PERFORMANCE ENHANCEMENT OF NUMERICAL APPROACHES FOR SCHEDULING PROBLEM ON MACHINE SINGLE

Omar Selt
Laboratory of pure and applied mathematics,msila university, algeria
E-mail: selt.omar@yahoo.fr

Submission: 3/28/2019
Revision: 10/10/2020
Accept: 10/17/2020

ABSTRACT

In this paper, we consider a single-machine scheduling problem, with the aim of minimizing the weighted sum of the completion time. This problem is NP-hard, making the search for an optimal solution very difficult. In this frame, two heuristics (H1), (H2) and metaheuristic tabu search are suggested.

To improve the performance of this techniques, we used, on one hand, different diversification strategies (TES1 and TES2) with the aim of exploring unvisited regions of the solution space. On the other hand, we suggested three types of neighborhoods (neighborhood by swapping, neighborhood by insertion and neighborhood by blocks). It must be noted that tasks movement can be within one period or between different periods.

Keywords: Scheduling; Single machine; NP-hard; Tabu search
1. INTRODUCTION

The scheduling problem of a single machine with minimization of the weighted sum of the tasks' end-dates, 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 (Zribi et al., 2005; Zitouni & Selt, 2016).

Zribi et al., (2005) have studied the problem $1//N-C//\sum_{j} w_{j} C_{j}$ and have compared two exact methods, the Branch and Bound method and the integer programming one. They have concluded that Branch and Bound method has better performance and it allows resolving instances of more than 1000 tasks. Chang et al. (2011) proposed a genetic algorithm (GA) enhanced by dominance properties for single machine scheduling problems to minimize the sum of the job's setups and the cost of tardy or early jobs related to the common due date.

Selt and zitouni (2014) have studied the following problem $(P_{m} // N-C // \sum_{j=1}^{n} w_{j} C_{j})$, carrying out a comparative study of heuristic and metaheuristic for three identical parallel machines.

In this paper, we propose an approach to solve tasks scheduling problem on machine single under unavailability periods.

2. PROBLEM DESCRIPTION

There are $n$ tasks to schedule in a single machine. All the tasks are available at time zero.

- Each task $j$ has associated a processing time $p_{j}$ and a weight $W_{j}$
- There is a time interval $T_{z}$ between the completion times of three consecutive maintenance activities.
- The tasks can not be interrupted.

Objective: To assign tasks to blocks between maintenance activities in such a way that the last task finishes as soon as possible, that is, to minimize the weighted sum of the completion time.

3. PROPOSED METHOD

3.1. Tabu Search
Tabu Search is a metaheuristic originally developed by Glover (1986). This method combines local search procedure with some rules and mechanism to surmount local optima obstacle avoiding the cycling trap. Tabu search is the metaheuristic that keeps track of the regions of the solution space that have already been searched in order to avoid repeating the search near these areas (Glover & Hanafi, 2002).

It starts from a random initial solution and successively moves to one of the neighbors of the current solution. The difference between tabu search and other Meta-heuristic approaches is based on the notion of the tabu list, which is a special short-term memory, storing of previously visited solutions including prohibited moves. In fact, short-term memory stores only some of the attributes of solutions instead of whole solutions. So, it gives no permission to revisit solutions, and then, avoids cycling and being stuck in local optima.

During the local search, only those moves that are not tabu will be examined, if the tabu move does not satisfy the predefined aspiration criteria. These aspiration criteria are used, because the attributes in the tabu list may also be shared by unvisited good quality solutions. A common aspiration criterion is better fitness, i.e. the tabu status of a move in the tabu list is overridden if the move produces a better solution.

The process of Tabu Search (TS) can be represented as follows:

3.2. Algorithm (TS)

Step 1 Generate initial solution x.

Step 2 Initialize the Tabu List.

Step 3 While a set of candidate solutions $X''$ is not complete.

Step 3.1 Generate candidate solution x" from current solution x.

Step 3.2 Add x" to X" only if x" is not tabu or if at least one

Step 4 Select the best candidate solution x* in X".

Step 5 If fitness(x*) > fitness(x), then x = x*.

Step 6 Update Tabu List and Aspiration Criteria

Step 7 If the termination condition met, then finish; otherwise, go to Step 3.

3.3. Intensification
One memorizes the best-found solutions and tries to determine common proprieties to define interesting regions and orient the research towards these regions, by considering all the movement that leads to leaving these regions as tabu, for example.

The intensification allows to stop periodically the normal exploration process and to intensify her research effort within a region that seems promising. One of the methods of intensification application is to memorize the best-found solutions to go back to one of these solutions.

