RAIRO-Oper. Res. 50 (2016) 83–90
DOI: 10.1051/ro/2015013

**RAIRO Operations Research**
www.rairo-ro.org

# METAHEURISTICS TO SOLVE A TASKS SCHEDULING PROBLEM IN PARALLEL IDENTICAL MACHINES WITH UNAVAILABILITY PERIODS

Rachid zitouni[1] and Omar selt[2]

**Abstract.** In this paper, we introduce an approach for scheduling problems of $n$ tasks on $m$ identical parallel machines with unavailability periods. This problem is strongly NP-complete which makes finding an optimal solution looks impossible task. In this frame, we suggest a novel heuristic in which availability periods of each machine are filled with the highest weighted tasks. To improve the performance of this heuristic, we have used, on one hand, different diversification strategies with the aim of exploring unvisited regions of the solution space, and on the other hand, two well-known neighborhoods (neighborhood by swapping and neighborhood by insertion). The computational experiment was carried out on three identical parallel machines with different availability periods. It must be mentioned that tasks movement can be within one machine or between different machines. The performance criterion to optimize in this problem is the weighted sum of the end dates of tasks. Note that all data in this problem are integer and deterministic.

**Mathematics Subject Classification.** 90C27, 90C59, 90B35, 68M20.

Received February 19, 2014. Accepted April 8, 2015.

## 1. Introduction

A scheduling problem consists in organizing tasks realization time with consideration of time constraints (time limits, tasks series character) and constraints related to using and availability of required resources. The scheduling constitutes a solution to the considered problem, describes the tasks execution and resources location during time and aims to satisfy one or many objectives.

A scheduling problem under machines availability constraints has been studied by many authors. For example the problem $P_m//N-\boldsymbol{C}//C_{\max}$ has been studied by Lee [14–16], Schmidt [22] and Yun-Chia *et al.* [24]. The tabu search is a metaheuristic originally developed by Glover [7, 8] and independently by Hansen [9]. This method combines a local search procedure with a certain number of rules and mechanism which allows surmounting the obstacle of local optima without cycling. Toward furthermore, it proved hight efficiently in resolution of the problems NP-complete and approximate more the optimal solution.

The scheduling problem of a single machine with minimization of the weighted sum of the end dates of tasks. without unavailability constraint is optimally resolved by using the WSPT (weighted shortest processing time) rules. The case of several machines is studied by many authors like Belouadeh [5], Haouari [8] and Sadfi [19].

*Keywords.* Scheduling, parallel identical machines, unavailability periods, metaheuristic, Tabu search.

[1] Department of Mathematics, Ferhat Abbas University of Setif 1, Algeria. rzitou@yahoo.fr

[2] Department of Mathematics, University of M'sila, Algeria. Selt.omar@yahoo.fr

In 1984, Schmidt [21] has studied the scheduling problem of parallel identical machines with different unavailability intervals and different tasks deadlines. He used the method of Branch and Bound based on two procedures: the first is the generation by decomposition and cut approach and the second is the hybridization of procedures of generation by cut. He also built an admissible preemptive scheduling of a complexity $O\left(n/m \ln n\right)$ where $n$ is the number of tasks and $m$ is the number of machines. In 2000, Lee and Chen [17] have studied the simultaneous scheduling of production works and maintenance activities in parallel identical machines to minimize the weighted sum of the end dates of tasks. They have studied two cases: the first, with sufficient number of resources, concerns the case where several machines can be checked up simultaneously (overlap of unavailability periods). The second case, with insufficient number of resources, concerns the case where only one machine can be checked up (overlap of unavailability periods not allowed). They could demonstrate that even if all tasks have the same weight, the problem is NP-hard. They proposed the method of Branch and Bound based on the approach of columns generation to solve the two cases. They have published an experimental study on average size instances.

In 2012, Adamu and Adewunmi [2] have studied the problem $P_m// \sum_{j=1}^{n} w_j \left(U_j + V_j\right)$, they proposed some metaheuristics for scheduling on parallel identical machines to minimize weighted number of early and tardy jobs.

