INDEPENDENT JOURNAL OF MANAGEMENT & PRODUCTION (IJM&P)
http://www.ijmp.jor.br v. 13, n. 1, January-march 2022
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

INNOVATION STRATEGY IN INDUSTRY: CASE OF THE SCHEDULING PROBLEM ON PARALLEL IDENTICAL MACHINES

Omar Selt
Laboratory of pure and applied mathematics, M'sila university, France
E-mail: selt.omar@yahoo.fr

Submission: 12/25/2020
Revision: 1/20/2021
Accept: 1/29/2021

ABSTRACT

In this paper, we propose an innovation strategy in the industry (case of the scheduling problem on two parallel identical machines), with the objective of minimizing the weighted sum of the end dates of jobs, this problem is NP-hard. In this frame, we suggested a novel heuristics: (H1), (H2), (H3), with two types of neighborhood (neighborhood by SWAP and neighborhood by INSERT). Next, we analyze the incorporation of three diversification times (T1), (T2), and (T3) with the aim of exploring unvisited regions of the solution space. It must be noted that job movement can be within one zone or between different zones. Computational tests are performed on 6 problems with up to 2 machines and 500 jobs.

Keywords: innovation; Scheduling; parallel identical machines

1. INTRODUCTION

A scheduling problem consists of organizing jobs realization time with consideration of time constraints (time limits, tasks series character) and constraints related to using and availability of required resources.

The case of scheduling problems on parallel identical machines is studied by many authors like (Schmidt, 1984; Zribi & al, 2005; Chang & al, 2011; Adamu & Adewunmi, 2012, 2013; Selt & zitouni, 2016).

In (1984) Schmidt has studied the scheduling problem of parallel identical machines with different unavailability intervals and different job 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.

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Zribi and al (2005) have studied the problem $1//N-\boldsymbol{C}//\sum_{j=1}^{n} w_j C_j$ and have compared two exact methods, the Branch and Bound method and the integer programming one. They have concluded that the Branch and Bound method has better performance and it allows resolving instances of more than 1000 tasks.

Chang and 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.

Adamu and Adewunmi (2012, 2013) have studied the problem $P_m // \sum_{j=1}^{n} w_j (U_j + V_j)$, they proposed some metaheuristics for scheduling problem on parallel identical machines to minimize a weighted number of early and tardy jobs.

In (2013), they carried out a comparative study of different (a genetic algorithm, particle swarm optimization and simulated annealing with their hybrids) metaheuristics for identical machines

Zitouni and Selt (2016) have studied the problem $P_m // N-\boldsymbol{C}// \sum_{j=1}^{n} w_j C_j$ they proposed a novel heuristic for scheduling problems on parallel identical machines to minimize the weighted sum of the end dates of tasks.

In this paper, the results of Zitouni and Selt research works are exploited to develop a different new approach to solve job scheduling problems on parallel identical machines under different constraints.

2. PROBLEM DESCRIPTION

This problem consists in scheduling n jobs for m parallel identical machines $\{M_1, M_2, ..., M_m\}$ where $n >> m \geq 2$ with unavailability zones.

We assume that the jobs $\{j_1, j_2, ..., j_n\}$ are all available at $t = 0$ and their operation times are independent of the choice of machines performing these jobs.

In the generic case of the problem, each one of the m machines can show some unavailability zones during scheduling horizon and each job must be executed on time.

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

This problem noted by $P_m \,//\, N - \boldsymbol{C} \,//\, \sum_{j=1}^{n} w_j C_j$ consists in assigning n jobs to m machines over availability zones 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 jobs to m machines (Sakarovitch, 1984).

3. PROPOSED METHOD

3.1. Tabu Search (TS)

Tabu Search is a metaheuristic originally developed by (Glover, 1986).

This method combines local search procedures with some rules and mechanisms to surmount local optima obstacles 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.