3.4. Diversification

This technique is the inverse of the intensification method. It directs the research towards the unexplored regions. Implementing this technique consists in memorizing the solutions the most frequently visited and imposing a penalty system, in order to favor the movement the less frequently used. In this paper, the first starting time is TES₁=25 minutes and the second restarting time is TES₂=20 minutes; these times are practically sufficiently enough for exploring the majority of regions.

3.5. Neighborhoods

Neighborhood determination constitutes the most important stage in metaheuristic methods elaboration. In the following part; we use three neighborhoods (neighborhood by swapping, neighborhood by insertion and neighborhood insertion by blocks).

Notations:

We denote by:

\[ J = \{1, 2, ..., n\} \] : The set of tasks.

\[ p_h \] : Execution time of the task \( h \).

\( M \) : Single machine

\( k \) : Number of availability zones.

\( Z = \{1, 2, ..., k\} \) : Availability Zones.

\( E_z \) : Period of unavailability zones.

\( \sigma \) : Sequence assigned to the machine \( I \).

\( w_h \) : Weight of the task \( h \).
4. HEURISTICS DESCRIBED

An initial solution is always necessary. For this reason, we suggest in this part the following heuristics based on two principles:

1-assigned the (best) task \( j \) where \( \left( w_j p_j \right) = \min \left( \frac{w_j}{p_j} \right) \) to machine \( M \).

2-assigned the (best) task \( j \) where \( p_n = \max( p_j ) \) to machine \( M \).

4.1. Formal statement

It is not useful to let the machine (idle) if a task can be assigned to this machine(smith,1956).

5. HEURISTICS

5.1. Heuristic (H1)

5.1.1. Initialization

Begin

\[ j=\{1, 2, \ldots, n\} ; E_i=0 ; \sigma=\phi, (\phi)=0 ; p_j = \text{random (1.99)} ; w_j = \text{random (1.10)} ; z=1 \]

Sort tasks \( h \in J \) in increasing order according to the criterion \( \frac{p_j}{w_j} \) in a list \( U \)

While \( (U \neq \phi \text{ and } z_k \geq p_n) \) do

Begin
Set \( p_h = p_h / w_h \) from the top list of \( U \);

**End if**

Assigned the task \( h \) to the machine M;

Delete the task \( h \) from the list \( U \);

Compute \( C_z = \sum p_j + E_z \);

Determine \( \sigma = \sigma \cup \{h\} \) and \( f_\sigma = f_\sigma + w_h C_z \);

**End**

**Else**

Begin

\( z = z + 1 \);

**End**

**End if**

**End**

5.2. **Heuristic (H2)**

**Initialization**

\( j = \{1, 2, \ldots, n\} ; E_i = 0 ; \sigma = \phi ; f(\phi) = 0 ; p_j = \text{random (1.99)} ; w_j = \text{random (1.10)} ; z = 1 \)

Begin

Sort tasks \( h \in J \) in increasing order according to the criterion \( p_j / w_j \) in a list \( U_1 \)

Sort tasks \( h \in J \) in decreasing order according to the criterion \( p_j \) in a list \( U_2 \)

**While** \((U \neq \phi \text{ and } z_k \geq p_h) \text{ do} \)

Begin

Set \( p_{h_1} = p_h / w_h \) from the top list of \( U_1 \)

Set \( p_{h_2} = \max p_h \) from the top list of \( U_2 \)
End if

Assigned the task \( h \) to the machine \( M \);

Delete the task \( h \) from the two lists \( U \) and \( U_2 \);

Compute \( C_z = \sum p_j + E_z \);

Determine \( \sigma = \sigma \cup \{ h \} \) and \( f_\sigma = f_\sigma + w_h C_z \);

End

Else

Begin

Set \( z = z + 1 \);

End

End if

End

5.3. Algorithm

Step 1 Get an initial solution \( \sigma \) and \( T[1] = 0 \);

Step 2 Do permutation by swapping

Step 3 Do permutation by insertion

Step 4 Do permutation by insertion by a bloc

Step 5 Compute: \( f_1 = f_{\text{swapp}}; f_2 = f_{\text{insert}}; f_3 = f_{\text{ins_bloc}} \)

Step 6 Consider \( L = \sqrt{N} \) (Tabu list size)

Step 7 for \( k = 1 \) to 3 Do

If \( f_{\text{init}} < f_k \)

Do: \( T[1] = f_{\text{init}} \);

else \( T[1] = f_k \);

End if

\( T_k = T[1] \);
End

Step 7.1 $f_{\text{best}} = \min (T_1, T_2, T_3)$

End if

Step 7.2 Display $\sigma(f_{\text{best}})$

5.3.1. Example 1

Consider the problem $P_1$ with the following data:

