Polyèdres et reconfiguration dynamique d'autocommutateurs répartis

Renaud Sirdey[1 2] et Hervé Kerivin[3]

1. Service d'architecture BSC, Nortel GSM Access R&D
Parc d'activités de Magny-Châteaufort, 78928 Yvelines cedex 09
renauds@nortel.com

2. Heudiasyc, UMR CNRS 6599, Université de Technologie de Compiègne
Centre de recherches de Royallieu, BP 20529, 60205 Compiègne cedex

3. Limos, UMR CNRS 6158, Université Blaise Pascal/Clermont-Ferrand II
Complexe scientifique des Céseaux, 63177 Aubière cedex
kerivin@math.univ-bpclermont.fr

Mots-clefs : Ordonnancement à contraintes de ressources, systèmes répartis.

1 Introduction

Cet article traite de l'étude d'un problème d'ordonnancement fortement NP-difficile à contraintes de ressources, le *problème de la Programmation des Déplacements de Processus*, lié à l'opérabilité de certains systèmes répartis temps réel à haute disponibilité [5].

Grossièrement, il s'agit d'ordonner des déplacements de processus de traitement d'appels de telle manière que des contraintes de capacité sur les processeurs du système ne soient jamais violées, les situations de blocage étant résolues en arrêtant temporairement des processus donc en renonçant temporairement à rendre une partie du service.

Étant donnés un ensemble de processus et un ensemble de processeurs, un état du système consiste en une distribution d'un sous-ensemble des processus sur les processeurs de manière à ce que les ressources consommées sur ces derniers n'excèdent pas leur capacité. Un processus peut être déplacé d'un processeur vers un autre de deux manières : soit il est *migré*, auquel cas il consomme des ressources sur les processeurs source et cible du déplacement pour la durée de la migration, soit il est *interrompu*, c'est-à-dire arrêté puis ultérieurement redémarré sur le processeur cible du déplacement. Soit M l'ensemble des déplacements nécessaires afin de faire passer le système d'un état arbitraire, l'*état initial*, vers un autre, l'*état final*, notre problème consiste à trouver $I \subset M$ et σ un ordonnancement réalisable des déplacements de $M \setminus I$ tel que $\sum_{m \in I} c_m$ soit minimum, c_m étant le coût d'interrompre le processus associé à m et I l'ensemble des déplacements interrompus (les interruptions étant toutes réalisées au début).

2 Un programme linéaire en nombres entiers

Soit U l'ensemble des processeurs du système. On note K_u la capacité résiduelle du processeur $u \in U$ dans l'état initial. Aussi, $S(u)$ et $T(u)$ dénotent l'ensemble des déplacements qui ont le processeur $u \in U$ pour source et l'ensemble de ceux qui l'ont pour cible, respectivement. Étant donné un déplacement $m \in M$, on note w_m la consommation du processus déplacé et s_m et t_m les processeurs source et cible du déplacement, respectivement.

Pour chaque paire $m, m' \in M$ $(m \neq m')$, nous introduisons alors les variables binaires

$$\delta_{mm'} = \begin{cases} 1 & \text{si } m \text{ précède à } m', \\ 0 & \text{sinon.} \end{cases}$$

En plus de satisfaire les contraintes [4]

$$\begin{cases} \delta_{mm'} + \delta_{m'm} = 1 & \forall \{m, m'\} \subseteq M, \quad (1) \\ \delta_{mm'} + \delta_{m'm''} - \delta_{mm''} \leq 1 & m \neq m' \neq m'' \neq m \in M \text{ (transitivité)}, \quad (2) \end{cases}$$

un programme de déplacement admissible se doit de satisfaire, pour tout $m \in M$, les contraintes

$$w_m \leq K_{t_m} + \sum_{m' \in S(t_m)} w_{m'} \delta_{m'm} - \sum_{m' \in T(t_m) \setminus \{m\}} w_{m'} \delta_{m'm} \qquad \forall m \in M. \quad (3)$$

Ces dernières contraintes imposent qu'au moment de la réalisation du déplacement m, la capacité du processeur cible doit être suffisante pour accueillir le processus déplacé.