In 2013, they carried out a comparative study of different metaheuristics for identical machines (a genetic algorithm, particle swarm optimization and simulated annealing with their hybrids) [2]. In this paper, we exploit the results obtained by Adamu and Adewunmi for developing a different new metaheuristic to solve the scheduling problem under different constraints.

## 2. Problem statement

This problem consists in scheduling $n$ tasks for $m$ parallel identical machines $\{M_1, M_2, \ldots, M_m\}$ where $n \gg m \geq 2$, with unavailability periods.

We assume that the tasks $\{j_1, j_2, \ldots, j_n\}$ are all available at $t = 0$ and their operation times are independent from the choice of machines performing these tasks. In the generic case of the problem, each one of the $m$ machines can show some unavailability periods during scheduling horizon and each task must be executed onetime.

This problem noted by $P_m//N - C// \sum_{j=1}^{n} w_j C_j$ consists in assigning $n$ tasks to $m$ machines over availability intervals in a manner to enforce the weighted sum of the end dates of tasks. Referred to as $\sum_{j=1}^{n} w_j C_j$ to be minimal.

It must be noted that there is $(n!)^m$ possibility to assign $n$ tasks to $m$ machines [20].

## 3. Neighborhood structure

Neighborhood determination constitutes the most important stage in metaheurstic methods elaboration. In the following part, we use two well know Neighborhoods (neighborhood by swapping) and (neighborhood by insertion).

It must be mentioned that tasks movement can be within one machine and between machines.

### 3.1. Neighborhood by swapping

**Definition 3.1.** Consider a sequence $\sigma$ composed of $n$ tasks. A neighborhood $\sigma^{'}$ is obtained by permuting two tasks. $j$ and $j'$ of respectively $k$ and $k^{'}$ positions $\sigma$ with $k' = k + 1, k + 2, \ldots, n$.

The set $N_1(\sigma) = \{\grave{\sigma}, \grave{\sigma}$ is obtained by permutation of two tasks$\}$ is called "neighborhood of $\sigma$". This set is consequently obtained by permutation of all tasks. of $\sigma$ two by two.

**Formal statement 1**. Consider a sequence $\sigma$, the set's cardinal of $N_1(\sigma)$ is $\frac{n(n-1)}{2}$.

*Proof.* The permutation of all tasks. two by two consists in permuting each task of the sequence with all remained tasks. without identical ones. The number of possible permutations in a sequence $\sigma$ composed of $n$ tasks is:

$$(n-1)+(n-2)+\cdots+2+1=\frac{n(n-1)}{2}\cdot \qquad \square$$

## 3.2. Neighborhood by insertion

**Definition 3.2.** Consider a sequence $\sigma$ composed of $n$ tasks.. A neighborhood $\sigma^{'}$ is obtained by inserting one task $j$ of a position $k$ in a new position $k^{'}$ in the sequence $\sigma$.

The set $N_2(\sigma)=\{\sigma^{'}, \sigma^{'}$ is obtained by inserting a task of position $k$ in $k'\}$ is a neighborhood of $\sigma$. This set is consequently obtained by realizing all possible insertions of all tasks. of $\sigma$.

**Formal statment 2.** Consider a sequence $\sigma$, the set's cardinal of $N_2(\sigma)$ is $(n-1)^2$.

*Proof.* Inserting a task $j$ of position $k$ in an other position $k^{'}$ in the sequence $\sigma$ allows getting $n-1$ possible insertions. Hence, for $n$ tasks., there is $n(n-1)$ insertions to be done. To avoid getting identical sequences, adjacent tasks insertions are counted once. Consequently $n-1$ insertions will be deleted. Finally, the number of obtained insertions is: $n(n-1)-(n-1)=(n-1)^2$. $\square$

## 4. Tabu list structure

The Tabu method is based on the principle that consists in maintaining in memory the last visited solutions and in forbidding the return to them for a certain number of iterations. The aim is to provide sufficient time to the algorithm so it can leave the local optimum. In other words, the Tabu method conserves in each stage a list $L$ of solutions (Tabu's) which it is forbidden to pass-by temporarily. The necessary space for saving a set of solutions tabus in the memory is indispensable.