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

3.2. Algorithm (TS)

Table 1: The process of (TS) can be represented as follows:

Initialization: **X = initial solution , f_{min} = f (x), X: = x**
Step 1: **generate a neighborhood N (x)**
Step 2: **f (x ') = [f (x_i)]**
Steps 3: add **(x ', TABU)**
Step 4: **x: = x '**
Step 5: **If f (x) < f min**
Step 6: **f min: = f (x)**
Step 7: **X min: = x**
Step 8: **End if**

4. NEIGHBORHOODS

4.1. Neighborhood by (swap)

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

Example

Consider a sequence σ =1234,

Table 2: the neighborhood N(σ) is: N(σ)={2134,3214,4231,1324,1432,1243}.

job i	job j	Sequence
1	2	2134
	3	3214
	4	4231
2	3	1324
	4	1432
3	4	1243

4.2. Neighborhood by (insert)

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

Example

Consider a sequence σ =1234,

Table 3: the neighborhood N(σ) is:
N(σ)={2134,2314,2341,1324,1342,3124,1243,4123,1423}

Position job	1	2	3	4
1		2134	2314	2341
2	~~2134~~		1324	1342
3	3124	~~1324~~		1243
4	4123	1423	~~1243~~	

5. PROPOSED HEURISTICS

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

http://www.ijmp.jor.br v. 13, n. 1, January-march 2022
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Assign the (best) job h where ($P_i = \frac{p_j}{w_j}$) and (P_i= maxP_i) to the best machine (Selt and Zitouni, 2014); based on two principles justified by the two following propositions:

Proposition 1. In optimal scheduling, it is necessary to schedule the tasks in each availability zone of the machine according to the order SWPT.

Proof. It results directly by adjacent job exchange like used by (Smith,1956) for the corresponding zones.

Proposition 2. It is not useful to let the machine (idle) if a job can be assigned to this machine.

Notations:

We denote by:

$J = \{1, 2, ..., n\}$: The set of jobs.

p_h : Execution time of the job h .

$I = \{1, 2, ..., m\}$: The set of machines

$Z = \{1, 2, ..., \alpha\}$: Availability Zones.

$S_z^{(i)}$ ($z \in Z$): The beginning of the unavailability time of the machine $i \in I$.

$T_z^{(i)}$ $(z \in Z)$: The end of the unavailability time of the machine $i \in I$.

$C_z^{(i)}$ $(z \in Z)$: Execution time of the job $j \in J_z^{(i)}$.

w_j : the weight of the job h .

5.1. Heuristic (H1)

Initialization

J={1, 2, …, n} , $\sigma = \emptyset$, , F=0; C_i =0 ; P_i= random (1.99); W_i= random (1.10) ; z=1 .

Begin

INDEPENDENT JOURNAL OF MANAGEMENT & PRODUCTION (IJM&P)
http://www.ijmp.jor.br _v. 13, n. 1, January-march 2022_
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Sort jobs $h \in J$ in increasing order according to the criterion $\frac{p_j}{w_j}$ in L_1

While ($L_1 \neq \emptyset$) do

if ($Z_a^2 > P_0$) and ($Z_a^1 >= P_0$)

Determine the machine M such that $Z_a^1 - C_{Z_a^1}^a \leq$ min(P_{h1}, P_{h0})

Assigned the job h to the machine M ;

Compute C_j ; **F**=$f_{\sigma 1} + f_{\sigma 2}$=$f_{\sigma 1} + f_{\sigma 2}$+$C_i W_i$

Delete the job h from L_1

Else

Set Z=Z+1 **;**

End if

$\sigma = \sigma_1 \cup \sigma_2$; //obtained sequence

End

Example1

Table 4: Consider the problem P1 with the following data:

job	_1_	_2_	_3_	_4_	_5_	_6_
_Pj_	_72_	_97_	_17_	_18_	_44_	_97_
_Wj_	_7_	_9_	_1_	_10_	_3_	_7_
_Pj/Wj_	_10.2_	_10.7_	_17_	_1.8_	_14.6_	_19.5_

Results of heuristic (H_1) are: f = 3576. Execution time = 0.034sec.

Results of tabu (swapping) are: f= 3110. Execution time = 0.006 sec.

Results of tabu (insertion) are: f = 2431. Execution time = 0.008 sec.

The best results are obtained by using tabu by swapping for f = 3310.