Bien sûr, il se peut que le système composé des inégalités (1), (2), (3) et $\delta_{mm'} \in \{0, 1\}$ $(m, m' \in M,\ m \neq m')$ n'ait pas de solution. En conséquence, nous introduisons, pour tout $m \in M$, les variables binaires

$$\delta_{mm} = \begin{cases} 1 & \text{si } m \text{ est interrompu}, \\ 0 & \text{sinon.} \end{cases}$$

Dans la mesure où les interruptions sont réalisées au début, il est requis que seul les déplacements non interrompus soient ordonnés. Pour ce faire, il convient de remplacer les contraintes (1) par les contraintes [6]

$$\begin{cases} \delta_{mm'} + \delta_{m'm} + \delta_{mm} + \delta_{m'm'} \geq 1 & \forall \{m, m'\} \subseteq M, \quad (4) \\ \delta_{mm'} + \delta_{m'm} + \delta_{mm} \leq 1 & m \neq m' \in M. \quad (5) \end{cases}$$

Enfin, en termes de capacité, les contraintes (3) deviennent, pour tout $m \in M$,

$$(1 - \delta_{mm}) w_m \leq K_{t_m} + \sum_{m' \in S(t_m)} w_{m'} (\delta_{m'm'} + \delta_{m'm}) - \sum_{m' \in T(t_m) \setminus \{m\}} w_{m'} \delta_{m'm}. \quad (6)$$

Le problème de la Programmation des Déplacements de Processus consiste alors à minimiser

$$\sum_{m \in M} c_m \delta_{mm}$$

sous les contraintes (4), (5), (2), (6) et $\delta_{mm'} \in \{0, 1\}$, pour tout $m, m' \in M$. Le polytope associé à ce programme s'appelle le *polytope des programmes de déplacements* et se note P^M_{PMP}.

3 Sur le polytope des programmes de déplacements

Notons tout d'abord que P^M_{PMP} est de pleine dimension sous des conditions peu restrictives : il est nécessaire et suffisant que tous les programmes qui interrompent tous les déplacements sauf deux soient admissibles. Nous faisons donc cette hypothèse dans la suite.

Par ailleurs, les inégalités (4) et (5) ainsi que les inégalités $\delta_{mm'} \geq 0$, pour tout $m, m' \in M$ avec $m \neq m'$, définissent toujours des facettes de P^M_{PMP}. Cela n'est par contre le cas ni pour les inégalités $\delta_{mm} \geq 0$, pour tout $m \in M$, et $\delta_{mm'} \leq 1$, pour tout $m, m' \in M$, ni pour les contraintes

de transitivité (2). Ces dernières peuvent néanmoins être remplacées par les *contraintes de transitivité étendues*,

$$\delta_{mm'} + \delta_{m'm''} - \delta_{mm''} + \delta_{m'm'} \leq 1 \qquad m \neq m' \neq m'' \neq m \in M, \tag{7}$$

qui, elles, définissent toujours des facettes de P^M_{PMP}.

Soit $m_0 \in M$ et soient $\emptyset \subset A \subseteq T(s_{m_0})$ et $B \subseteq S(s_{m_0}) \setminus \{m_0\}$ tels que

$$\sum_{m \in A} w_m > K_{s_{m_0}} + \sum_{m \in \bar{B}} w_m, \tag{8}$$

avec $\bar{B} = S(s_{m_0}) \setminus (B \cup \{m_0\})$. L'inégalité de s-recouvrement engendrée par m_0, A et B est alors définie par

$$\sum_{m \in A} \delta_{mm_0} + \sum_{m \in B} \delta_{m_0 m} \leq (|A| + |B| - 1)(1 - \delta_{m_0 m_0}). \tag{9}$$

De la même manière, soit $m_0 \in M$ et soient $A \subseteq T(t_{m_0}) \setminus \{m_0\}$ et $\emptyset \subset B \subseteq S(t_{m_0})$ tels que

$$w_{m_0} + \sum_{m \in A} w_m > K_{t_{m_0}} + \sum_{m \in \bar{B}} w_m, \tag{10}$$

avec $\bar{B} = S(t_{m_0}) \setminus B$. L'inégalité de t-recouvrement engendrée par m_0, A et B est alors définie par

$$\sum_{m \in A} \delta_{mm_0} + \sum_{m \in B} \delta_{m_0 m} \leq (|A| + |B| - 1)(1 - \delta_{m_0 m_0}). \tag{11}$$

Ces deux familles de contraintes sont assez naturelles : la condition (8) (respectivement (10)) exprime le fait que *tous* les déplacements de A (respectivement $A \cup \{m_0\}$) ne peuvent pas avoir été réalisé si *aucun* des déplacements de $B \cup \{m_0\}$ (respectivement B) ne l'a été ou, autrement dit, que réaliser *tous* les déplacements de $\bar{B}$ ne libère pas suffisamment de ressources pour réaliser *tous* les déplacement de A (respectivement $A \cup \{m_0\}$). Les inégalités (9) (respectivement (11)) permettent d'éviter cela, dès lors que m_0 n'est pas interrompu.