The list, that we propose, contains the found solutions sequences. After many tests, a dynamic size list, which varies according to the search amelioration state, is conceived. The initial size of this list is considered to be $\dfrac{3\sqrt{n}}{2}$ where $n$ is the tasks. number. After that, during the search, when 5 successive iterations pass without amelioration of solution, the list is reduced to a number inferior or equal to $\sqrt{n}$. On the other hand, when 5 successive iterations pass and the solution is ameliorated, the list is increased to a number superior or equal to $2\sqrt{n}$. The Tabu list is consequently dynamic and its size varies within the interval $[\sqrt{n}, 2\sqrt{n}]$. The decrease or the increase of list size must always be done at the end of the list.

**Heuristic for the problem** $(P)$
An initial solution is always necessary. For this reason, we suggest in this part the following heuristic: assign the (best) task $h$ where $(\frac{p_h}{w_h}=\min\limits_{j\in J}\{\frac{p_j}{wj}\})$ to the best machine (the most available[3]) based on two principles justified by the two following formal statements:

**Formal statement 3.** *In an optimal scheduling, it is necessary to schedule the tasks. in each availability period of the machine according to the order SWPT.*

*Proof.* It results directly by adjacent task exchange like used by Smith [16] for the corresponding periods. $\square$

[3]A machine is supposed to be the most available if it has an availability period the most close to $t=0$ and it is able to realize the required task.

**Formal statement 4.** It is not useful to let the machine (idle) if a task can be assigned to this machine.

Next we present some useful notations before giving the detailed description of our algorithm.

**Notations.**

We denote by:

$J = \{1, 2, \ldots, n\}$: the set of tasks.
$p_h$: execution time of the task $h$.
$I = \{1, 2, \ldots, m\}$: the set of machines
$I_{NA}$: the set of non-assigned machines.
$\alpha$: Number of availability zones.
$Z = \{1, 2, \ldots, \alpha\}$: availability zones.
$S_z^{(i)}$ $(z \in Z)$: the beginning of the unavailability time of the machine $i \in I$.
$T_F$: Final time.
$T_z^{(i)}$ $(z \in Z)$: the end of the unavailability time of the machine $i \in I$.
$\sigma_z^{(i)}$ $(z \in Z)$: the set of partial sequences assigned to the machine $i \in I$.
$\sigma_z = \sigma_z^{(1)} \cup \sigma_z^{(2)} \cup \ldots \cup \sigma_z^{(m)}$.
$J_z^{(i)}$ $(z \in Z) = \left\{ j/\ j \text{ task assigned to the machine } i \text{ with } T_z^{(i)} \leq C_j^{(i)} \leq S_z^{(i)} \right\}$.
$C_z^{(i)}$ $(z \in Z)$: execution time of the task $j \in J_z^{(i)}$.

We assume that for each task $h \in J$, there is at least one machine $i \in I$ such that $P_h^{(i)} \leq S_z^{(i)} - T_z^{(i)}$.