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 |
|-----|---|---|---|---|---|---|
| $P_j$ | 11 | 36 | 88 | 10 | 91 | 31 |
| $W_j$ | 3 | 6 | 8 | 7 | 4 | 1 |
| $P_j/W_j$ | 3.67 | 6 | 11 | 1.42 | 22.75 | 31 |

Results of heuristic (H1) are: $f = 2666$; execution time $= 0.156$ s

Results of tabu (swapping) are: $f = 2145$; execution time $= 0.991$ s

Results of tabu (insertion) are: $f = 2431$; execution time $= 1.024$ s

Results of tabu (insertion by bloc) are: $f = 2567$; execution time $= 0.306$ s

The best results are obtained by using tabu by swapping for $f = 2145$.

5.3.2. Example 2

Consider the problem $P_2$ with the following data:

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 |
|-----|---|---|---|---|---|---|
| $P_j$ | 11 | 36 | 88 | 10 | 91 | 31 |
| $W_j$ | 3 | 6 | 8 | 7 | 4 | 1 |
| $P_j/W_j$ | 3.67 | 6 | 11 | 1.42 | 22.75 | 31 |
| $P_j(\text{MAX})$ | 91 | 88 | 36 | 31 | 11 | 10 |

Results of heuristic (H2) are: $f = 2548$; execution time $= 0.650$ s

Results of tabu (swapping) are: $f = 1986$; execution time $= 0.991$ s

Results of tabu (insertion) are: $f = 2367$; execution time $= 1.542$ s

Results of tabu (insertion by bloc) are: $f = 2410$; execution time $= 0.945$ s

The best results are obtained by using by tabu (swapping) for $f = 1986$.

6. Computational Analysis

6.1. Data generation
The proposed approaches were tested on problems generated with 1000 tasks similar to those used in previous studies (M'Hallah & Bulfin, 2005). For each task $j$, an integer processing time $p_j$ was randomly generated in the interval $(1, 99)$ with a weight $w_j$ randomly chosen in the interval $(1, 10)$.

The tables 1 and 2 below presents:

1) The initial mean values of the objective function corresponding to the initial sequence.
2) The initial mean values of the objective function.
3) The average times corresponding to the three neighborhoods.
4) The best costs.

AC: Average costs.
AT: Average time.

7. RESULTS

The results listed in tables (III and IV) show clearly that the tabu method based on neighborhood by swapping presents the best costs compared with tabu method based on neighborhood by insertion and by blocks.

This is due to the fact that the first neighborhoods ensures a faster tasks movement; besides that, the search space is richer with optimal partial sequences in each availability zones. This can also be explained by the nature of adopted neighborhoods.

The results show that execution time obtained by the proposed neighborhoods is acceptable. On the other hand, the heuristics amelioration rate between the three neighborhoods is remarkable (Figure 1 and 2, Graphic 1 and 2).

8. CONCLUSION

In this paper, $n$ tasks scheduling problem for machine single under availability constraint is discussed, with the aim of minimizing the weighted sum of the completion time. The approach based on tabu search allowed solving this problem, with the enhancement of initial solution obtained by a heuristics ($H_1$ and $H_2$) of complexity $o(n\log n)$.

By considering three types of neighborhoods, tabu list and diversification strategy, the results of tabu search method were encouraging, and they will be more encouraging if good neighborhood based on problem's data is defined.
More encouraging if good neighborhood based on problem's data is defined.

Figure 1: Histogram of heuristic (H1) cost amelioration based on tabu search for different N values.

Graphic 1: Circle graph of heuristic (H1) cost amelioration

Figure 2: Histogram of heuristic (H2) cost amelioration based on tabu search for different N values.
Graph 2: Circle graph of heuristic (H2) cost amelioration based on tabu search.