5.2. Heuristic (H2)

Initialization

J={1, 2, …, n} , $\sigma = \emptyset$, F=0; C_i =0 ; P_i= random (1.99);W_i= random (1.10) ;

z=1 .

INDEPENDENT JOURNAL OF MANAGEMENT & PRODUCTION (IJM&P)
http://www.ijmp.jor.br v. 13, n. 1, January-march 2022
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Begin

Sort jobs $h \in J$ in increasing order according to the criterion $\frac{p_j}{w_j}$ in a list L_1

Sort jobs $h \in J$ in decreasing order according to the criterion p_j in a list L_2

While ($L_1 \neq \emptyset$) **do**

If ($Z_a^2 > P_0$) and ($Z_a^1 >= P_0$)

Determine the machine M such that $Z_a^1 - C_{z_a^1}^a \leq$ min (P_{h1}, P_{h0})

Assigned the job h_0 to the machine M

Compute C_j; $f_{\sigma 1} = f_{\sigma 1} + C_i W_i$

Delete the job h_0 from L_1 and L2

Else

If ($Z_a^2 > P_1$) and ($Z_a^1 >= P_1$)

Determine the machine M such that $Z_a^2 - C_{z_a^2}^a \leq$ min(P_{h1}, P_{h0})

Assigned the job h_1 to the machine M

Compute C_j ; **F**=$f_{\sigma 1} + f_{\sigma 2} = f_{\sigma 1} + f_{\sigma 2} + C_i W_i$

Delete the job h_1 from L_1 and L2

Else

Set Z=Z+1 **;**

End if

$\sigma = \sigma_1 \cup \sigma_2$; //obtained sequence

End

Example 2

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Table 5: Consider the problem P2 with the following data:

job	1	2	3	4	5	6
Pj	82	56	52	19	19	85
Wj	5	3	6	4	1	9
Pj/Wj	16.4	18.6	8.6	4.7	19	9.4

Results of heuristic (H_2) are: f = 3536. Execution time = 0.039 sec.

Results of tabu (swapping) are: f = 3110. Execution time = 0.005 sec.

Results of tabu (insertion) are: f = 3120. Execution time = 0.006 sec.

The best results are obtained by using tabu by swapping for f =3110.

5.3. Heuristic (H3)

Initialization

J={1, 2, …, n} , $\sigma = \emptyset$, F=0 ; $C_i =$0 ; P_i= random (1.99); W_i= random (1.10); z=1

Begin

Sort jobs $h \in J$ in increasing order according to the criterion $\frac{p_j}{w_j}$ in L_1

Sort jobs $h \in J$ increasing order according to the criterion p_j in L_2

While ($L_1 \neq \emptyset$) do

If (Z_a^2>P_0) and (Z_a^1>=P_0)

Determine the machine M such thatZ_a^1 - $C_{z_a^1}^a$≤ min (P_{h1}, P_{h0})

Assigned the job h_0 to the machine M; Compute C_j;

F =$f_{\sigma 1} + f_{\sigma 2}$=$f_{\sigma 1} + f_{\sigma 2}$+$C_i W_i$

Delete the job h_1 from L_1 and L_2

Else

If (Z_a^2>P_1) and (Z_a^1>=P_1)

Determine the machine M such that Z_a^2 - $C_{z_a^2}^a$≤ min (P_{h1}, P_{h0})

Assigned the job h_1 to the machine M

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Compute C_j ;

F $=f_{\sigma 1}+f_{\sigma 2}=f_{\sigma 1}+f_{\sigma 2}+C_i W_i$

Delete the job h_1 from L_1 and L2

Else

Set Z=Z+1 **;**

End if

$\sigma=\sigma_1 \cup \sigma_2$; // obtained sequence

End

Example 3

Table 6: Consider the problem P 3 with the following data

job	*1*	*2*	*3*	*4*	*5*	*6*
Pj	*82*	*56*	*52*	*19*	*19*	*85*
Wj	*5*	*3*	*6*	*4*	*1*	*9*
Pj/Wj	*16.4*	*18.6*	*8.6*	*4.7*	*19*	*9.4*

Results of (H_3) are: f = 3530. Execution time = 0.016sec.

Results of tabu (swapping) are: f =3091. Execution time = 0.007 sec.