## 5. Algorithm

```
Initialization

$J = \{1, 2, \ldots, n\}$, $I = \{1, 2, \ldots, m\}$; $Z = \{1, 2, \ldots, \alpha\}$; $I_{NA} = I$;
$S_\alpha^{(i)} = T_F$ (given), $\sigma = \emptyset$ $f_\sigma = 0$, $z = 1, C_z^{(i)} = 0$ and $T_1^{(i)} = 0$.
Sort task $h \in J$ in increasing order according to the criterion $p_h/w_h$ in a list
$L_1$
Sort task $h \in J$ in increasing order according to the criterion $p_h$ in a list $L_2$
While
($L_1 \neq \emptyset$ and $z \leq \alpha$) do
Begin
  Set $p_{h_1} = p_h/w_h$ from the top list of $L_1$.
      $p_{h_2} = p_h$ from the top list of $L_2$.
  Determine the machine $k \in I_{NA}$ and the task $h \in J$ such that
      $S_z^{(k)} - C_z^{(k)} = \max_{i \in I_{NA}} \left\{ S_z^{(i)} - C_z^{(i)} \right\} >= \min(p_{h_1}, p_{h_2})$
  If $\{k\} = \emptyset$ then
  Determine the machine $k \in I$ and the task $h \in J$ such that
      $S_z^{(k)} - C_z^{(k)} = \max_{i \in I} \left\{ S_z^{(i)} - C_z^{(i)} \right\} >= \min(p_{h_1}, p_{h_2})$
  Endif
  If $\{k\} \neq \emptyset$ then
    Begin
    Assigned the task $h$ to the machine $k$
    Delete the task $h$ from the two lists $L_1$ and $L_2$
    Compute $C_z^{(k)} = \sum_{j \in J_z^{(k)}} p_j + T_z^{(k)}$;
    Determine $\sigma_z^{(k)} = \sigma_z^{(k)} \cup \{h\}$ and $f_\sigma = f_\sigma + w_h C_z^{(k)}$;
    Set $I_{NA} = I_{NA} \setminus \{k\}$
    End
  Else
    Begin
      Set $z = z + 1$;   $I_{NA} = I$;
    End
  Endif
End
```

## 6. Computational analysis

### 6.1. Data generation

The heuristic were tested on problems generated with 200 tasks similar to that used in previous studies [1,4,12,18] for each task $j$ an integer processing time $p_j$ was randomly generated in the interval (1, 99) with a weight randomly$w_j$ chosen in interval (1, 10). The search time to define a neighborhood and to determine minimal cost is chosen equal to 90 $s$. The number of machines fixed (3 machines) with 3 availability zones for each machine.

### 6.2. Diversification strategies

The final time to execute this problem is chosen as $T_F = 1200$ s. It is divided according to diversification strategy to two times $T_1$, $T_2$. After many experiments, these periods are chosen as follows:

$T_1 = 700$ s: initial starting time: uses long term memory to store the frequency of the moves executed through of the search.
$T_2 = 500$ s: first restarting time makes use of influential moves.

Table 1 presents:

1- The initial mean values of objective function corresponding to initial sequence.
2- The initial mean values of objective function obtained by using on one hand, the neighborhood by swapping and on the other hand, the neighborhood by insertion.
3- The average times corresponding to the two neighborhoods.
4- The percentage of cost improvement.
5- The best costs.

## 7. Results

The results presented in Table 1 shows clearly that the tabu method based on neighborhood by insertion (with complexity $O(\frac{(n-1)}{3}n^2)$) presents the best costs compared to tabu method based on neighborhood by swapping (with complexity $O(n^2(n-1))$). This is due to the fact that the first neighborhood ensures a faster tasks movement besides that the search space is richer with optimal partial sequences in each availability.

TABLE 1. Percentage of heuristic cost amelioration based on metaheuristic.

| $n$ | Initial cost by heuristic (AC of 5 instances) | Tabu search by Swapping | | | Tabu search by Insertion | | | Best |
|---|---|---|---|---|---|---|---|---|
| | | AC | AT (second) | PIIC | AC | AT (second) | PIIC | Cost |
| 50 | 46290 | 41142 | 131 | 11% | 41012 | 81 | 12% | 41012 |
| | 54 046 | 50 550 | 179 | 6% | 50444 | 139 | 7% | 50444 |
| | 44648 | 43 134 | 146 | 3% | 43 030 | 107 | 4% | 43 030 |
| 100 | 198110 | 179808 | 197 | 9% | 179 640 | 252 | 10% | 179 640 |
| | 176 650 | 169 064 | 314 | 4% | 168 868 | 480 | 5% | 168868 |
| | 202 408 | 186198 | 256 | 8% | 195082 | 418 | 4% | 186 198 |
| 200 | 735 058 | 633 146 | 431 | 14% | 733 038 | 403 | 0.5% | 633 146 |
| | 692042 | 688 068 | 327 | 1% | 688 962 | 498 | 1% | 688 068 |
| | 700 578 | 699 438 | 535 | 0.4% | 699 360 | 443 | 3.8% | 699 360 |

