the deactivation of constraints allows to ensure that only the constraints that concern the on-time jobs are taken into account.

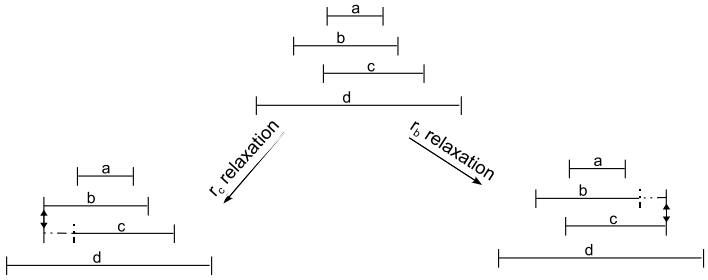


Fig. 1. Two relaxation strategies for turning a problem into a perfect one

7 Numerical experiments

For evaluating the performances of our MIP models, Baptiste et al.'s problem instances have been used (see [2]). For $n \in \{80, 100, 120, 140\}$, 120 problem instances are provided and, for each of them, thanks to the authors who provide us with their detailed results, either the optimal $\sum U_j$ value (OPT) or, at least, an upper-bound of this value ($BEST$), is known. For each problem instance, using a commercial MIP solver, we determined:

- two lower bounds by solving the MIP corresponding to the perfect SMSP obtained either by relaxation of the r_i (LBr_i) or by relaxation of the d_i (LBd_i), as explained in Section 6 ;
- one upper bound (UB) by solving the MIP described in Section 5.

In each experiment, the CPU time has been bounded to one hour. Table 1 displays, for the three kinds of MIP, the percentages of instances that were optimally solved within one hour, as well as the min / mean / max CPU time. For instance, when $n = 80$, the solver returns the optimal LBr_i value in 94.16% of the cases, with a min / mean / max CPU time of 0.01/42.35/1395.4 seconds respectively. The values of Table 1 can be compared with the percentage of time the Baptiste et al.'s method finds an optimal solution in less than one hour (see Table 3).

Instances	LB_r N/$Tcpu$	LB_d N/$Tcpu$	UB N/$Tcpu$
n=80	94.16 % (0.01; 42.35; 1395.54)s	96.66 % (0.02;61.15; 2363.76)s	98.33 % (0.02; 27.03; 1757.48)s
n=100	82.50 % (0.02; 141.15; 3531.11)s	81.66 % (0.03; 85.70; 1318,86)s	94.13 % (0.02; 33.84; 1778.42)s
n=120	80.83 % (0.02; 106.78; 1340.29)s	84.16 % (0.04; 108.43; 2149.83)s	85 % (0.02; 127.67; 2600.95)s
n=140	65.83 % (0.05; 139.77; 1490.64)s	65.00 % (0.03; 173.97; 3072.82)s	73.33 % (0.02; 134.62; 2600.95)s

Table 1. Percentage of MIP solved to optimality

A few observations can be made at this point. First, even if some problem instances seem very hard to solve, optimal solutions are found in most of the cases. The computation of the upper bound is globally less time expensive than the one of the lower bound. This is not surprising since in the former case, because tops are assumed to be on time, the search space is less extended than in the latter case, where any job can be late or on time. We also observe in our experiments that the lower bound given by the relaxation of the d_i is often better than the one obtained by relaxation of the r_i. Since the two kinds of relaxation are symetric, this is possibly due to the way problem instances have been generated.

	$UB = LB$ with $LB = LB_r = LB_d$		$\delta = UB - LB$
Instances	All instances	Instances: $Tcpu < 1h$	$(min, mean, max)$
n=80	70.83 %	73.21%	(0; 0.38; 6)
n=100	70%	73.62 %	(0; 0.39; 2)
n= 120	56.66%	60.68 %	(0; 0.5; 2)
n=140	60.83 %	62.31 %	(0; 1.45; 2)

Table 2. Percentages of optimal solutions

Table 2 takes an interest in the cases where the solution is optimal, *i.e.*, the upper bound equals one of the lower bounds ($UB = LB = \max(LBr_i, LBd_i)$. Two sub-cases are distinguished according if one considers the total set of instances or only the instances for which the CPU time is lower than one hour. As one can see, optimality can be proved in many cases, even if the Baptiste et al.'s ad-hoc approach remains better from this point of view. Let us point out that our MIP approach proves the optimality of 3 instances that were not optimally solved by Baptiste et al.. The table also indicates the min / mean / max difference between the upper bound and the best lower bound: it is always small even for the largest instances.