Table 3: Results obtained by heuristic (H1) and tabu search

| N   | Initial Solution (H1) (average of 3 instances) | Tabu search by swap | Tabu search by insertion | Tabu search by blocs | Best costs |
|-----|------------------------------------------------|---------------------|--------------------------|----------------------|------------|
|     | AC AT (second) | AC AT (second) | AC AT (second) | AC AT (second) | AC AT (second) |           |
| 20  | 29190 0,185 41647 0,654 31779 0,307 35466 0,13 29190 |
| 50  | 45768 0,201 40772 0,578 29096 0,224 32259 0,166 29096 |
| 100 | 30731 0,194 29720 0,576 37763 0,447 33526 0,161 29720 |
| 17886722 | 8,120 9730402 9,789 8964538 10,876 8763401 7,871 8763401 |
| 17986067 | 7,182 9056321 9,562 9254175 10,501 9342104 7,549 9056321 |
| 18410307 | 7,298 8657831 9,861 8765109 10,612 9297364 7,724 8657831 |
| 41931982 | 12,20 34537327 15,87 33762437 16,87 35518023 14,73 33762437 |
| 43858415 | 13,01 37612943 16,08 39576182 16,49 38003183 13,98 37612943 |
| 39895222 | 12,44 36581193 15,83 32789193 16,81 34997710 14,01 32789193 |
| 75377034 | 16,80 67451280 20,563 68673189 25,675 67645329 17,977 67451280 |
| 74190765 | 17,44 66347819 20,926 58976897 25,241 68936103 18,672 58976897 |

Table 4: Results obtained by heuristic (H2) and tabu search

| N   | Initial Solution (H2) (average of 3 instances) | Tabu search by swap | Tabu search by insertion | Tabu search by blocs | Best costs |
|-----|------------------------------------------------|---------------------|--------------------------|----------------------|------------|
|     | AC AT (second) | AC AT (second) | AC AT (second) | AC AT (second) | AC AT (second) |           |
| 20  | 49635 0,897 44957 1,83 44845,66 2,76 45738 1,75 49635 |
| 50  | 41544 1,01 41235 1,72 41094 2,64 40763 1,70 40763 |
| 100 | 34203 0,897 33208 1,97 34008 2,38 33872 1,84 33872 |
| 500 | 202482 4,43 200848 9,25 201830 14,40 199673 9,99 199673 |
| 75377034 | 16,80 67451280 20,563 68673189 25,675 67645329 17,977 67451280 |
| 74190765 | 17,44 66347819 20,926 58976897 25,241 68936103 18,672 58976897 |
REFERENCES

Abdenacer, G. (2014). Metaheuristic algorithms for solving quadratic assignment problem, IJACSA, 5, 15-21.

Adamu M. O., & Adewunmi. A. (2013). Comparative study of meta-heuristics for identical parallel machines, J. Eng. Technol. Res., 5(7), 207-216.

Chang, P.-C., Chen, S.-H., Lie, T., & Liu, J. Y.-C. (2011). A genetic algorithm enhanced by dominance properties for single machine scheduling problems with setup costs, International Journal of Innovational Computing Information and Control, 7(5A), 2323–2344.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res., 13, 533-549.

Glover, F., & S. Hanafi, S. (2002). Tabu Search and Finite Convergence, Special Issue on Foundations of heuristics in Combinatorial Optimization. Discrete Appl. Math, 119, 3-36.

Hansen, P. (2006). The steepest ascent mildest descent heuristic for combinatorial programming, Proceedings of the Congress on Numerical Methods, Capri, Italy.

Haouari. M., & Ladhari, T. (2003). A branch-and-bound-based local search method for the flow shop problem, J. Oper. Res. Soc., 54, 1076-1084. DOI: 10.1057/palgrave.jors.2601612

Lee. C. Y. (1996). Machine scheduling with an availability constraints, J. Global Optim., 9, 395-416.

Lee C. Y. (1997). Minimising the makespan in two machines flow shop scheduling problem with availability constraints, Oper. Res. Lett., 20, 129-139.

M'Hallah, R., & Bulfin, R. L. (2005). Minimizing the weighted number of tardy jobs of parallel processors, Eur. J. Oper. Res., 160, 471-4847.

Sakarovitch, M. (1984). Optimisation combinatoire: Programmation discrete, Hermann, France.

Schmidt, G. (2000). Scheduling with limited machine availability, European J. Oper, 121, 1-15.

Selt, O., & Zitouni, R. (2014). A comparative study of heuristic and metaheuristic for three identical parallel machines, Cjpas, 3147-3153.

Smith, W. E. (1956). Various optimizes for single-stage production, NavaRes.Logist, 3, 59-66.

Liang, Y.-C., Hsiao, Y.-M., & Tien, C.-Y. (2013) Metaheuristics for drilling operation scheduling in Taiwan PCB industries, International Journal of Production Economics, 141(1), 189-198. DOI: 10.1016/j.ijpe.2012.04.014

Zribi, N., Kacem, I., El-Kamel, A., & Borne, P. (2005). Minimisation de la somme des retards dans un job shop flexible, Revue e-STA (SEE), 6(6), 21-25.

Zitouni, R., & Selt, O. (2016). Metaheuristics to solve tasks scheduling problem in parallel identical machines with unavailability periods, RAIORRes, 50(1), 90.