AUSTRALIAN JOURNAL OF BASIC AND APPLIED SCIENCES

ISSN:1991-8178 EISSN: 2309-8414
Journal home page: www.ajbasweb.com

Using Local Search methods to solve Unrelated Parallel Machine scheduling problem

[1]Hanan A. Chachan and [2]Marwa B. kadhem

[1,2]*Mathematics Department, College of science, AL_Mustansiriah University*

Address For Correspondence:
Hanan A. Chachan, Mathematics Department, College of science, AL_Mustansiriah University
E-mil: Hanan_altaai@yahoo.com

ARTICLE INFO

Article history:
Received 19 September 2016
Accepted 10 December 2016
Published 31 December 2016

Keywords:
Scheduling; unrelated parallel machines; total completion time; total weighted tardiness; Local search methods ; Descent method (DM); Simulating annealing (SA).

ABSTRACT

The problem of scheduling of unrelated parallel machines is considered. In this environment, a set of n jobs has to be scheduled on m unrelated parallel machines. Each job is available for processing at time zero and each machine can process at most one job at a time and a job can be processed by at most one machine at a time. A case study is considered to schedule jobs in a cutting workshop to minimize the sum of total completion time and total weighted tardiness. Six heuristics are proposed and two local search methods are Descent method(DM) and Simulated annealing (SA). That can find nearly optimal solutions with in a reasonable amount of computation time.The performance of exact and approximation algorithms above are tested on large class of test problems. From our computational results, we found that the approximation algorithms solve problem up to (7500) jobs and (50) machines in reasonable amount of time.

INTRODUCTION

Parallel machine scheduling is a typical scheduling problem which is important in both theoretical and practical aspects. From the theoretical viewpoint, it generalizes the single machine scheduling problem and is a special case of the flexible flow shop. From the practical viewpoint, it is important because the occurrence of a bank of machines in parallel is very common in practice.Though there is extensive literature on parallel machine scheduling problems, most of it is limited to situations in which the processing times or speed rates are the same across all machines. When machines are not identical to one another and cannot be completely correlated by simple rate adjustments, they are said to be unrelated parallel machines. Based on the complex hierarchy of deterministic scheduling (Brucker, P., 2007; Chen, J.F. 2009; Crauwels, H., 1998; Glass, C.A., *et al*., 1994; Kirkpatrick *et al*., 1983; Kim, D., *et al*., 2002; Lin, Y.K., *et al*., 2012; Lenstra, J.K., A.H.G. Rinnoykan, 1978; Lin, Y.K., *et al*., 2011; Ogbu, F.A. and D.K. Smith, 1990; Pinedo, M.L. 2012) among the traditional classification of parallel machine environments, unrelated parallel machines are some of the most difficult problems to solve.

We consider a problem in which n jobs are to be scheduled without preemption on munrelated parallel machines. Each job is to be assigned to one of the machines and, at any time, each machine can process at most one job. Job j (j = 1,. . . , n) becomes available for processing at time zero and requires a positive integer processing time p_{ij} if it is assigned to machine j $(j = 1, \dots, m)$.

Research on local search methods in scheduling is quite extensive, especially for the flow shop (Ogbu, F.A. and D.K. Smith, 1990 ; Pinedo, M.L. 2012; Pfund, M., *et al*., 2004; Piersma, N. and W. Van Dijk, 1996; vanLaarhoven, P.J.M., *et al*., 1992; Widmer, M. and A. Hertz, 1989) and the job shop (Aarts, E.H.L. *et al*., 1991; Brucker, P., 2007; Chen, J.F. 2009; Crauwels, H., 1998; Glass, C.A., *et al*., 1994; Kirkpatrick *et al*., 1983;

To Cite This Article: Hanan A. Chachan and Marwa B. kadhem., Using Local Search methods to solve Unrelated Parallel Machine scheduling problem. ***Aust. J. Basic & Appl. Sci.,*** *10(18): 267-275, 2016*

Kim, D., *et al*., 2002; Lin, Y.K., *et al*., 2012; Lenstra, J.K., A.H.G. Rinnoykan, 1978; Lin, Y.K., *et al*., 2011; Ogbu, F.A. and D.K. Smith, 1990). However, applications to parallel machine scheduling are scarce. Furthermore, there are relatively few computational studies that compare different local search methods on the same scheduling problems. In this paper, we help to fill this gap in the literature by giving an extensive computational comparison of local search methods for unrelated parallel machine scheduling.

