Étude de l'intégration de techniques d'algorithmique exponentielle pour la résolution d'un problème d'ordonnancement par une méthode arborescente.

Olivier Ploton[1], Vincent T'kindt[1]

Université de Tours, Laboratoire d'Informatique Fondamentale et Appliquée (LIFAT, EA 6300), ERL CNRS 7002 ROOT, Tours, France

{olivier.ploton,vincent.tkindt}@univ-tours.fr

Mots-clés : *Ordonnancement, Branch-and-Bound, Relaxation*

1 Introduction

Considérons un problème où n travaux doivent être ordonnancés dans un environnement à une machine. Chaque travail i est défini par une durée opératoire p_i, une date de fin impérative ou deadline $\tilde{d}_i$, et un poids w_i. On cherche à minimiser la somme pondérée des dates de fin C_i. Ce problème, noté $1|\tilde{d}_i|\sum w_i C_i$ ([6]), est NP-difficile au sens fort.

Shang, T'kindt et Della Croce [8] ont proposé pour le résoudre une méthode arborescente, où chaque nœud est un sous-problème particulier : certains travaux étant déjà ordonnancés à la fin, on cherche à ordonnancer au début un ensemble $S = \{i_1, \dots, i_k\}$ de travaux pour constituer une permutation $(i_1, \dots, i_k)$, tout en améliorant la meilleure solution connue. Expérimentalement, cet algorithme donne de très bons résultats en moyenne. Néanmoins, son comportement au pire cas n'est pas bon, ce qui nous amène à essayer de l'améliorer grâce à des techniques inspirées de l'algorithmique exponentielle, comme l'Inclusion-Exclusion. Cette technique permet d'obtenir des algorithmes théoriques dont la complexité au pire cas peut être intéressante.

2 Dérivation de techniques d'élagage inspirées de l'Inclusion-Exclusion

La technique d'Inclusion-Exclusion a été introduite par Karp [7] et décrite dans [3]. Elle permet d'établir des algorithmes exponentiels exacts de complexité au pire cas réduite. Étant donnés un ensemble de travaux S et une borne UB, la formule d'Inclusion-Exclusion permet de calculer le nombre $N(S, UB)$ d'ordonnancements admissibles d'objectif strictement inférieur à UB, et donc elle permet de couper un nœud du Branch&Bound si ce nombre est nul.

Notons, pour $X \subset S$, X^* l'ensemble des listes d'éléments de X, $\mathfrak{S}_X$ l'ensemble des permutations sur X, et E l'espace des ordonnancements admissibles relâchés d'objectif strictement inférieur à UB. Alors on a :

$$\underbrace{\operatorname{card} E \cap \mathfrak{S}_S}_{N(S,UB)} = \sum_{X \subset S} (-1)^{|S|-|X|} \underbrace{\operatorname{card} E \cap X^*}_{N_X \text{ à calculer}} \tag{1}$$

Chaque N_X se calcule par programmation dynamique selon le schéma suivant : $N_X[C, \ell, \varepsilon]$ représente le nombre d'ordonnancements relâchés, avec ℓ travaux, tous dans X, de date de fin exactement C et d'objectif inférieur ou égal à ε. Donc $N_X = N_X[\sum_{i \in S} p_i,\ |S|,\ UB - 1]$, avec :

$$N_X[C,0,\varepsilon] = \; 1 \qquad \text{si } C=0 \text{ et } \varepsilon \geq 0, \quad 0 \text{ sinon} \tag{2}$$
$$N_X[C,\ell,\varepsilon] = \sum_{i \in X | \tilde{d}_i \geq C} N_X[C-p_i, \ell-1, \varepsilon - w_i C] \text{ si } C \geq 0 \text{ et } \varepsilon \geq 0, \quad 0 \text{ sinon} \qquad \forall \ell > 0 \tag{3}$$

Chaque calcul d'un N_X s'effectue en nombre d'états pseudopolynomial, majoré par $|S|(\sum p_i)^2 \sum w_i$. La somme d'Inclusion-Exclusion elle-même comporte $2^{|S|}$ termes. Il existe plusieurs techniques pour réduire ce nombre de termes, notamment les inégalités de Bonferroni [4], les Abstract Tubes [2] et la transformation de Möbius [5]. Ces techniques ont donné expérimentalement des résultats peu concluants. On s'intéresse donc à une relaxation de cette approche dans laquelle on ne s'intéresse plus au comptage d'ordonnancements mais plus simplement au test d'existence d'une solution de valeur strictement inférieure à *UB* à un nœud donné du Branch&Bound. Cela évide donc la génération des $2^{|S|}$ ensembles, et cela nous ramène à la résolution d'un problème d'ordonnancement relâché où les travaux peuvent être dupliqués comme l'on fait Abdul-Razaq et Potts [1].

Il s'agit donc dans un premier temps de calculer l'objectif optimum opt_R sur les ordonnancements relâchés, grâce à ce schéma de programmation dynamique : soit $opt_R[C]$ le meilleur objectif des ordonnancements de nombre quelconque de travaux et de date de fin exactement C. On a alors $opt_R = opt_R[P(S)]$, avec :

$$opt_R[\,0\,] = 0 \tag{4}$$
$$opt_R[C] = +\infty \qquad \forall C < 0 \tag{5}$$
$$opt_R[C] = \min_{i \in S | \tilde{d}_i \geq C} opt_R[C-p_i] + w_i C \qquad \forall C > 0 \tag{6}$$

Itérativement, il faut ensuite heuristiquement chercher à éliminer les travaux en exemplaires multiples : on définit des pénalités λ_i à ajouter à l'objectif pour les travaux i dont la duplication diminue artificiellement sa valeur, et l'on modifie les pénalités par itérations successives pour aboutir à une bonne borne de l'objectif ou même à une solution stricte optimale.

La manière de modifier les pénalités λ_i à chaque itération joue un rôle crucial. L'étude expérimentale est en cours et les résultats ne sont pas encore connus. Nous espérons identifier des situations où cette technique permet d'accélérer la résolution du problème $1|\tilde{d}_i|\sum w_i C_i$.

Références

[1] T. S. Abdul-Razaq, C. Potts. Dynamic Programming State-Space Relaxation for Single-Machine Scheduling. *Journal of the Operational Research Society*, (39) : 141–152, 1988.

[2] K. Dohmen. Improved Bonferroni Inequalities via Abstract Tubes : Inequalities and Identities of Inclusion-Exclusion Type. Springer, 2003.

[3] F.V. Fomin, D. Kratsch. Exact exponential algorithms. Springer, 2010.

[4] J. Galambos. Bonferroni Inequalities. *The Institute of Math. Stat.*, 5(4) : 577–581, 1977.

[5] X. Goaoc, J. Matoušek, P. Paták, Z. Safernová, M. Tancer. Simplifying inclusion-exclusion formulas. *European Conference on Combinatorics, Graph Theory and Applications*, 2013.

[6] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan. Optimization and Approximation in Deterministic Sequencing and Scheduling Problems : a Survey. *Proceedings of the ARI on Discrete Optimization and Systems Applications*, Elsevier, (5) : 287–326, 1979.

[7] R.M. Karp. Dynamic Processing meets the principle of inclusion and exclusion. *Operational Research Letters*, 1(2) : 49–51, 1982.

[8] L. Shang, V. T'Kindt, F. Della Croce. Branch & Memorize exact algorithms for sequencing problems : efficient embedding of memorization into search trees. *Computers & Operations Research*, (128) : 105–171, 2021.