Modélisation d'un *flexible job shop* avec opérateurs et setups suspendus non triangulaires pour la production d'acier

Léa Petit-Jean Genat

Hexaly, 251 Boulevard Pereire, Paris, France

lpetitjean@hexaly.com

Mots-clés : *ordonnancement, solveur, modélisation, setups, opérateurs, industrie.*

1 Introduction

Dans le cadre d'un projet mené avec un acteur du secteur sidérurgique, la production d'acier doit être planifiée en séquences d'opérations avec une flexibilité dans le choix des machines. Le problème s'apparente au problème académique du *flexible job shop* mais s'avère bien plus complexe en raison de nombreuses contraintes métier.

Pour traiter de telles instances, nous utilisons Hexaly [1], un solveur d'optimisation mathématique basé sur différentes techniques de recherche opérationnelle, combinant des méthodes exactes, telles que la programmation linéaire, non linéaire et par contraintes, et heuristiques, comme la recherche locale. Son formalisme de modélisation ensembliste permet d'écrire des modèles très compacts, ce qui permet de traiter de grandes instances. Le problème est de grande taille (environ 1000 opérations) ce qui rend les autres formalismes de modélisation difficiles à envisager en raison du très grand nombre de variables de décision nécessaires.

Les sections suivantes présentent le problème métier et l'approche de modélisation ensembliste d'Hexaly, comparée à d'autres modélisations, pour résoudre efficacement la planification de production d'acier.

2 Problème métier

Les caractéristiques classiques du problème académique du *flexible job shop* sont le centre du problème métier. Plusieurs commandes de produits en acier doivent être produites en respectant des dates de disponibilité (*release date*) et d'échéance (*due date*). Chaque commande nécessite une séquence spécifique d'opérations sur différents types de machines, telles que des machines de tréfilage, des grenailleuses ou des fours. Le temps d'exécution des opérations dépend de la machine utilisée et peut inclure des délais de transport entre les opérations. Des temps de *setup* sont nécessaires entre les différentes opérations d'une même machine. L'objectif est d'attribuer une machine et une plage d'exécution à chaque opération afin de minimiser les délais de production ainsi que les temps de *setup*.

D'autres contraintes métier rendent le problème plus complexe. En effet, chaque machine est associée à un ou plusieurs opérateurs et leurs disponibilités déterminent l'emploi du temps de la machine, incluant les maintenances planifiées. L'exécution des opérations ainsi que le réglage des machines nécessitent la présence d'opérateurs. En leur absence, les opérations et les temps de *setup* sont suspendus. Certaines machines nécessitent des temps de *setup* supplémentaires lorsqu'un changement de caractéristiques du produit, comme la taille, survient entre deux opérations consécutives. Dans le cas de fours, les temps de *setup* dépendent de la température, qui varie d'une opération à l'autre. Passer directement d'une opération froide à une opération chaude peut prendre plus de temps qu'intercaler une opération à température intermédiaire, ce qui crée des temps de *setup* qui ne respectent pas l'inégalité triangulaire.

3 Modélisation ensembliste du problème

Pour modéliser le problème métier, le formalisme de modélisation ensembliste d'Hexaly propose des variables d'intervalles pour représenter les plages d'exécution des opérations et des variables de listes [1] pour décider l'ordre des opérations sur chaque machine.

La disponibilité des opérateurs est représentée par un tableau d'entiers valant 1 si l'opérateur est disponible et 0 sinon, pour chaque unité de temps. Le temps d'exécution effectif d'une opération correspond alors à la somme d'unités de temps pendant lesquelles l'opérateur est disponible au cours de sa plage d'exécution.

Ces contraintes s'expriment naturellement grâce à l'opérateur de somme variadique, qui calcule la somme des valeurs renvoyées par une fonction lambda. La somme variadique est donc appliquée sur la variable d'intervalle représentant l'opération et doit être égale au temps d'exécution prévu sur la machine choisie par le solveur, identifiée à l'aide de l'opérateur *find*.

En suivant le même principe, il est également possible de modéliser les temps de *setup* suspendus, comme nous le montrerons dans la présentation.

```
operation[0...nbOperations] <- interval(0, maxEnd);
ordre[0...nbMachines] <- list(nbOperations);
machine[i in 0...nbOperations] <- find(ordre, i);

for [i in 0...nbOperations]
    constraint sum(operation[i], t => dispo[machine[i]]) == processingTime[machine[i]][i];
```

4 Résultats et conclusion

Le formalisme de modélisation ensembliste et non-linéaire d'Hexaly permet de formuler simplement et de façon compacte les contraintes du problème, avec seulement $O(n+m)$ variables de décision pour les contraintes de *non-overlap*, n et m étant respectivement le nombre d'opérations et de machines. La complexité du modèle, présentée dans le tableau 1, est bien plus élevée avec les autres formalismes de modélisation.

Afin de comparer la résolution de ce modèle ensembliste avec d'autres méthodes, nous avons considéré des instances de *flexible job shop* de la littérature avec 1000 opérations [2]. Les solutions d'Hexaly en 10 minutes sont en moyenne 8,5% meilleures que celles de CP Optimizer en 6 heures. OR-Tools, Gurobi et Cplex ne trouvent aucune solution réalisable dans cette même limite de temps.

TAB. 1 – Comparaison des solveurs sur les instances *flexible job shop* de taille 1000.

	Hexaly	**Gurobi**	**Cplex**	**CP Optimizer**	**OR-Tools**
	14.0	11.0.1	22.1.0	12.10.0	9.14
Écart moyen	0%	–	–	+8.5%	–
Complexité	$O(n+m)$	$O(n^2m)$	$O(n^2m)$	$O(nm)$	$O(nm)$

Références

[1] F. Gardi, T. Benoist, J. Darlay, B. Estellon, et R. Megel. *Mathematical Programming Solver Based on Local Search*, Wiley, 2014

[2] G. Da Col et E. C. Teppan. Industrial-size job shop scheduling with constraint programming. *Operations Research Perspectives*, 9 :100249, 2022. https ://www.sciencedirect.com/science/article/pii/S2214716022000215

1. Variable de décision dont la valeur est une permutation d'un sous-ensemble de $\{0, \ldots, n-1\}$