Most unrelatedparallelmachine scheduling problemsare NP-Hard. Pfundetal. (Pfund, M., *et al*., 2004) Presented asurveyofalgorithms for single- and multi-objectiveunrelatedparallelmachine deterministic schedulingproblems, butindicatedthatunrelatedparallel machine problemsremainrelativelyunstudied. Inparticular, they noted thattherearefewsolutionapproachestominimizedue- date-related criteria.

(Glass *et al* 1994) propose Descent method, tabu search and Genetic method for problem $R_m \| C_{max}$.Piersma and Van Dijk (1996) propose a local search heuristic for the problem with efficient neighborhood search,Kim *et al*. (2002) proposed a simulated annealing method with the objective to minimize the total tardiness, Chen (2009) also studied the UPMSP with sequence and machine-dependent setup times and due date constraints ,Lin *et al*. (2011) compares the performance of several existing heuristics and a GA-based algorithm for the UPMSP with respect to three regular performance measures, including makespan, total weighted completion time and total weighted tardiness. Their computational results demonstrated that the GA-based algorithm outperformed other existing heuristics for each of the three performance measures, Lin *et al* (2012) suggest Ant colony for $R_m \| \sum_{i=1}^{n} W_i T_i$.

The remaining sections of the paper are organized as follows. In section 2 presents the problem formulated, our heuristic methodsare presented in section 3,in section 4 and presents local search methods the computational experiments in section 5 finally conclusion are drawn in section 6

2. Problem Descriptions and Integer Programming:

In this section, we give some notations and present integer programing which use characterization to solve for $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$ problem.

The notations as follows:-

N set of jobs.
M set of the machine.
m number of the machine (m = $|M|$) .
n number of the jobs (n = $|N|$) .
w_i Weighted of job i.
d_i Due date of job i.
p_{ik} Processing time of job i on machine k .
c_{ik} Completion time of job i assigned to machine k.

2.1 Integer Programming:

The $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$ problem can be formulated as the following integer programming problem (P).

$Z_p = min \sum_{i=1}^{n} \sum_{k=1}^{m} \sum_{t=1}^{n} C_{ik}^t X_{ik}^t + W_i T_{ik}^t X_{ik}^t$

Subject to

$$\sum_{k=1}^{m} \sum_{t=1}^{n} X_{ik}^t = 1 \quad \left.\begin{matrix} i,t = 1,2,\dots,n \\ k = 1,2,\dots,m \end{matrix}\right\} \quad (2)$$

$$\sum_{i=1}^{n} X_{ik}^t \leq 1 \quad \left.\begin{matrix} i,t = 1,2,\dots,n \\ k = 1,2,\dots,m \end{matrix}\right\} \quad (3)$$

$$C_{ik}^t = \sum_{l=1}^{n} \sum_{s=1}^{t-1} p_{lk} X_{ik}^s \quad \left.\begin{matrix} i,t = 1,2,\dots,n \\ k = 1,2,\dots,m \end{matrix}\right\} \quad (4) \quad (p)$$

$$T_{ik}^t = \max(0, C_{ik}^t - d_i) \quad \left.\begin{matrix} i,t = 1,2,\dots,n \\ k = 1,2,\dots,m \end{matrix}\right\} \quad (5)$$

$$X_{ik}^t = 0 \text{ or } 1 \quad \left.\begin{matrix} i,t = 1,2,\dots,n \\ k = 1,2,\dots,m \end{matrix}\right\} \quad (6)$$