On peut alors montrer que les contraintes de s- et de t-recouvrement définissent des facettes de P^M_{PMP} sous des conditions raisonnablement peu restrictives et qu'elles sont séparables en temps pseudopolynomial.

Les preuves de ces résultats ainsi que d'autres classes de facettes de P^M_{PMP} sont données dans [6,3].

4 Résultats empiriques

Le passage à la pratique s'est fait par le biais d'un algorithme de recherche arborescente polyédrique (*branch-and-cut*) dont le principal ingrédient est une relaxation linéaire comprenant les contraintes (4) et (5), les contraintes de transitivité étendues (7) ainsi que les contraintes de s- et de t-recouvrement. Cette relaxation, dont on peut théoriquement venir à bout en temps pseudopolynomial, est résolue à chaque nœud de l'arbre de recherche à l'aide d'un algorithme de coupe.

Nous avons illustré la pertinence pratique de cet algorithme en l'utilisant pour attaquer des instances *difficiles*[1] [7]. En particulier, nous avons pu résoudre exactement des instances ayant

[1] À la fois en termes de taille mais aussi de capacité résiduelle des processeurs.

jusqu'à 119 déplacements (pour un système à 70 processeurs) en moins de quatre heures. En pratique cependant, si le temps de calcul est limité à quatre heures, l'algorithme n'a de bonnes chances de succès que jusqu'à de l'ordre de 80 déplacements. Lorsque la taille des instances augmente, l'algorithme a néanmoins été en mesure de fournir des solutions approchées de bonne qualité *i. e. prouvées* ne se situer qu'à moins de 5% (généralement moins) de l'optimum, à la quasi-totalité des instances (de taille allant jusqu'à 180 déplacements) sur lesquelles nous l'avons essayé, toujours sous la contrainte d'une limitation du temps de calcul à quatre heures.

5 Perspectives

Le champs d'application de ces travaux peut être élargi à des problèmes de reroutage dans certains réseaux de télécommunications. En effet, le problème de la Programmation des Déplacements de Processus se trouve être un cas particulier d'un problème de reconfiguration de chemins de routage dans les réseaux MPLS[2] (*Multi-Protocol Label Switching* [1]). Grossièrement, il s'agit d'ordonner des déplacements de chemins de routage de telle manière que des contraintes de capacité sur les liens du réseau sous-jacent ne soient jamais violées, les situations de blocage étant résolues en supprimant temporairement des chemins [2].

Il s'avère que les résultats et méthodes brièvement présentés dans cet article se généralisent assez naturellement à ce problème et qu'ils pourraient ainsi contribuer significativement à l'amélioration des méthodes de résolution disponibles [8].

Références

[1] D. O. Awduche (1999). MPLS and traffic engineering in IP networks. *IEEE Communication Magazine*, 37(12) :42-47.

[2] B. C. Józsa et M. Makai (1999). On the solution of reroute sequence planning problems in MPLS networks. *Comuter Networks*, 42 :199-210.

[3] H. Kerivin et R. Sirdey (2006). *Polyhedral combinatorics of a resource-constrained ordering problem part II : on the process move program polytope*. Rapport technique PE/BSC/INF/17913 V01 EN, Nortel GSM Access R&D (soumis).

[4] M. Queyranne et A. Schulz (1994). *Polyhedral approaches to machine scheduling*. Rapport technique 408/1994, Technische Universität Berlin.

[5] R. Sirdey, J. Carlier, H. Kerivin et D. Nace (2005). *On a resource-constrained scheduling problem with application to distributed systems reconfiguration*. Rapport technique PE/BSC/-INF/15593 V01 EN, Nortel GSM Access R&D (soumis).

[6] R. Sirdey et H. Kerivin (2006). *Polyhedral combinatorics of a resource-constrained ordering problem part I : on the partial linear ordering polytope*. Rapport technique PE/BSC/INF/17254 V01 EN, Nortel GSM Access R&D (soumis).

[7] R. Sirdey et H. Kerivin (2006). *A branch-and-cut algorithm for a resource-constrained scheduling problem*. Rapport technique PE/BSC/INF/17912 V01 EN, Nortel GSM Access R&D (soumis).

[8] R. Sirdey (2006). *A polyhedral approach to reroute sequence planning in MPLS networks*. Rapport technique PE/BSC/INF/xxxx V01 EN, Nortel GSM Access R&D.

[2]Les auteurs souhaitent remercier O. Klopfenstein qui leur a suggéré cette équivalence.