Results of tabu (insertion) are: f = 3095. Execution time = 0.008 sec.

The best results are obtained by using tabu by swapping for f =3091.

6. DATA GENERATION

The heuristics were tested on problems generated with 500 jobs similar to that used in previous studies : (M'Hallah & Bulfin, 2005; Lee, 1996, 1997; Schmidt, 2000) for each task j an integer processing time Pj was randomly generated in the interval (1.99), with a weight randomly wj chosen in the interval (1.10).

7. DISCUSSION OF RESULTTS

We have chosen MATLAB as our programming and testing tool. In this part we illustrate a comparison between heuristics (H1), (H2), (H3) and metaheuristic TS, from our testing, we noticed the following: If the number of jobs n is less than 150, then the proposed heuristics present good results. If the number of jobs n is between 150 and 250, the Tabu method by Swapping gives better results (Figures 1, 2 and 3). If the number of jobs exceeds

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

250, in this case, the tabu method by swapping whose complexity is o(n3) becomes practically useless (results of tables 3,4 and 5).

Tables 7, 8 and 9 below presents: BC: The best costs, AC: Average costs, AT: Average time.

Table 7: Heuristic (H1) cost amelioration based on (TS).

JOBS	Results of heuristic (H_1)	AT (sec)	(TS)Swap		(TS)Insert		BC
			AC	AT **(sec)**	AC	AT **(sec)**	
N=30	**38432**	0.013	**29562**	0.85	**30659**	1.22	**29562**
	48056	0.012	37335	1.02	**37258**	1.95	**37238**
	34420	0.01	**26967**	0.93	**27265**	159	**26967**
N=50	**113123**	0.04	**96256**	5.50	**97945**	8.56	**96256**
	105562	0.08	**93625**	6.60	**93959**	9.62	**93625**
	102225	0.07	94215	5.30	**93165**	8.23	**93165**
N=150	**931265**	0.17	**869856**	52.35	**911025**	60.10	**869856**
	926921	0.15	912694	60.25	**908223**	62.68	**908223**
	882230	0.19	**858541**	58.12	**859624**	63.31	**858541**
N=250	**2655846**	0.26	**2630354**	135.53	**2641873**	137.36	**2630354**
	2559125	0.21	2549623	165.36	**2545280**	164.23	**2545280**
	2478415	0.22	**2459225**	123.68	**2465968**	124.65	**2459225**
N=350	**4965280**	0.26	**4962171**	265.25	**4964382**	276.95	**4962171**
	4771183	0.31	**4767183**	296.32	**4768245**	300.34	**4767183**
	4896954	0.24	4889864	240.36	**4887262**	268.21	**4887262**
N=500	**9213434**	0.55	**9107596**	436.6	**9110652**	435.24	**9107596**
	9126543	0.6	**9122261**	370.65	**9123621**	381.23	**9122261**
	9506951	0.7	9499251	395.12	**9498926**	397.15	**9498926**

Table 8: Heuristic (H2) cost amelioration based on (TS).

JOBS	Results of heuristic (H_2)	AT (sec)	(TS)Swap		(TS) Insert		BC
			AC	AT (sec)	AT	AC (sec)	
N=30	**40586**	0.012	**31657**	0.8	**31057**	1.18	**31057**
	38213	0.013	**29564**	0.93	**30526**	1.78	**29564**
	37592	0.015	**28456**	1.01	**28915**	1.64	**28456**
N=50	**100172**	0.022	**89743**	5.9	**91254**	8.36	**89743**
	111228	0.016	**98625**	6.2	**99147**	9.82	**98625**
	99273	0.018	**89745**	5.75	**91450**	8.11	**89745**
N=150	**851935**	0.041	**803856**	53.47	**809256**	63.23	**803856**
	889098	0.052	**832159**	62.71	**820541**	61.54	**820541**
	867296	0.039	**813753**	58.99	**819369**	60.73	**813753**
N=250	**2324111**	0.077	**2299840**	131.66	**2293647**	140.73	**2293647**
	2411968	0.063	**2365486**	164.32	**2394935**	174.66	**2365486**
	2395659	0.070	**2360258**	129.38	**2373281**	129.81	**2360258**
N=350	**4569054**	0.12	**4528346**	260.54	**4542563**	266.49	**4528346**
	4656261	0.129	**4597750**	285.97	**4616542**	298.67	**4597750**
	4448628	0.122	**4425698**	243.83	**4416581**	270.36	**4416581**
N=500	**9031909**	0.35	**9016549**	446.77	**9029512**	431.94	**9016549**
	9225172	0.33	**9210975**	365.16	**9209964**	388.58	**9078964**
	9340531	0.34	**9318620**	399.52	**9319753**	421.42	**9168620**