Where C_{ik}^t denoted the completion time of job i when it is scheduled in the t^{th} position on machine k and

$$X_{ik}^t = \begin{cases} 1 & if\ job\ i\ is\ scheduled\ in\ t^{th} position\ on\ machine\ k \\ 0 & o.w \end{cases}$$

Constraint (2) state that each job is assigned to exactly one position on machine, whereas constraint (3) guaranties the assignment at most one job to each position on each machine. Constraint (4) given the completion time of job i when it is scheduled in the t^{th} position on machine k. Constraint (5) is definition of tardiness. Finally constraint (6) is a decision variable if job i scheduled on machine k in position t, then $X_{ik}^t = 1$ otherwise 0.

3. Heuristic Methods:

For the problem (P) we proposed six heuristic methods

Heuristic 1:

The first heuristic (H_1) with value (UB_1) is simply obtained as follows:

Step (1): For each job i find $P_{j^*}^{`}= \min_{1\le j\le m} P_{ij}$.

Step (2): Order the jobs on each machine j* according to non-decreasing of $P_{j^*}^{`}$.

Step (3): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Heuristic 2:

The second heuristic method $H_2$with value (UB_2)is obtained as follows:

Step (1): Let S_j^* average of the time of the jobs for each machine j, j=1,…,m.

Step (2): Calculate I_{ij}^* such that $I_{ij}^*=p_{ij}^*/ S_j^*$for each job i ,i=1,…,n.

Step (3): Find the unscheduled job i^* with the smallest I_{ij}^{**}

i:e $I_{ij}^{**} = min_{1\le j^*\le m} I_{ij}^*$.

Step (4): Let g`be the set of unscheduled jobs for $I_{ij}^{**}$ then order the jobs in g` according to non-decreasing on this machine j*.

Step (5): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Heuristic 3:

The third heuristic method (H_3) with value (UB_3)is simply obtained as follows:

Step (1):$\forall i = 1, \dots, n$ find $P_{j^*}^{`}= \min_{1\le j\le m} P_{ij}$.

Step (2): Order the jobs on each machine j* according to non-decreasing of $P_{j^*}^{`}$

Step* (3):calculate completion time for scheduled jobs*

$i{:}e\ C_i \ \ \forall\, i = 1, \dots, n.$

Step (4): take $\min_{1\le j\le m} C_j \ \forall\, j = 1, \dots, m$

Step (5): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Heuristic 4:

The fourth heuristic method (H_4) with value (UB_4)is simply obtained as follows:

Step (1): Let S be the set of unscheduled jobs and K denote the number of job schedule on machine $j, j = 1, \dots, m$. In the first time k=1 ∀ j=1,…,m .

Step (2): $\forall\, i, i = 1, \dots, n$ find $P_{j^*}^{`}= \min_{1\le j\le m} P_{ij}$.

***Step* (3)**: Find job $\ell_{j^*}^{`} = maxP_{j^*}^{`}$, scheduling this job on machine j^* to last position, $S = S - \{\ell_{j^*}^{`}\}$.

Step (4): ∀ machine j^*, the value of processing time for unscheduled job will be $kp_{ij^*}/p_{i^*j^*}$.when $p_{i^*j^*}$ the last job scheduled on machine j^*.

***Step* (5)**: Repeat step 1_4 until all set $S = \emptyset$.

Step (6): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Heuristic 5:

The five heuristic methods (H_5) with value (UB_5)is simply obtained as follows:

Step (1): Let S be the set of unscheduled jobs and K denote the number of job schedule on machine $j, j = 1, \dots, m$. In the first time k=1 ∀ j=1,…,m .

Step (2): $\forall\, i, i = 1, \dots, n$ find $P_{j^*}^{`} = \min_{1 \le j \le m} P_{ij}$.

