Transformer un branch-and-bound en heuristique état de l'art : example sur le flowshop de permutation

Luc Libralesso[1,2], Pablo Andres Focke[1], Aurélien Secardin[1], Vincent Jost[1]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, G-SCOP, 38000 Grenoble, France
[2] Université Clermont Auvergne, LIMOS, CNRS UMR 6158, Aubière France
luc.libralesso@uca.fr

Mots-clés : *optimisation combinatoire, ordonnancement, permutation flowshop, iterative beam search*

Nous présentons une méthode simple (beam search itérative) qui intègre des composants de branch-and-bound et permet de trouver des résultats état-de-l'art sur le problème du flowshop de permutation (minimisation du makespan et minimisation flowtime). Plus d'informations dans [3].

Dans le flowshop de permutation, l'objectif est de trouver un ordonnancement des tâches, où chaque tâche suit la même route de machines. et minimise un certain objectif. Par exemple : temps de fin pour la dernière tâche (makespan) et somme des temps de fin des tâches (flowtime). Le flowshop de permutation est très fréquent (et fondamental) dans la littérature. Il consiste à ordonnancer les tâches dans le même ordre (une permutation des tâches est donc suffisante pour exprimer une solution).

Le flowshop de permutation étant un problème fondamental, un grand nombre de méthodes (exactes et heuristiques) ont été proposées pour le résoudre. Parmi les méthodes heuristiques, un consensus semble émerger, et considérer que des techniques comme les *Iterated Greedy Algorithms (IGA)* sont les plus efficaces. Parmi les méthodes exactes, des branch-and-bound bidirectionnels (qui ordonnancent des tâches à la fois du début et de la fin de l'ordonnancement) ont obtenu de très bonnes performances (récemment un branch-and-bound a même été capable de fermer des instances de grande taille [1, 2]).

Les branch-and-bounds sont des algorithmes qui explorent un espace de recherche sous la forme d'un arbre. La plupart des branch-and-bound mettent l'accent sur la preuve d'optimalité, par l'exploration complète de l'arbre. Dans ce contexte, il est donc naturel de se tourner vers des méthodes de type *Profondeur d'abord (DFS)* ou *Meilleur d'abord (BFS)*. Cependant, il existe des techniques de recherche arborescentes meta-heuristiques (par example *Iterative Beam Search*) qui permettent de focaliser la recherche vers des bonnes solutions rapidement. Nous proposons d'intégrer les stratégies de branchement de branch-and-bounds ainsi que de nouveaux guides dans une beam search itérative. Cet algorithme nous a permis d'obtenir un algorithme simple (moins de 400 lignes de code), et compétitif avec les méthodes de type *Iterated Greedy* et reporter un grand nombre de meilleures solutions connues : 101 pour la minimisation du makespan sur le benchmark VFR, et 51 pour la minimisation du flowtime sur le benchmark de Taillard.

Références

[1] Jan Gmys. Solving large permutation flow-shop scheduling problems on gpu-accelerated supercomputers. *arXiv preprint arXiv :2012.09511*, 2020.

[2] Jan Gmys, Mohand Mezmaz, Nouredine Melab, and Daniel Tuyttens. A computationally efficient branch-and-bound algorithm for the permutation flow-shop scheduling problem. *European Journal of Operational Research*, 284(3) :814–833, 2020.

[3] Luc Libralesso, Pablo Andres Focke, Aurélien Secardin, and Vincent Jost. Iterative beam search algorithms for the permutation flowshop. *arXiv preprint arXiv :2009.05800 (http://librallu.gitlab.io/pdfs/2020_pfsp_ibs.pdf)*, 2020.