http://www.ijmp.jor.br v. 13, n. 1, January-march 2022
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

Table 9: Heuristic (H3) cost amelioration based on (TS).

JOBS	Results of heuristic (H_3)	AT (sec)	(TS)Swap AC	(TS)Swap AT (sec)	(TS)Insert AC	(TS)Insert AT (sec)	BC
N=30	35910	.014	**27365**	0.77	27801	0.98	**27365**
	35708	.013	**28054**	0.98	28456	1.50	**28054**
	36114	.011	28282	1.02	**28067**	1.39	**28067**
N=50	98285	.016	**90170**	6.2	91843	7.90	**90170**
	104207	.021	96408	6.3	**95290**	9.58	**95290**
	102195	0.02	**91819**	5.9	94014	8.45	**91819**
N=150	936879	0.06	**867658**	51.79	880170	63.77	**867658**
	901542	0.063	**822964**	61.3	839753	65.20	**822964**
	910925	.067	861490	59.48	**852135**	66.95	**852135**
N=250	2530927	0.08	**2489321**	37.32	2500325	142.78	**2490321**
	2300753	0.09	2270845	59.91	**2259658**	174.66	**2232658**
	2276966	0.078	**2245547**	132.45	2249528	131.59	**2236547**
N=350	4628100	0.12	**4590596**	255.32	4598212	262.58	**4678596**
	4523330	0.2	4491627	291.75	**4489365**	300.12	**4482365**
	4559716	0.21	**4530129**	252.01	4533156	277.81	**4518029**
N=500	9363249	0.3	9339824	420.44	**9321598**	445.43	**9236598**
	9203920	0.25	9195893	383.61	**9172589**	398.70	**9032589**
	9023792	0.28	**9001569**	400.59	9004796	427.91	**9001569**

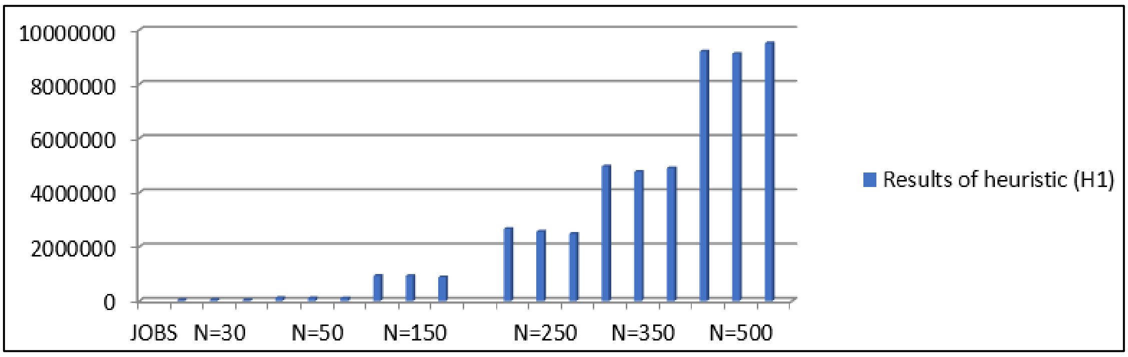

Figure1: Histogram of heuristic (H1) cost amelioration based on tabu search for different N values.