Step (3): Find job $\ell_{j^*}^{`} = max P_{j^*}^{`}$, scheduling this job on machine j^* to last position, $S = S - \{\ell_{j^*}^{`}\}$.

Step (4): ∀machine j^*,the value of processing time for unscheduled job will be kp_{ij^*}.

Step (5): Repeat step 1_4 until all set $S = \emptyset$.

Step (6): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Heuristic 6:

The Six heuristic methods (H_6)with value (UB_6) is simply obtained as follows:

Step (1): Let S be the set of unscheduled jobs and K denote the number of job schedule on machine $j, j = 1, \dots, m$. In the first time k=1 ∀ j=1,…,m .

Step (2): $\forall\, i, i = 1, \dots, n$ find $P_{j^*}^{`} = \min_{1 \le j \le m} P_{ij}$.

Step (3): Find job $\ell_{j^*}^{`} = max P_{j^*}^{`}$, scheduling this job on machine j^* to last position, $S = S - \{\ell_{j^*}^{`}\}$.

Step (4): ∀machine j^*,the value of processing time for unscheduled job will be

$kp_{ij^*}/\overline{p_j^*}$.when $\overline{p_j^*}$represent the average processing time of machine j^* i:e$\overline{p_j^*} = \sum_{i=1}^{n} p_{ij}^* /$ n.

Step (5): Repeat step 1_4 until all set $S = \emptyset$.

Step (6): Evaluate the $\sum_{i=1}^{n} C_i + \sum_{i=1}^{n} W_i T_i$ with respect to $R_m \| \sum_{i=1}^{n} (C_i + W_i T_i)$.

Definition [1]:

An instance of combinatorial optimization problem is a pair (S,f) where the solution set S is the set of all feasible solutions and the cost function f is a mapping f:S→ R. the problem is to find a globally optimal (minimal) solution i.e. an S^*є S such that f(S^*)≤f(S) for all sєS.

Definition [10]:

A neighborhood function $N^*$is a mapping N^*:S→P(S) which specifies for each sєS a subset $N^*(s)$of S neighbors of s.

4. Local Search methods:

Local search is family of method that interactively search through the set of solutions starting from initial solutions a local search procedure moves from one feasible solutions to a neighboring solution until some stopping criteria are met the choice of suitable neighborhood function has an important influence on the performance of local search. Research a local search in scheduling is quite extensive, but application to unrelated parallel machine scarce there are a few computation studies that compare different local search methods on the same scheduling problem. We develop a local search method here where two operations are used to generate local search neighborhoods these operations are the:

A. Move operations: Reassigning one job from a machine to another machine.

B. Swap operation: Swap one job from a machine with one job from another machine.

Local search methods share the following features [5]

- Initialization: an feasible solution s is generated randomly or by applying a heuristic method and declared as the current solution the objective function value of the current solution is completed.
- Neighborhood generation: a "move" is made through the solution space S from neighbor to neighbor to select a neighbors of S

- Acceptance test: each local search method has its own acceptance test to decide whether is replace S as the current solution.
- Termination criteria: the method is repeated until some termination criteria are satisfied. The output will be the best solution generated.

In this section two local search heuristic are implemented on the problem of scheduling n jobs on unrelated parallel machines to minimize the $\sum_{i=1}^{n}(C_i+W_iT_i)$.

For the representation of solution the natural representation will be used. For each local search method a set of parameters to the problem are described.

4.1 Descent method (DM)[6]:

The descent method (DM) is the simple kind of local search methods but it can be trapped in a local optimal. (DM) can be executed for problem$R_m \| \sum_{i=1}^{n}(C_i+W_iT_i)$as allowed in figure (1).

