Bornes rapides pour l'ordonnancement dans LocalSolver

Philippe Laborie

LocalSolver, 24 avenue Hoche, Paris, France
plaborie@localsolver.com

Mots-clés : *solveur, bornes inférieures, ordonnancement.*

1 Introduction

Étant donné un problème d'ordonnancement exprimé en LocalSolver, notre objectif est de calculer *rapidement*, en début de résolution, une borne inférieure décente sur le *makespan* en complément des bornes plus coûteuses calculées en cours de recherche. Par *rapidement*, nous entendons en un temps de calcul faible devant celui alloué à la résolution du problème, soit, typiquement en dessous de la seconde pour des problèmes de quelques milliers de tâches.

2 Contexte et notations

Pour simplifier les notations, nous supposons que le problème est transcrit dans la représentation canonique des problèmes d'ordonnancement proposée dans [3]. En particulier :

— Les activités sont représentées par des variables d'intervalles x_i pouvant être optionnelles.

— Pour une variable d'intervalle x_i, les variables $p(x_i)$, $s(x_i)$, $e(x_i)$, $l(x_i)$ représentent respectivement la présence, début, fin et longueur de l'intervalle.

— Les contraintes temporelles sont représentées par des contraintes de type : $e(x_i) \leq s(x_j)$.

— Les contraintes de présence sont représentées par des implications de type $p(x_i) \Rightarrow p(x_j)$.

— La contrainte $alternative(x, \{x_1, ..., x_n\})$ signifie que lorsque x est présent, un et un seul des x_i $(i \in [1, n])$ est présent et qu'il est alors égal à x.

— La contrainte $disjoint(\{x_1, ..., x_n\})$ signifie que les x_i présents sont disjoints deux à deux.

— La contrainte $cumul(\{x_1, ..., x_n\}\{y_1, ..., y_n\}) \leq C$, où les y_i sont des variables entières, signifie que le cumul des contributions des intervalles x_i avec la valeur y_i n'excède pas C.

Si z est une variable entière, nous notons respectivement z^{min} et z^{max} les bornes inférieure et supérieure du domaine de z après la propagation initiale du problème. Nous supposons par ailleurs une variable entière fictive h telle que $\forall i, e(x_i) \leq h$ (valeur de fin de l'ordonnancement).

3 Bornes inférieures

Les bornes inférieures calculées sont basées sur des relaxations énergétiques du problème et mettent en œuvre les composants ci-dessous.

Propagation de type Energy Precedence. La propagation d'*energy precedence* [2] pour une variable d'intervalle x_i participant à une contrainte de ressource (de type *disjoint* ou *cumul*) assure que pour tout sous-ensemble Ω de prédécesseurs de x_i dans le graphe de contraintes temporelles, la ressource contient assez d'énergie pour exécuter tous les intervalles de Ω entre le début au plus tôt de Ω et le début de x_i. Cette propagation permet d'augmenter les valeurs de $s^{min}(x_i)$ même lorsqu'il n'y a pas de contrainte sur la valeur de fin au plus tard de l'ordonnancement h. Elle peut être effectuée pour chaque variable d'intervalle et pour chaque sous-ensemble Ω avec une complexité au pire cas en $O(n(p + \log(n)))$ où n désigne le nombre de variables d'intervalle et p le nombre maximal de prédécesseurs d'une variable donnée dans le graphe de précédences ($p < n$). Cette propagation est exécutée à la racine de la résolution.

Relaxation linéaire énergétique. Cette relaxation consiste en un programme linéaire dont les seules variables de décision sont les variables Booléennes de présence $p(x_i)$. Les contraintes de présence et d'*alternative* sont directement traduites tandis qu'une relaxation énergétique est introduite pour les ressources. Par exemple pour une contrainte de cumul nous aurons :

$$C \cdot \min_{i \in [1,n]} s^{min}(x_i) + \sum_{i \in [1,n]} p(x_i) \cdot l^{min}(x_i) \cdot y_i^{min} \leq C \cdot h$$

Cette formulation de base est améliorée en considérant les durées de latence des intervalles (longueur minimale du plus long chemin après un intervalle donné dans le graphe de précédence) ainsi que l'énergie non-utilisable du fait des $s^{min}(x_i)$ en début d'ordonnancement.

L'objectif du programme linéaire est la minimisation de la date de fin h. En pratique, pour des raisons de performance, seule la relaxation linéaire LP est résolue.

Reformulation des cumuls quasi-disjonctifs. La propagation de type *energy precedence* ainsi que la relaxation énergétique, utilisent des reformulations des contraintes de cumul inspirées de [1]. Plus précisément, pour une contrainte $cumul(\{x_1, ..., x_n\}\{y_1, ..., y_n\}) \leq C$, lorsqu'il existe 3 intervalles i, j, k tels que $y_i^{min} + y_j^{min} + y_k^{min} > C$, une contrainte de cumul redondante de capacité 2 avec des contributions $y \in \{2, 1, 0\}$, optimale au niveau énergétique, est utilisée.

4 Résultats

Le tableau (1) résume la déviation des bornes inférieures par rapport aux meilleures solutions primales connues sur quelques benchmarks classiques d'ordonnancement. Les temps de calcul sont négligeables. Sur le RCPSP, parmi les 2520 instances testées, 121 bornes inférieures de la littérature sont améliorées en particulier sur les plus grosses instances à 300 tâches (RG300).

	Instances	Taille	Déviation
Jobshop	133	36-2000	6.1%
Flexible jobshop	330	50-500	5.9%
Flexible jobshop avec temps de transition	20	50-100	15.0%
Openshop	60	16-400	0.7%
RCPSP	2520	30-300	4.6%

TAB. 1 – Déviation moyenne des bornes inférieures aux meilleures solutions connues

5 Conclusions et perspectives

Les bornes inférieures décrites dans cet article sont en cours d'implémentation dans LocalSolver. Les extensions prévues visent à améliorer ces bornes en présence de temps de transition ainsi qu'à introduire d'autres techniques de bornes rapides.

Références

[1] P. Baptiste and N. Bonifas. Redundant cumulative constraints to compute preemptive bounds. *Discrete Applied Mathematics*, 234, p68-177, 2018.

[2] P. Laborie. Algorithms for propagating resource constraints in AI planning and scheduling : Existing approaches and new results. *Artificial Intelligence*, 143(2), p151-188, 2003.

[3] P. Laborie, J. Rogerie, P. Shaw and P. Vilím. Reasoning with Conditional Time-intervals, Part II : an Algebraical Model for Resources. *Proc. FLAIRS*, 2009.