Contrainte de flot et RCPSP avec temps de transfert

T. Benoist[1] et M. Diamantini[2]

[1] Bouygues e-lab, 1 av. Eugène Freyssinet, 78061 St Quentin en Yvelines Cedex, France,
tbenoist@bouygues.com
[2] ENSTA - 32 Boulevard Victor 75739 Paris Cedex 15, France
Maurice.Diamantini@ensta.fr

Mots clés : RCPSP, programmation par contraintes, contrainte globale de flot, ordonnancement.

1 Présentation du problème

Le problème RCPSP avec temps de transfert (noté $PS|temp, s_{ij}|Cmax$ dans [2]) consiste à ordonnancer un projet composé de n activités J_i avec $i \in \{1..n\}$ utilisant m ressources renouvelables $k \in \{1..m\}$ différentes. Chaque ressource k existe en une quantité limitée R_k. Chaque activité J_i utilise simultanément une quantité $r_i^k \in \mathbb{N}$ de chacune des ressources k pendant une durée indivisible p_i. Deux activités virtuelles J_0 et J_{n+1} de durée nulle sont ajoutées de façon à modéliser le début et la fin du projet.

Les *variables de décision* sont la date de début S_i de chaque activité ainsi que les flux ϕ_{ij}^k d'unités de chaque ressource k entre chaque paire d'activité (J_i, J_j). *L'objectif* consiste alors à minimiser la date de démarrage de l'activité J_{n+1}, c'est à dire la durée totale du projet.

Certains couples d'activités (J_i, J_j) sont soumis à des *contraintes de précédence généralisées*[3] entre leur date de démarrage S exprimées par $S_j - S_i \geq \delta_{ij}$ avec $\delta_{ij} \in \mathbb{Z}$ et S_i est la date de démarrage de J_i. Les contraintes de flots stipulent que pour chaque ressource k, les ϕ_{ij}^k forment un flot. La contrainte de capacité s'exprime alors simplement en limitant à R_k le flot sortant de J_0.

Lorsqu'une unité de ressource k passe d'une activité J_i à une activité J_j, un *temps de transfert* v_{ij}^k (*sequence dependent changover time*) doit être pris en compte, modélisant par exemple le temps de formation de personnels changeant d'activité, le temps de reconfiguration d'une usine, ou simplement le temps de nettoyage plus ou moins important d'un poste de peinture selon la différence de couleur entre deux carrosseries successivement peintes. Dans le problème RCPSP classique (sans temps de transfert), l'activité J_j peut débuter dès que suffisamment de ressources requises sont libérées par des activités terminées, indépendamment de ces activités. L'introduction des temps de transfert v_{ij}^k oblige à s'intéresser également à l'activité d'origine J_i de chacune des ressources transférées vers J_j. Toutes les unités d'une même ressource k étant identiques, seule la quantité globale transférée ϕ_{ij}^k est à considérer. Ces temps de transfert interviennent donc comme contraintes de précédence *conditionnelles* de la forme : $\phi_{ij}^k > 0 \Rightarrow S_j - S_i \geq p_i + v_{ij}^k$.

Finalement, la demande r_i^k est modélisée par un flot requis ϕ_{ii}^k sur un arc interne associé à chaque activité J_i, et notre problème RCPSP consiste à minimiser la durée du projet (le mSpan) sous des contraintes de précédence généralisées, de précédence conditionnelles liées aux temps de transfert et de flots de ressoures intra- ou inter-activités.

2 Technique de résolution

Bien que reputé difficile, le problème classique RCPSP (sans temps de transfert) est relativement bien traité par les techniques de Programmation Par Contraintes [4,3]. Cependant la contrainte sur les temps de transfert rend le problème encore plus difficile par le couplage qu'elle induit entre le séquencement de la solution, et les flots ϕ_{ij}^k entre chaque paire d'activités. Or, le flot est bien pris en compte dans [1] qui exploite le calcul de coupes maximales dans les graphes de flots des ressources au sein d'un *Branch and Bound*.

Par ailleurs, les contraintes globales sont un moyen classique en PPC pour exploiter l'efficacité d'algorithmes dédiés à des problèmes particuliers. Elles permettent d'améliorer l'efficacité du filtrage du point de vue de la réduction des domaines de variables. Or il existe une contrainte globale

[3] Ces contraintes de précédence sont dites généralisées dans le sens où elles permettent de modéliser des fenêtres temporelles arbitraires du style « Le béton ne peut être vibré que dans un certain délai après avec été coulé » à l'aide de deux arcs en sens inverse pondérés par des δij et δji de signe contraire.

de flot [5,6,8], qui a la propriété de maintenir l'existence d'un support (un flot compatible avec les domaines courants des variables de flux). Ce flot support n'est pas nécessairement réalisable du point de vue des autres contraintes du problème (sinon le problème serait facile), mais peut efficacement servir de guide pour les heuristiques de branchements au cours de l'énumération [7]. En tant que problème contraint avec une structure de flot sous-jacent, le RCPSP avec temps de transfert nous semble un bon problème candidat pour tester cette approche mixte.

En nous appuyant sur les instances qui sont utilisées dans [1], nous étudierons donc l'apport de la contrainte globale de flot et les utilisations possibles de son support pour le problème RCPSP avec temps de transfert. L'énumération des solutions consistera essentiellement – en maintenant les contraintes consistantes, en particulier celles de flot – à observer le profil de ressources obtenu en plaçant les activités au plus tôt, et à résoudre le premier conflit constaté en « écartant » deux activités impliquées dans ce conflit de façon à tendre vers un profil réalisable. L'heuristique du choix du couple d'activités à écarter prendra en compte les dates de démarrage au plus tôt, au plus tard ainsi que le support de flot entre chaque paire d'activités.

Les premiers tests semblent encourageants et les résultats sur de petites ou moyennes instances de [1] sont confirmés, voire améliorés[4].

Références

1. K. Neuman and Ch. Schwindt and J. Zimmermann : *Project Scheduling with time windows and scarse ressources* (édition 2002). Springer pp 150–160
2. P. Brucker et all (1999) : *Resource-constrained projet scheduling : Notation, classification, models, and methods.* European Journal of Operational Research 112 pp 3–41
3. S. Demassey (2003) : *Méthodes hybrides de programmation par contraintes et de programmation linéaire pour le problème d'ordonnancement de projet à contraintes de ressources.* Université d'Avignon.
4. Ch. Artigues (2004) : *Optimisation et Robustesse en Ordonnancement sous Contraintes de Ressources*, Thèse d'Habilitation. Université d'Avignon
5. A. Bockmayr, N. Pisaruk et A. Aggoun (2001) : *Network flow problems in constraint programming.* Principles and Practice of Constraint Programming, CP'2001. LNCS 2239, 196–210. Springer-Verlag
6. Th. Benoist, É. Gaudin et B. Rottembourg (2002) : *Constraint Programming Contribution to Benders Decomposition : a Case Study.* CP'2002 Ithaca - USA.
7. T. Benoist et É. Bourreau (2003) : *Improving global constraints support by local search.* In CP'03 Workshop on Cooperative Solvers (COSOLV'03).
8. É. Gaudin, N. Jussien et G. Rochart (2004) : *Explained global constraints at work.* RR0403-info. École des Mines de Nantes
9. Choco : *environnement java de Programmation Par Contraintes* (http ://choco.sourceforge.net/)

[4] Les tailles d'instances retenues sont de 10, 20 et 50 activités.