Descent method
Step (1): Initialization
In this step a feasible solution $\delta = (\delta_1, \dots, \delta_n)$ obtained from the best of the upper bound $UB = (UB_1, \dots, UB_6)$ heuristic is chosen to be the initial current solution for the (DM)
Step (2): Neighborhood Generation
For scheduling problem that involve permutations there are two generating mechanisms to obtain neighborhoods. In this work we use variable neighborhood search (VNS) which is a simple change of neighborhood within the search. In these paper neighborhoods these operations are the A&B. In order to improve the sequence δ the traveling between different neighborhoods gives a new sequence δ^* that will be obtained with its objective function value $f(\delta^*)$.
Step (3): Evaluation
a: if the improvement is made (i:e $f(\delta^*) < f(\delta)$) then δ^* is accepted as the current solution and set $\delta = \delta^*$,then go to step (2)
b: otherwise (i:e$f(\delta^*) \geq f(\delta)$) , δ is retained as the current solution and go to step (2).
Step (4): Termination
This algorithm is terminated after (10000) iteration at near optimal solution

Fig. 1: structure of descent method

4.2 Simulated annealing (SA):

Simulated annealing is an optimization met heuristic based on the cooling properties of solid [7].The structure of simulated annealing method is given in figure (2):

Simulated Annealing method
Step (1):Initialization
The same technique describe in step (1) of (DM) is used to obtained initial current solution $\delta = (\delta_1, \dots, \delta_n)$ and compute its objective function value$f(\delta)$.
Step (2): Neighborhood Generation
The neighborhood δ^* of the current solution δ is generated by using A&B procedure which described in step (2) of (4.1) and compute its objective value $f(\delta^*)$
Step (3): Acceptance Test
In this step the different Δ between the current solution value $f(\delta)$ and the new value $f(\delta^*)$ is calculated, where $\Delta = f(\delta^*) - f(\delta)$
A: if $\Delta \leq 0$ then $\delta^*$is accepted as the new current solution and set $\delta = \delta^*$.
B: if $\Delta > 0$ then δ^* is accepted with probability $P(\Delta, t) = \exp(-\Delta/t)$ which is the probability of accepting a deter orating move, where t is parameter is known as the temperature in this step a feasible solution $\delta = (\delta_1, \dots, \delta_n)$ obtained from the best of the $(H_1, \dots, H_6)$ heuristic is chosen to be initial current solution for the (SA).
Step (4): Updating the Temperature The term premature is set to relatively high value and gradually decreases slowly as the progress. The temperature at each each iteration (m) is derived by following recursive formula $t_{i+1} = t_i/(1+Bt_i)$ where
$B_i = (t_i - 1)/(m * t_i)$;$t_1 = 100$; m = 600 ; i = 1,2,…,m
Step (5): Termination Test
The algorithm is terminated after (10000) iteration at a feasible solution.

Fig. 2: structure of simulated annealing

5. Computational results of local search algorithms and comparison:-

The local search methods have tested by programming using Matlab (2016a) and running on program Inter (R) core (TM) i5-4200M CPU@, 2.50GHz and RAM 4.00GB the performance of proposed algorithm is reported on randomly generated problem, processing times are randomly chosen from U[1,10] the value of w_i for each i is chosen randomly from U[1,10],due date are generated from U[$\bar{P}$(1-TF-R$DD/2$),$\bar{P}$(1-TF+R$DD/2$)] where $\bar{P} = \sum_{i=1}^{n}\sum_{j=1}^{m} P_{ij}/m^2$, For both TF and RDD are the tardiness factor ,and relative rang of due date respectively the values 0.2 ,0.4, 0.6,0.8 and 1.0.For each selected value of n where n is the number of jobs one problem are generated for each of the ten values of parameters producing 10 problems for each value of n.

Tables (1),(2),…,(10) show for each algorithm, the value of objective function for problem(P) and how many times it can catch the optimal value for each value of "problem size and "number of machine. In additional, describes the deviation of local search methods from optimal solution.