Lastly, Table 3 gives a more tightened analysis of the quality of our upper bound UB. It is compared with either the optimal value OPT found by the

Instances	Baptiste et al.		Dauzère-Pérès et al.		$UB = OPT$ $Tcpu < 1h$		$UB \leq BEST$
n=80	96.70 %	117.3 s	98.3 %	49.0 s	95.83 %	27.03 s	100 %
n=100	90.00 %	273.5 s	95.0 %	78.4 s	90.00 %	33.84 s	100 %
n=120	84.20 %	538.2 s	93.3 %	89.70 s	81.66 %	127.64 s	99.17%
n=140	72.50 %	1037.3 s	73.3 %	233 s	71.66 %	134.62 s	98.33 %

Table 3. Analysis of the upper bound quality

Baptiste et al.'s method (when it is computed in less than one hour) or with the best solution $BEST$ that was returned otherwise. We observe that, nearly for all the instances, our upper-bound equals the optimal or the best solution found by the Baptiste et al's method. Moreover, we also observe that its computation is less time expensive in any cases. These observations seems to indicate that, in order to increase the percentage of solutions that our approach is able to certify optimal, the way to relax the problem for finding lower bounds should be improved. Mixed relaxation schemes where r_i and d_i values would be both relaxed, for instance intending to minimize the sum of their variations, could be explored.

8 Conclusion

Designing MIP models for solving efficiently basic SMSPs is of interest since MIP approaches are often adaptable for dealing with new constraints or new objective. As a proof of this statement, this paper shows how an original MIP model, used for solving the $1|r_j|L_{\max}$ problem, can be adapted for dealing with the more complex $1|r_j|\sum U_j$ problem. Since the analytical dominance condition used for designing the MIP formulation of the former problem is valid for the $\sum U_j$ criterion only under some restrictions (tops are not late), only an upper bound can be computed. However, as shown by the experiments, this upper bound is optimal in most of the cases, or at least very close to the optimum. In the particular case where the considered SMSP is *perfect* (see Section 4), the paper gives a MIP model that allows to directly find the optimal $\sum U_j$ value. Since it is always possible to relax the release dates or the due dates of any SMSP in order to make it perfect, this MIP also allows to compute good lower bounds.

For the future works, we plan to investigate preprocessing methods by applying variable-fixing techniques from Integer Linear Programming. Such techniques, successfully used in several papers (see for instance [3]), allow to definitively fix the value of some binary variables, namely those of y_{t_k}, x^+_{ki} and x^-_{ki} in our MIP, intending to tighten the search space and speed up the solving phase. We guess that these techniques will improve our approach from a computational viewpoint such that it becomes more competitive in comparison with the best existing branch and bound approaches.

References

1. Baptiste P., "Polynomial time algorithms for minimizing the weighted number of late jobs on a single machine when processing times are equal", Journal of Scheduling, Vol 2, pp245-252 (1999).
2. Baptiste P., Peridy L and Pinson E., "A Branch and Bound to Mininimze the Number of Late Jobs on a Single Machine with Release Time Constraints", European Journal of Operational Research, 144 (1), pp1-11 (2003).
3. Baptiste P., Della Croce F., Grosso A. and T'kindt V. (2009), "Sequencing a single machine with due dates and deadlines: an ILP-based approach to solve very large instances", Journal Of Scheduling, ISSN 1099-1425 (Online).
4. Briand C., Ourari S. "Une formulation PLNE efficace pour $1|r_j|L_{\max}$", 10ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'09), Nancy (France), pp.104-105 (2009) (*in french*).
5. R. M'Hallah, R.L. Bulfin "Minimizing the weighted number of tardy jobs on a single machine with release dates", European Journal of Operational Research, 176, pp727-744 (2007).
6. Carlier J., "Problèmes d'ordonnancements à durées égales", QUESTIO, 5(4), 219-228 (1981) (*in french*).
7. Chrobak M., Dürr C., Jawor W., Kowalik L., Kurowski M., "A Note on Scheduling Equal-Length Jobs to Maximize Throughput", Journal of Scheduling, Vol. 9, No 1, pp71-73 (2006).
8. Dauzère-Pérès S., Sevaux M., "An exact method to minimize the number of tardy jobs in single machine scheduling", Journal of Scheduling, Vol. 7, No 6, pp405-420 (2004).
9. Erschler, J., Fontan, G., Merce, C., Roubellat, F.,"A New Dominance Concept in Scheduling n Jobs on a Single Machine with Ready Times and Due Dates", Operations Research, Vol. 31, pp114-127 (1983).
10. Michael R. Garey, David S. Johnson, "Computers and Intractability, A Guide to the Theory of NP-Completeness", W. H. Freeman and Company (1979).
11. Kise H., Toshihide I., Mine H., "A Solvable Case of the One-Machine Scheduling Problem with Ready and Due Times", Operations Research, 26(1), pp121-126 (1978).
12. Lawler E.L., "Scheduling a single machine to minimize the number of late jobs", Preprint. Computer Science Division, University of California, Berkeley (1982).
13. E.L. Lawler, "A dynamic programming algorithm for preemptive scheduling of a single machine to minimize the number of late jobs", Ann. Oper. Res., 26, pp125-133 (1990).
14. Lenstra J.K., Rinnooy Han A.H.G , Brucker P., "Complexity of machine scheduling problems", Annals of Discrete Mathematics, vol. 1, pp343-362 (1977).
15. Michael J. Moore, "An n job, one machine sequencing algorithm for minimizing the number of late jobs", Management Science, 15(1), pp102-109 (1968).
16. Ourari S., Briand C. "Conditions de dominance pour le problème une machine avec minimisation des travaux en retard" 9ème Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'08), Clermont-Ferrand (France), pp.351-352 (2008)(*in french*).