Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 12, Number 5 (2016), pp. 4325–4333

http://www.ripublication.com/gjpam.htm

Numerical solution for solving scheduling problem

Omar Selt

Laboratory of pure and applied mathematics,
Department of Mathematics,
University of M'sila Algeria.

Abstract

In this paper, a numerical solution for solving scheduling problem of n tasks on single machine is proposed. This problem is NP-difficult, making the search for an optimal solution very difficult. In this frame, two heuristics (H_1), (H_2) are suggested. Finally, we make a comparative study. Besides that, all data in this problem are supposed to be integer and deterministic.

AMS subject classification:
Keywords: Scheduling, Single machine, NP-difficult.

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.

It is well known that the majority of scheduling problems are of NP-difficult type, which is practically impossible to find a polynomial algorithm to solve this sort of problems [1–4]. Therefore, many research works were carried out, in this frame, for tasks scheduling problem for one machine or different machines ([5–10]).

Zribi and et al. have studied the problem $1//N - C// \sum_{j=1}^{n} w_j C_j$ 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.

Selt and al have studied the problem $1//N-C//\sum_{j=1}^{n} w_j C_j$ and have compared an heuristic and metaheuristic in single machine [9].

Chang et al. 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.

In this paper, we propose a polynomial approach to solve tasks scheduling problem in single machine , based on two heuristics (H_1 and H_2).

2. Problem Statements

Tasks scheduling problem under availability constraint of the machine is defined as follow: Starting dates and durations of unavailability zones (E_Z) of any machine are defined and known initially.

Each machine can realize only one task at a time, and each task requires only one machine to be realized.

All tasks j are available at the beginning of scheduling. These tasks are characterized by:

1. Their treatment time p_j.
2. Their weigh w_j

The set $J = (1, 2, \ldots, n)$ contains n independent tasks where each one is executed once. Tasks to be scheduled are strictly preemptive, which means that the execution of each task cannot be interrupted neither by unavailability period nor by the execution of another task. The objective of scheduling is to determine the input sequence of the machine's tasks such as the weighted sum of tasks' ending date ($w_j C_j$) be minimal.

3. Tabu Search

Tabu Search is a metaheuristic originally developed by GLover (1986 and 1999). 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 [16–17].

Algorithm 1
Step 1 Generate initial solution x.
Step 2 Initialize the Tabu List.
Step 3 While 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 termination condition met, then finish; otherwise, go to Step 3.

4. Amelioration techniques

Among strategies that are recently proposed to raise the efficiency of tabu search method, there are intensification technique and diversification technique.

4.1. 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; theses times are practically sufficiently enough for exploring the majority of regions.

4.2. Neighborhoods

Neighborhood determination constitutes the most important stage in metaheurstic methods elaboration [15]. In the following part, we use (neighborhood by insertion).

Notations:
We denote by:
$J = 1, 2, \ldots, n$: The set of tasks.
p_h: Execution time of the task h.
M: Single machine
k: Number of availability zones.
$Z = 1, 2, \ldots, k$: Availability zones.
E_z: Period of unavailability zones.
σ: Sequence assigned to machine M.
w_h: Weight of the task h.
C_h: Execution time of the task h by the machine M.
$C_z(z \in Z)$: Execution time of the task $j \in J$, allocated to the zone z.
$f(\sigma)$: Objective function cost.
f_{insert}: Insertion algorithm cost.
f_{best}: Minimal cost.
T: Tabu List.
L: Tabu Liste Size.