Table 1: comparison results between (DM) and (SA) method form = 10, n =1000

Job	Ex	DM	Time	SA	Time	Best
1000	1	272445	5.7123	288312	5.7791	272445
	2	278085	5.584	275328	5.5667	275328
	3	274674	5.6111	269847	5.5346	269847
	4	268022	5.5876	272190	5.5067	268022
	5	281314	5.6038	279203	5.4874	279203
	6	265885	5.5821	285012	5.492	265885
	7	277835	5.5973	276029	5.4996	276029
	8	281386	5.5884	268101	5.5305	268101
	9	282781	5.595	285417	5.4948	282781
	10	279091	5.5818	276733	5.5041	276733
No. Best		4		6		
Av.Time		5.6		5.54		

Table 2: comparison results between (DM) and (SA) method for m = 10, n =7500

Job	Ex	DM	Time	SA	Time	Best
7500	1	15550564	42.0897	15461748	41.2451	15461748
	2	15643280	42.0918	15715993	40.8135	15643280
	3	15485982	41.9349	15514805	40.7314	15485982
	4	15564023	42.0217	15546860	40.6716	15546860
	5	15394103	41.8396	15388214	40.7144	15388214
	6	15523713	41.8315	15565081	40.7388	15523713
	7	15363037	41.9848	15367899	42.1582	15363037
	8	15225589	41.9291	15545402	41.8296	15225589
	9	15452421	42.2915	15556419	44.2973	15452421
	10	15742252	41.9491	15634836	41.0585	15634836
No. Best		6		4		
Av.Time		42		41.43		

Table 3: comparison results between (DM) and (SA) method for m = 20, n =1000

Job	Ex	DM	Time	SA	Time	Best
1000	1	148883	7.8986	146355	7.9239	146355
	2	137434	7.7339	142391	7.7661	137434
	3	144312	7.7458	142649	7.7689	142649
	4	136991	7.7576	136976	7.7258	136976
	5	139963	7.7347	137812	7.8932	137812
	6	141755	7.7342	145186	7.7641	141755
	7	137090	7.7768	143845	7.7427	137090
	8	138677	7.7348	133494	7.7848	133494
	9	138452	7.7605	142332	7.7544	138452
	10	143113	7.7422	144355	7.7525	143113
No. Best		5		5		
Av.Time		7.76		7.79		

Table 4: comparison results between (DM) and (SA) method for m = 20, n =7500

Job	Ex	DM	Time	SA	Time	Best
7500	1	7738931	59.9716	7712099	56.4588	7712099
	2	7684547	58.8029	7752423	57.6378	7684547
	3	7810176	59.2201	7741914	56.1345	7741914
	4	7765154	58.5094	7749729	56.0308	7749729
	5	7736131	61.7339	7729332	56.1459	7729332
	6	7767752	59.6502	7750994	56.2925	7750994
	7	7719445	59.2151	7768994	56.0397	7719445
	8	7780239	58.337	7716520	56.2839	7716520
	9	7763600	57.887	7831901	56.2172	7763600
	10	7823652	58.7448	7750552	56.5689	7750552

No. Best		3	7
Av.Time		59.21	56.38

Table 5: comparison results between (DM) and (SA) method for m = 30, n =1000

Job	Ex	DM	Time	SA	Time	Best
1000	1	95492	9.73	99709	9.7436	95492
	2	96602	9.7957	96094	9.5955	96094
	3	97818	9.6548	98863	9.5004	97818
	4	96580	9.6267	95289	12.3084	95289
	5	91790	9.5877	98043	12.2832	91790
	6	97140	9.5448	94871	11.7716	94871
	7	94846	9.7014	95645	11.7552	94846
	8	96037	9.68	93796	11.7525	93796
	9	93528	9.5379	95136	11.7776	93528
	10	98225	9.4837	94039	11.7428	94039
No. Best		5		5		
Av.Time		9.63		11.22		

