Bornes rapides dans Hexaly Optimizer basées sur les problèmes d'ordonnancement à une machine

Léa Petit-Jean Genat, Philippe Laborie

Hexaly, 251 Boulevard Pereire, Paris, France
{lpetitjean, plaborie}@hexaly.com

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

1 Introduction

Étant donné un problème d'ordonnancement exprimé dans le formalisme d'Hexaly Optimizer (anciennement LocalSolver), notre objectif est de calculer *rapidement*, en début de résolution, une borne inférieure décente sur l'objectif 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 d'activités. Pour ce faire nous proposons d'exploiter des algorithmes classiques, de complexité au plus $O(n \log(n))$, sur des sous-problèmes à une machine.

2 Notations et algorithmes polynomiaux considérés

Nous notons $\mathcal{A}$ l'ensemble des variables d'intervalles (activités) du modèle écrit pour Hexaly Optimizer. Pour $x \in \mathcal{A}$, $C(x)$ désigne la fin de x et $d(x)$ sa longueur (durée) minimale. Soit $\mathcal{P}(x) \subset \mathcal{A}$ l'ensemble de intervalles finissant nécessairement avant $C(x)$ du fait des contraintes de précédences et $\mathcal{M} \subset 2^{\mathcal{A}}$ l'ensemble des contraintes disjonctives du problème (machines).

Les algorithmes considérés sont ceux de Moore-Hodgson [3] et Potts-Wassenhove [2] pour les problèmes $(1||U_i)$ et $(1||w_iU_i)$, celui de Smith [4] pour $(1||\sum w_iC_i)$, et de Jackson [5] pour $(1||L_{\max})$. Ces algorithmes ont une complexité en $O(n \log(n))$ et donnent une solution optimale pour leur problème, à l'exception de celui de Potts-Wassenhove qui est une relaxation.

Dans les sections suivantes, nous illustrons notre travail sur l'algorithme de Smith. Les autres algorithmes sont traités dans le même esprit.

3 Exemple de l'algorithme de Smith

Pour rappel, l'algorithme de Smith pour résoudre le problème $(1||\sum w_iC_i)$ consiste à ranger les tâches dans l'ordre des d_i/w_i croissants.

3.1 Détection des expressions traitables par l'algorithme

Toutes les expressions du modèle qui s'expriment comme une somme pondérée de valeurs de fin de variables d'intervalles sont détectées. Soit $r = \sum w_iC(x_i)$ une telle expression. Nous calculons une borne inférieure sur r comme décrit ci-dessous. Il est à noter que l'approche n'est pas limitée à des expressions r participant à la fonction objectif : toute expression de cette forme fait l'objet d'un calcul de borne inférieure qui est propagé dans le reste du problème.

3.2 Mise en oeuvre de l'algorithme

Etant donné une machine $m \in \mathcal{M}$, nous pouvons calculer une borne sur $r = \sum w_i C(x_i)$ grâce à l'algorithme de Smith en répartissant les poids w_i des intervalles x_i de r sur les intervalles de m. Par exemple considérons une expression du modèle $r = w_3C(x_3)+w_4C(x_4)+w_5C(x_5)$ et une machine $m = \{x_1, x_2, x_3\}$. Supposons par ailleurs que $\{x_1, x_2, x_3\} \subset \mathcal{P}(x_4)$ et $\{x_1, x_2\} \subset \mathcal{P}(x_5)$. Il est possible de choisir des poids à utiliser par l'algorithme de Smith sur la machine m qui donnent une borne valide sur r, par exemple en répartissant uniformément les w_i selon les précédences : $\{\frac{w_4}{3} + \frac{w_5}{2}, \frac{w_4}{3} + \frac{w_5}{2}, w_3 + \frac{w_4}{3}\}$. D'autres répartitions peuvent être utilisées afin de produire des bornes valides, la valeur maximum de ces bornes étant conservée.

Il est aussi possible de raisonner globalement en répartissant les poids sur l'ensemble des machines afin de pouvoir prendre la somme (plutôt que le maximum) des valeurs des coûts optimaux relâchés de chaque machine produits par l'algorithme de Smith. Ainsi, dans l'exemple ci-dessus, si nous avons une seconde machine $m' = \{x_1, x_4, x_5\}$ une répartition globale valide pourrait être $\{\frac{w_4}{6} + \frac{w_5}{4}, \frac{w_4}{6} + \frac{w_5}{4}, w_3 + \frac{w_4}{6}\}$ sur m et $\{0, \frac{w_4}{2}, \frac{w_5}{2}\}$ sur m'. Notre implémentation exécute plusieurs instances de l'algorithme de Smith sur les différentes machines avec différentes configurations des poids afin de conserver la meilleure borne inférieure.

4 Résultats

L'approche a été étendue aux autres algorithmes et type d'objectifs évoqués ci-dessus et évaluée sur des benchmarks classiques (jobshop, etc.), utilisés tels quels ou modifiés selon l'objectif. Dans la table ci-dessous, les colonnes *Écart* désignent l'écart moyen des bornes inférieures aux meilleures bornes supérieures, respectivement avant et après le travail décrit ici.

Objectif	Nombre d'instances	Taux moyen d'instances améliorées	Écart avant	Écart après	Différence
U_i	640	84%	66%	48%	-18%
w_iU_i	760	77%	63%	54%	-9%
w_iC_i	652	95%	55%	24%	-31%
$L_{\max}$	700	69%	-	-	-35%

TAB. 1 – Amélioration moyenne des bornes inférieures par type d'objectif.

5 Conclusions et perspectives

La détection de sous-problèmes à une machine sur lesquels appliquer des algorithmes polynomiaux permet d'améliorer les bornes dans Hexaly Optimizer 12.5. Il reste à étudier plus en détails les différentes façons de répartir les poids w_i (et, selon les objectifs, les dates d'échéance) des expressions r entre les intervalles des machines afin d'obtenir des bornes encore meilleures.

Références

[1] P. Laborie. Bornes rapides pour l'ordonnancement dans LocalSolver. *ROADEF 2023.*

[2] C. Potts et L. Van Wassenhove. Algorithms for scheduling a single machine to minimize the weighted number of late jobs. *Management Science 1988.*

[3] J. M. Moore. An n job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science 1968.*

[4] W. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly 1956.*

[5] J.R. Jackson. Scheduling a production line to minimize maximum tardiness. *Office of Technical Services 1955.*