4.3. Formal statement 1

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

5. Heuristic (H_1)

Initialization
$j = \{1, 2, \ldots, n\}$; $E1 = 0$; $\sigma = \emptyset$; $f(\emptyset) = 0$; $p_j = random(1..99)$;
$w_j = random(1..10)$; $z = 1$
Begin
Sort tasks $h \in J$ in increasing order according to the criterion $\frac{p_j}{wj}$ in a list U
While ($U \neq \emptyset$ and $z_k \geq p_h$) **do**
Begin
Set $p_h = \frac{p_j}{wj}$ from the top list of U;
Endif
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
Set $z = z + 1$;
End
End if
End

6. Heuristic (H_2)

Initialization
$j = \{1, 2, \ldots, n\}$; $E1 = 0$; $\sigma = \emptyset$; $f(\emptyset) = 0$; $p_j = random(1..99)$;
$w_j = random(1..10)$; $z = 1$
Begin
Sort tasks $h \in J$ in increasing order according to the criterion $\frac{p_j}{wj}$ 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 ($U \neq \emptyset$ and $z_k \geq p_h$) **do**
Begin
Set $p_{h_1} = \frac{p_j}{wj}$ from the top list of U_1;
Set $p_{h_2} = \max p_h$ from the top list of U_2;
Endif
Assigned the task h to the machine M;
Delete the task h from the two lists U_1 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

7. Algorithm

Step 1 Get an initial solution σ and $T[1] = 0$;
Step 2 Do permutation by insertion;
Step 3 Compute: $f_1 = f_{insert}$;
Step 4 Consider L(Tabu list size)
Step 5 for $k = 1$ to 2 **Do**
If $f_{init} < f_k$ Do $T[1] = f_{init}$;
else $T[1] = f_k$;
End if
$Tk = T[1]$;
End
Step 5.1 $f_{best} = \min(T1, T2)$
End if
Step 5.2 Display $\sigma(f_{best})$

Example 1
Consider the problem P_1 with the following data:

Table 1: Tasks Scheduling Results for Example 1

j	1	2	3	4	5	6
P_j	11	36	88	10	91	31
w_j	3	6	8	7	4	1
$\frac{P_j}{w_j}$	3.66667	6	11	1,42	22.75	31

Results of heuristic (H_1) are: $f = 2666$; execution time = 0,156 s
Results of tabu (insertion) are: $f = 2431$; execution time = 1,024 s

Example 2
Consider the problem P_2 with the following data:
Results of heuristic (H_2) are: $f = 2548$; execution time = 0,650 s
Results of tabu (insertion) are: $f = 2367$; execution time = 1,542 s

Table 2: Tasks Scheduling Results for Example 2

j	1	2	3	4	5	6
P_j	11	36	88	10	91	31
w_j	3	6	8	7	4	1
$\frac{P_j}{w_j}$	3.66667	6	11	1,4	22.75	31
$P_j(MAX)$	91	88	36	31	11	10

Table 3: Result obtained by Heuristic (H_1) and Tabu Search

N	**Initial Solution (H1) (average of 3 instances)**		**Tabu search by insertion**		**Best costs**
	AC	**AT (second)**	**AC**	**AT (second)**	
N =20	**29190**	**0,185**	**31779**	**0,307**	**29190**
	45768	**0,201**	**29096**	**0,224**	**29096**
	30731	**0,194**	**37763**	**0,447**	**30731**
N=50	**205130**	**0,583**	**223921**	**1,524**	**205130**
	207358	**0,429**	**219091**	**1,019**	**207358**
	214071	**0,919**	**194182**	**0,973**	**194182**
N=100	**734976**	**2,212**	**593040**	**5,179**	**593040**
	1994099	**2,191**	**786843**	**5,356**	**786843**
	839684	**1,997**	**685365**	**5,808**	**685365**
N=250	**5090272**	**5,90**	**4659201**	**9,18**	**4659201**
	4897215	**6,14**	**4061839**	**9,63**	**4061839**
	4658617	**6,142**	**3312510**	**8,89**	**3312510**

8. Computational analysis

8.1. Data generation

The proposed approaches were tested on problems generated with 1000 tasks similar to those used in previous studies (Ho and Chang,1995; M'Hallah and 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 interval (1,10).