Table 6: comparison results between (DM) and (SA) method for m = 30, n =7500

Job	Ex	DM	Time	SA	Time	Best
7500	1	5174849	77.1042	5196411	72.4923	5174849
	2	5197308	77.882	5232941	71.8298	5197308
	3	5176390	78.9103	5098536	72.0961	5098536
	4	5154263	74.6657	5166928	71.7328	5154263
	5	5166232	75.313	5184596	71.7039	5166232
	6	5221470	74.5944	5156261	71.9897	5156261
	7	5163626	74.0082	5121290	71.6486	5121290
	8	5169930	73.7421	5256180	71.9053	5169930
	9	5173253	73.9946	5196290	71.7319	5173253
	10	5225706	73.8677	5181825	72.419	5181825
No. Best		6		4		
Av.Time		75.41		71.95		

Table 7: comparison results between (DM) and (SA) method for m = 40, n =1000

Job	Ex	DM	Time	SA	Time	Best
1000	1	73243	11.8871	72593	11.6364	72593
	2	70154	11.6559	73178	11.5476	70154
	3	71392	11.6355	72859	11.5231	71392
	4	70302	11.6576	69584	11.7127	69584
	5	67553	11.6173	71819	11.5487	67553
	6	72609	11.6001	71978	11.5306	71978
	7	70449	11.6175	72628	11.4982	70449
	8	72396	11.5981	68493	11.5377	68493
	9	73858	11.6159	74983	11.5162	73858
	10	72216	11.599	70296	11.6009	70296
No. Best		5		5		
Av.Time		11.65		11.57		

Table 8: comparison results between (DM) and (SA) method for m = 40, n =7500

Job	Ex	DM	Time	SA	Time	Best
7500	1	3927164	91.0687	3912425	93.0072	3912425
	2	3924438	90.8095	3860352	90.8336	3860352
	3	3900301	91.1752	3890893	90.6454	3890893
	4	3830719	91.0854	3904432	90.6658	3830719
	5	3866127	90.5757	3897022	90.4483	3866127
	6	3877149	90.367	3871673	90.2969	3871673
	7	3975398	90.8807	3898293	90.4269	3898293
	8	3806044	91.8637	3903813	90.2649	3806044
	9	3943596	90.3142	3922551	90.3825	3922551
	10	3893858	90.939	3900031	90.2177	3893858
No. Best		4		6		
Av.Time		90.91		90.72		

Table 9: comparison results between (DM) and (SA) method for m = 50, n =1000

Job	Ex	DM	Time	SA	Time	Best
1000	1	57870	13.7745	57661	13.6893	57661
	2	58881	13.5885	58091	13.5208	58091
	3	57290	13.5358	57783	13.5275	57290
	4	56986	13.5778	59305	13.4817	56986
	5	57762	13.5366	56638	13.5269	56638
	6	58658	13.5482	60395	13.5117	58658

	7	58946	13.5229	59816	13.5673	58946
	8	58718	13.5812	55704	13.6078	55704
	9	59745	13.5601	58012	13.5396	58012
	10	58226	13.5673	59072	13.4841	58226
No. Best		5		5		
Av.Time		13.58		13.55		

Table 10: comparison results between (DM) and (SA) method for m = 50, n =7500

Job	Ex	DM	Time	SA	Time	Best
7500	1	3147627	120.2476	3089006	105.6161	3089006
	2	3136074	117.5486	3141402	119.1226	3136074
	3	3123000	115.9546	3131731	127.4727	3123000
	4	3113981	115.3837	3124304	127.6691	3113981
	5	3124369	116.325	3162713	127.5214	3124369
	6	3136334	117.1933	3178395	128.2185	3136334
	7	3137501	114.5286	3118722	127.925	3118722
	8	3124852	113.8661	3107665	128.1936	3107665
	9	3085691	109.9127	3123995	129.1045	3085691
	10	3115426	109.8174	3132527	127.2254	3115426
No. Best		7		3		
Av.Time		115.08		124.81		

Note of tables (1),(2),…, (10)
Ex: the number of test problem.
Job: number of jobs.
Time: the completion time in second.
DM: the descent method.
SA: the simulated annealing method.
Best: the Best value.
No. Best: the number of examples that catch the best value.
Av.Time: the average of time in second for getting solution.