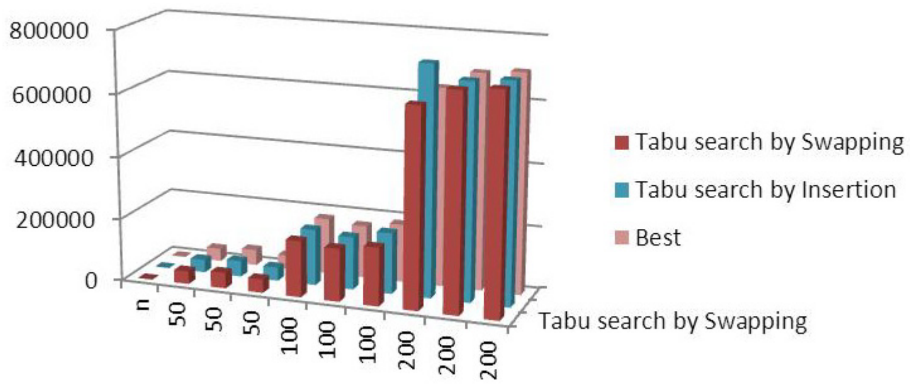

FIGURE 1. Comparison of heuristic and metaheuristic for $n = 200$.

This can also be explained by the nature of used neighborhoods, besides the left shifting of other tasks in the swapping neighborhood.

On the other hand, the heuristic amelioration rate between the two neighborhoods is remarkable (Figs. 1 and 2). It is also noted that the amelioration rate between the proposed heuristic cost and that obtained by the Tabu search is situated between 0.4% and 14%.

## 8. CONCLUSION

In this paper, we proposed novel metaheuristic polynomial approach (Tabu search) to solve a scheduling problem with parallel identical machines and availability periods. The tabu list of this problem is dynamic and its size varies according to the amelioration state of the solution. The developed approach is based on diversification strategy using solution search algorithm that restarts from the point of the solution that was chosen among the earlier best unmaintained found solutions. According to the curried out tests, it is concluded that the proposed approach ensures better results (heuristic amelioration cost up to 14%). It must be noted that the neighborhood by insertion presents the best costs with an acceptable execution time.

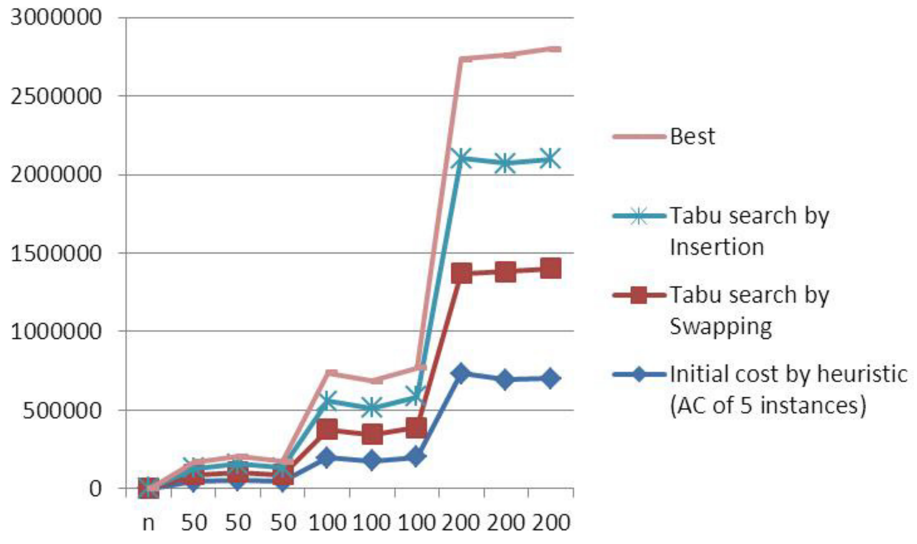

Figure 2. Comparison of heuristic and metaheuristic for $n = 200$.

*Acknowledgements.* The authors kindly acknowledge anonymous referees for their constructive comments and helpful suggestions in improving the readability of this paper.