The tables 1 and 2 below presents:

Table 4: Result obtained by Heuristic (H_2) and Tabu search

N	**INITIAL SOLUTION (H2) (AVERAGE OF 3 INSTANCES)**		TABU SEARCH BY INSERTION		**BEST COSTS**
	AC	**AT (SECOND)**	AC	AT (SECOND)	
N =20	49635	0,897	**44845**	2,76	**44845**
	41544	1,01	41094	2,64	41094
	34220	**0,99**	34008	2,38	34008
N=50	202482	4,43	201830	14,40	201830
	220786	6,65	**210600**	16,03	**210600**
	230501	3,98	**226582**	14,19	**226582**
N=100	903625	**7,65**	856713	28,09	856713
	863040	**9,86**	**786173**	30,53	**786173**
	875476	**10,23**	855364	27,98	855364
N=250	989476	**20,14**	985476	35,80	985476
	1098437	28,76	**906718**	42 ,71	906718
	1287142	27,97	973027	40,93	973027

1- The initial mean values of objective function corresponding to initial sequence.
2- The initial mean values of objective function.
3- The average times corresponding to the neighborhood.
4- The best costs.
AC: Average costs.
AT: Average time

9. Conclusion

In this paper, two heuristics (H_1,H_2) for solving scheduling problem are compared to each other. In the two approaches, metaheuristic tabu search and neighborhood by insertion are used. Firstly, heuristic H1 is executed; then with the same problem's parameters, the second heuristic H_2 is executed. The times reported in the tables present the heuristics execution time. The cost of objective function of the initial sequence are also reported in the tables.

It is remarked that the heuristic H_1 is more flexible than the heuristic H_2, since this latter performs more iterations to get an amelioration. For problems with bigger size,

the amelioration is minor for the neighborhood. Nevertheless, it is clear that the choice of good neighborhood diversification, which offers more possible neighbors will have the best results.

References

[1] M. Sakarovitch, Optimisation combinatoire: Programmation discrete, Hermann, France, 1984.

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

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

[4] J. Ho, and YL. Chang, (1995), Minimizing the number of tardy jobs for m paralell machines, Eur. J. Oper. Res, pp. 334–355.

[5] C. Sadfi, Problèmes d'ordonnancement avec minimisation des encours Thèse PhD., Institut National Polytechnique de Grenoble, France, 2002.

[6] N. Zribi, I. Kacem, A. El-Kamel and P. Borne, (2005), Minimisation de la somme des retards dans un jobshop flexible, Revue e-STA (SEE), 6(6), pp. 21–25.

[7] P. C. Chang, Shih-Hsin, Lie, Ting, Liu, Julie Yu-Chih, (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), pp. 2323–2344.

[8] O. Selt and R. Zitouni, (2014), A comparative study of heuristic and metaheuristic for three identical parallel machines, Cjpas, pp. 3147–3153.

[9] O. Selt et al. (2016), An approach for scheduling on single machine, IJCSIS, V 14, N 7, pp. 879–883.

[10] R. Zitouni and O. Selt, (2016), Metaheuristics to solve tasks scheduling problem in parallel identical machines with unavailabilite periodes, RAIRO. Oper. Res, 50(1), pp. 83–90.

[11] R. Zitouni and O. Selt, (2014), Impoved tabu search scheduling problem in parallel identical machines, RJASET, Vol. 8 N3, pp. 423–428.

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

[13] P. King-Wah, (2013), A genetic algorithm based heuristic for two machine nowait flowshop scheduling problems with class setup times that minimizes maximum lateness, International Journal of Production Economics, 141, pp. 127–136.

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

[15] L. Yun-Chia, H. Yu-Ming and Chia-Yun, (2013), Metaheuristics for drilling operation scheduling in Taiwan PCB industries, Int. J. Prod. Econ, 141(1), pp. 189–198.

[16] F. Glover, (1986), Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res, 13, pp. 533–549.

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