Conclusions:

This paper, the problem of scheduling is given set of independent jobs on a set of unrelated parallel machine to minimize the sum total completion time and total weighted tardiness which is NP-hard and examined. We Us local search methods descent method and simulated annealing method to get near optimal solutions and we present six heuristics methods for obtaining initial solutions for which use n local search methods .we report on the results of extensive computations test of local search methods.

REFERENCES

Aarts, E.H.L. and J.K. Lenstra (eds), 1997. "*Local search in combinatorial optimization*",John Wiley and Sons, Chichester.

Aarts, E.H.L. van P.J.M. Laarhoven and N.L.J Ulder, 1991. "*Local-search-based algorithms for job shopScheduling",CQM-Note 106,* Centre for Quantitative Methods, Nederlandse Philips Bedrijven B.V., Eindhoven.

Brucker, P., 2007. "*Scheduling Algorithm"* ,(5th ed.), Springer Verlag, Berlin.

Chen, J.F. 2009.*"Scheduling on Unrelated Parallel Machines with Sequence- and Machine-dependent Setup times and Due-date Constraints"*,International Journal of Advanced Manufacturing Technology, 44: 1204-1212.

Crauwels, H., 1998. "*Acomparative study of local search methods for one Machine sequencing problems"* Ph.D. thesis, Katholieke university Leuren, Belgium.

Glass, C.A., C.N. Potts and P. Shade, 1994. *"unrelated Parallel MachineScheduling Using Local Search"*, 20(2): 41-52.

Kirkpatrick and J. Gelatt and d.D. Vecchi, 1983. *"Optimization by simulated annealing".*Science, 220: 671-680.

Kim, D., K. Kim, W. Jang and F. Chen, 2002. *"Unrelated parallel machine Scheduling with setup times using simulated annealing"*, Robotics and Computer Integrated Manufacturing, 18: 223-231.

Lin, Y.K., H.T. Hsieh and F.Y. Hsieh, 2012. " *Unrelated Parallel Machines Scheduling Problem Using an Ant Colony Optimization Approach"*,World Academy of Science, Engineering and Technology, 6: 08-28.

Lenstra, J.K., A.H.G. Rinnoykan, 1978.*"Complexity of scheduling under Precedence constraints"* Operation Research.

Lin, Y.K., M.E. Pfund and J.W. Fowler, 2011. *"Heuristics for Minimizing Regular Performance Measures in Unrelated Parallel Machine Scheduling Problems"*, Computers and Operations Research, 38: 901-916.

Ogbu, F.A. and D.K. Smith, 1990. *"The application of the simulated annealing algorithm to the solution ofthen|m|C_{max}flowshop problem*", Computers & Operations Research, 17: 243-253.

Pinedo, M.L. 2012.*" Scheduling Theory, Algorithms, and Systems"*,(4th ed.),Springer Science and Business Media, LLC.

Pfund, M., J.W. Fowler, J.N.D. Gupta, 2004. "*A survey of algorithm for single and multi- objective unrelatedparallel-machine deterministic scheduling problems*", Journal of the Chinese Institute of Industrial Engineers, 21: 230-41.

Piersma, N. and W. Van Dijk, 1996. *" A local search heuristic for unrelated parallel machine scheduling with efficient neighborhood search*", Mathematical and Computer Modeling, (24): 9-11.

vanLaarhoven, P.J.M., E.H.L. Aarts and J.K. Lenstra, 1992. "*Job shop scheduling by simulated annealing*", Operations Research, 40: 113-125.

Widmer, M. and A. Hertz, 1989. *" A new heuristic method for the flow shop sequencing problem"*, European Journal of Operational Research, 41: 186-193.