# Recherche tabou pour le jobshop: temps fixe versus nombre d'itérations fixe

Julien Bernard[1], Hervé Manier[2], Marie-Ange Manier[2], Israël Tsogbetse[2]

[1] FEMTO-ST, CNRS, UFC, 25000 Besançon, France
julien.bernard@femto-st.fr
[2] FEMTO-ST, CNRS, UTBM, 90010 Belfort, France
{herve.manier, marie-ange.manier, israel.tsogbetse}@utbm.fr

**Mots-clés** : *jobshop, recherche tabou, codage, voisinage*

## 1 Introduction

La recherche tabou est une métaheuristique qui a été très utilisée par le passé pour calculer des solutions de bonne qualité pour le problème du jobshop. Dans les expériences réalisées à l'aide de cette méthode, on peut : soit fixer le temps alloué à la métaheuristique pour tenter de trouver la meilleure solution possible et dans ce cas, le nombre d'itérations peut varier en fonction du codage et du voisinage utilisés dans la métaheuristique ; soit fixer un nombre d'itérations et dans ce cas, le temps pour effectuer ces itérations peut varier considérablement.

Nous proposons une série d'expériences dans laquelle nous utilisons un ensemble de codages et de voisinages pour résoudre des problèmes de jobshop à l'aide d'une recherche tabou, en fixant soit le temps alloué, soit le nombre d'itérations. Puis nous comparons les résultats ainsi obtenus.

## 2 Jobshop et recherche tabou

Le problème de jobshop est un problème classique d'ordonnancement dans lequel $n$ tâches *(job)* doivent être exécutées sur $m$ machines. Chaque tâche $i$ est constituée de $N_i$ opérations. La $j$-ième opération de la tâche $i$, notée $O_{ij}$, est affectée à une machine donnée et a un temps opératoire $p_{ij}$ donné. On note $N = \sum_{i=1}^{n} N_i$ le nombre total d'opérations. L'objectif est de réduire le temps de complétion maximale $C_{\max}$.

Pour résoudre le problème de job shop, la recherche tabou a beaucoup été utilisée. Taillard [4] choisit une recherche tabou pour résoudre le problème de jobshop. Il utilise un voisinage consistant à échanger deux opérations successives sur le chemin critique. Nowicki et al [2] utilise des techniques avancées pour calculer le voisinage dans le but d'améliorer la recherche tabou précédente. Plus récemment, Strassl et al [3] utilise plusieurs algorithmes pour résoudre le problème de job shop et montre que la recherche tabou donne d'excellents résultats parmi ceux-ci.

## 3 Expériences et résultats

Pour nos expériences, nous utilisons les 242 instances classiques répertoriées par Hoorn [1] (`abz`, `dmu`, `ft`, `la`, `orb`, `swv`, `ta`, `yn`), ainsi que 2500 instances générées par Strassl et al [3]. Les codages utilisés dans la recherche tabou sont : `job` (un vecteur de taille $N$ qui contient $N_i$ fois la tâche $i$), `ope` (un vecteur de taille $N$ qui contient une permutation des $N$ opérations), `mch` (un vecteur de $m$ listes ordonnées d'opérations affectées à chacune des machines), `tim` (un vecteur de $N$ entiers qui indiquent la durée minimale entre le début d'une opération et la fin

| Combinaison | Instances classiques | | Instances générées | |
|---|---|---|---|---|
| | $T = 10s$ | $I = 5000$ | $T = 10s$ | $I = 5000$ |
| `job-ins` | 1.19 | 1.17 | 2.59 | 2.59 |
| `mch-adj` | 3.11 | 4.87 | 7.24 | 7.35 |
| `ope-ins` | 3.44 | 3.54 | 6.05 | 5.84 |
| `job-swp` | 3.61 | 2.38 | 3.74 | 4.05 |
| `ope-swp` | 3.81 | 5.09 | 6.27 | 6.75 |
| `tim-tim` | 6.33 | 4.40 | 3.06 | 3.08 |
| `mch-cas` | 6.51 | 6.25 | 3.06 | 2.22 |
| `ope-rev` | 7.49 | 9.12 | 9.35 | 9.63 |
| `ope-cas` | 7.92 | 7.52 | 3.72 | 3.92 |
| `job-rev` | 8.99 | 8.64 | 8.11 | 8.32 |
| `mch-ins` | 10.81 | 10.39 | 12.31 | 12.28 |
| `ope-adj` | 11.55 | 12.60 | 13.66 | 13.77 |
| `job-adj` | 11.83 | 12.67 | 13.66 | 13.77 |
| `mch-swp` | 12.69 | 12.43 | 12.40 | 12.38 |
| `mch-rev` | 13.36 | 13.03 | 12.37 | 12.30 |

TAB. 1 – Rangs moyens des combinaisons codage-voisinage pour les instances classiques et générées, pour chacune des deux conditions d'arrêt $T = 10s$ ou $I = 5000$

de l'opération précédente). Dans ce dernier codage, une fois l'ordonnancement obtenu, il est rendu actif en supprimant les délais non-nécessaires. Les voisinages utilisés dans la recherche tabou sont : `ins` (insertion d'un élément à un emplacement aléatoire), `swap` (échange de deux éléments aléatoires), `rev` (inversion des éléments entre deux éléments aléatoires), `adj` (échange de deux éléments adjacents aléatoires), `cas` (échange de deux opérations critiques dans un bloc critique, c'est-à-dire une suite d'opérations critiques affectées à la même machine) et `tim` (modification des durées en utilisant une distribution binomiale négative). Avec ces notations, la variante utilisée par Taillard [4] est `mch-cas`. À noter que toutes les combinaisons entre codages et voisinages ne sont pas possibles (`cas` ne peut s'appliquer que sur `ope` et `mch` ; `tim` ne s'applique que sur `tim`). Au total, nous avons quinze combinaisons de codages-voisinages.

Pour chaque instance et chaque combinaison, on effectue dix exécutions et on garde la meilleure des dix exécutions. Puis on classe les quinze combinaisons sur la base de cette meilleure exécution (en prenant en compte les ex-æquo). Enfin, sur la base de ce classement, on déduit le rang moyen sur l'ensemble des instances considérées et pour la méthode d'arrêt considérée. Le temps alloué a été fixé à $T = 10s$, le nombre d'itérations a été fixé à $I = 5000$.

La table 1 montre que, entre les deux conditions d'arrêt, l'écart absolu entre les rangs est en moyenne de 0.79 pour les instances classiques et de 0.2 pour les instances générées. Malgré des écarts relatifs très importants pour le nombre d'itérations en cas de temps fixe (facteur de 1 à 210) ou pour le temps alloué en cas d'itérations fixes (facteur de 1 à 54), les deux conditions donnent sensiblement les mêmes résultats sur ce problème.

## Références


[1] Jelke J. Hoorn. The current state of bounds on benchmark instances of the job-shop scheduling problem. *J. of Scheduling*, 21(1) :127–128, feb 2018.

[2] Eugeniusz Nowicki and Czesław Smutnicki. An advanced tabu search algorithm for the job shop problem. *Journal of Scheduling*, 8 :145–159, 2005.

[3] Simon Strassl and Nysret Musliu. Instance space analysis and algorithm selection for the job shop scheduling problem. *Computers & Operations Research*, 141 :105661, 2022.

[4] Eric D Taillard. Parallel taboo search techniques for the job shop scheduling problem. *ORSA journal on Computing*, 6(2) :108–117, 1994.