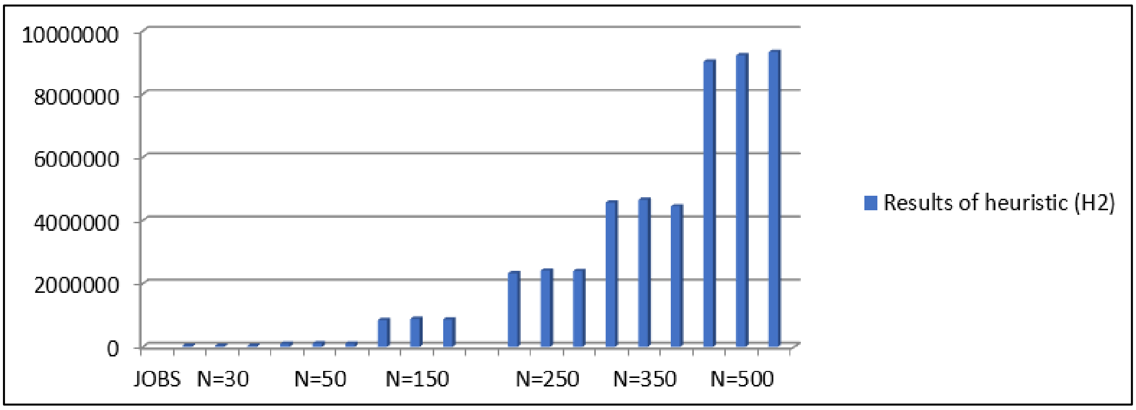

Figure2: Histogram of heuristic (H2) cost amelioration based on tabu search for different N values.

ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

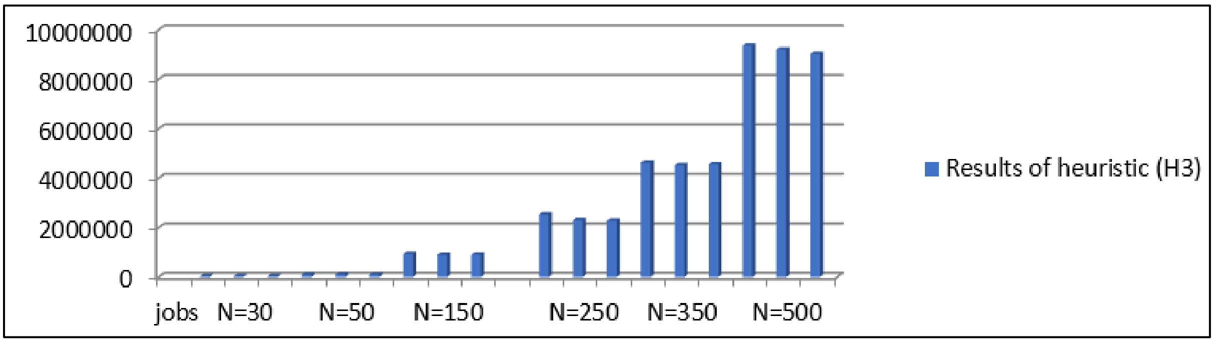

Figure3: Histogram of heuristic (H3) cost amelioration based on tabu search for different N values.

8. CONCLUSION

In this work, we propose a novel approach for scheduling problems on two parallel identical machines).

The developed approach uses a diversification technique based on search restarting from the point of the solution that was chosen among the earlier best unmaintained found solutions. According to the curried out tests, it can be concluded that the proposed heuristics ensure good results (polynomial complexity o(n3)).

It must be noted that the heuristic (H2) and the neighborhood by (SWAP) present the best costs with an acceptable execution time.

REFERENCES

Adamu M. O., & dewunmi, A. (2012). Metaheuristics for scheduling on the parallel machine to minimize the weighted number of early and tardy jobs. **Int. J. Phys. Sci.**,7(10), 1641-1652.

Adamu M. O., & Adewunmi. A. (2013). Comparative study of metaheuristics 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.

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

http://www.ijmp.jor.br v. 13, n. 1, January-march 2022
ISSN: 2236-269X
DOI: 10.14807/ijmp.v13i1.1533

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.

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

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

Zitouni, R., & Selt, O. (2016). Metaheuristics to solve tasks scheduling problem in parallel identical machines with unavailability periods, **RAIR. O Res**, 50(1), 90-97

Smith, W. E. (1956). Various optimizes for single-stage production, **Nava Res. Logistc**, 3, 59- 66.

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.