## References

[1] M.O. Adamu and O. Abass, Parallel machine sheduling to maximize the weighted number of just-in-time jobs. *J. Appl. Sci. Technol.* **15** (2010) 12–34.

[2] M.O. Adamu and A. Adewumi, Metaheuristics for scheduling on parallel machine to minimize weighted number of early and tardy jobs. *Int. J. Phys. Sci.* **7** (2012) 1641–1652.

[3] M. Adamu and A. Adewunmi, A Comparative study of metaheuristics for identical parallel machines. *J. Eng. Technol. Res.* **5** (2013) 207–216.

[4] P. Baptiste, A. Jouglet, CL. Pape and W. Nuijten, *A constraint based approach to minimize the weighted number of late jobs on parallel machines.* Technical Report 2000/228, UMR, CNRS 6599, Heudiasyc, France (2000).

[5] H. Belouadah, M.E. Posner and C.N. Potts, Scheduling with relates dates on a single machine to minimize total weighted completion time. *Discrete Appl. Math.* **36** (1992) 213–231.

[6] J. Carlier and P. Chrétienne, Problèmes d'ordonnancement: Modélisation, Complexité, Algorithmes. Masson, France (1988).

[7] F. Glover, Futurepathsfo rinteger programming and links to artificial intelligence. *Comput. Open Res.* **13** (1986) 533–549.

[8] F. Glover and S. Hanafi, Tabu Search and Finite Convergence. Special Issue on "Foundations of heuristics in Combinatorial Optimization". *Discrete Appl. Math.* **119** (2002) 3–36.

[9] P. Hansen, The Steepest Ascent Mildest Descent Heuristic for Combinatorial Programming. In *Proc. of the Congress on Numerical Methods* (1986).

[10] M. Haouari and T. Ladhari, Branch and bound-based local search method for the flow shop problème. *J. Oper. Res. Soc.* **54** (2003) 1076–1084.

[11] A. Hertz and M. Widmer, Guidelines for the use metaheuristics in combinatorial optimization. *Eur. J. Oper. Res.* **151** (2003) 247–52.

[12] J.C. Ho and Y.L. Chang, Minimizing the number of tardy jobs for m parallel machines. *Eur. J. Oper. Res.* **84** (1995) 334–355.

[13] F. Laburthe, *Contraintes et algorithmes en optimisation combinatoire.* Ph.D. thesis. University Paris VII, France (1988).

[14] C.Y. Lee, Machine scheduling with an availability constraints. *J. Global Optim.* **9** (1996) 395–416.

[15] C.Y. Lee, Minimizing the makespan in two machines flow shop scheduling problem with availability constraints. *Oper. Res. Lett.* **20** (1997) 129–139.

[16] C.Y. Lee, Two machines flow shop scheduling problem with availability constraints. *Eur. J. Oper. Res.* **114** (1999) 420–429.

[17] C.Y. Lee and Z.L. Chen, Scheduling jobs and maintenance activities on parallel machines. *Naval Res. Logist.* **47** (2000) 145–165.

[18] R. M'Hallah and R.L. Bulfin, Minimizing the weighted number of tardy jobs of parallel processors. *Eur. J. Oper. Res.* **160** (2005) 471–484.

[19] C. Sadfi, *Problèmes d'ordonnancement avec minimisation des encours*. Ph.D. thesis. Institut National Polytechnique de Grenoble, France (2002).

[20] M. Sakarovitch, Optimisation combinatoire: Programmation discrète. Hermann, France (1984).

[21] G. Schmidt, Scheduling on semi-identical processors. *Z. Oper. Res.* **A28** (1984) 153–162.

[22] G. Schmidt, Scheduling with limited machine availability. *Eur. J. Oper. Res.* **121** (2000) 1–15.

[23] W.E. Smith, Various optimizes for single-stage production. *Naval Res. Logist.* **3** (1956) 59–66.

[24] L. Yun-Chia, H. Yu-Ming and T. Chia-Yun, Metaheuristics for drilling operation scheduling in Taiwan PCB industries. *Int. J. Prod. Econ.* **141** (2013) 189–198.