University of Benghazi
Faculty of Education Al marj
Global Libyan Journal

جامعة بنغازي
كلية التربية المرج
المجلة الليبية العالمية

العدد السابع – مايو 2016

Solve job shop scheduling problem with sequence dependent setup times using a simulated annealing algorithm with different neighborhood search structure

---

Dear Mohammad Mahmoud Hamed.

Solve job shop scheduling problem with sequence dependent setup times using a simulated annealing algorithm with different neighborhood search structure

Abstract:

This paper examines job shop scheduling problems with sequence dependent setup times under objective function minimization of makespan (JSSP/SDST/Cmax). An effective meta-heuristic, simulated annealing is developed to potentially solve the problem. Simulated annealing is a well-recognized algorithm and historically classified as a local-search-based meta-heuristic. The performance of the simulated annealing critically depends on its operators and parameters. The proposed algorithm to an effective meta-heuristic simulated annealing with novel operators to potentially solve the problem. In this paper, proposed Simulated Annealing an effective neighborhood search structure based on insertion neighborhoods as well as analyzing the behavior of simulated annealing with different types of operators and parameters. The results showed that the proposed Simulated Annealing PSA algorithm gives less makespan value and CPU time with different problem size taken from the OR- library compared to previous well known SA algorithm. It note when changed in some factors with proposed NS the makespan change for the better. In independent setup times it compared the result with the solution from OR- library, have been results indicated the proposed simulated annealing algorithm near from best solution, with medium and large problems, and in small problems given best solution. It can say when the No. of temperatures between T0 and Tf increases with the proposed neighbors will improve the solution, but not continuously.

Keyword: Job shop scheduling, Sequence-dependent setup times, Simulated annealing, neighborhood search.

1. An Introduction

Job-Shop Scheduling Problem (JSSP) is one of the well-known hardest combinatorial optimization problems. JSSP being amongst the worst members of the class of NP-hard problems (non-deterministic polynomial, Gary and Johnson, in 1979) (1), there is still a lot of room for improvement in the existing techniques. Improving scheduling systems for greater customer satisfaction and operations efficiency requires an optimization criterion minimizing the makespan. Simultaneously, the sequence dependent setup time environment is a very common scheduling problem in both manufacturing and service organizations. Setup time is defined as the time interval between the end of processing time of the current job and the beginning of processing time of the next job.

Sequence dependent setup times are a tool for modeling a problem where there are different classes of operations which require machines to be reconfigured. For example, two tasks in a machine shop may both be performed on the same drill press, but require different drill bits. In Job Shop Scheduling sequence dependent setup times occur when a machine setup time or cost for a particular job is determined by not only by that job, but also by the previous job that the machine is currently set up for. Information on sequence-dependent setup times or costs for N jobs can be stored in *N*-by-*N* matrix with all diagonal entries being zero. Diagonal entries must be zero since they correspond to the condition that the machine is already setup for the next job; hence the setup time is zero.

2. Literature review

Xiaoping sun and James S. Noble (6) decomposed the job shop scheduling problem -with release dates, due dates, and sequence-dependent setup times with the scheduling objective to minimize the weighted sum of squared tardiness- into a series of single-machine scheduling problems within a shifting bottleneck framework.

Ali Allahverdi et al (8) provided an extensive review of the scheduling literature on models with setup times (costs) from then to date covering more than 300 papers. Scheduling problems are classified into those with batching and non-batching considerations, and with sequence-independent and sequence-dependent setup times, and are categorized according to shop environments, including single machine, parallel machines, flow shop, no-wait flow shop, flexible flow shop, job shop, open shop, and others.

R. Moghaddas and M.Houshmand (9) proposed a mathematical model which presents a good performance to obtain feasible solutions. The mathematical model is unable to reach the optimum results in larger problems. Thus, they developed a heuristic model based on priority rules.

B. Naderi, S.M.T. Fatemi Ghomi, and M. Aminnayeri (11) investigated scheduling job shop problems with sequence-dependent setup times under minimization of makespan. They developed an effective meta-heuristic, simulated annealing with novel operators, to potentially solve the problem. They proposed an effective neighborhood search structure

based on insertion neighborhoods as well as analyzing the behavior of simulated annealing with different types of operators and parameters. An experiment based on Taillard benchmark is conducted to evaluate the proposed algorithm against some effective algorithms. The results showed that the proposed algorithm outperforms the other algorithms.

A.Tamilarasi and T. Anantha kumar Anantha kumar (12) proposed a new method for solving job-shop scheduling problem using hybrid Genetic Algorithm (GA) with Simulated Annealing (SA). This method introduces a reasonable combination of local search and global search for solving JSSP.

Mehrzad Abdi Khalife et al (13) used simulated annealing algorithm for multi-objective flexible job shop scheduling problem with overlapping in operations to find a suitable solution. To evaluate performance of the algorithm, a mixed integer linear programming Model is developed, and solved it with the classical method (branch and bound).

A. Bagheria, and M. Zandiehb(14) considered flexible job-shop scheduling problem (FJSP) with sequence-dependent setup times to minimize makespan and mean tardiness. Neighborhood structures related to the sequencing problem and the assignment problem were employed to generate neighboring solutions. To evaluate the performance of the proposed algorithm, 20 test problems in different sizes are randomly generated.

Saeid Nourali and et al (15) proposed a mixed integer linear programming model which includes process planning and scheduling tasks simultaneously in a flexible assembly job shop with sequence dependent setup times. The objective function is minimizing maximum completion time (makespan) of final products. Also, in order to exploit the maximum flexibility in the shop floor for process planning and scheduling tasks, identical parts have been considered as distinct parts.

Mohamed A. Shalaby et al (16) considered a dynamic flexible job shop scheduling in which there are sequence-dependent setup times and machines are prone to failure. They proposed three new routing rules, and compared them with another two rules from the literature through simulation experiments.

Cemal Ozguven et al (18) considered the flexible job-shop scheduling problems with separable and non-separable sequence dependent setup times. They transformed the problem into an equivalent network problem, and formulated two mixed integer goal programming models.

M. B. Fakhrzad, A. Sadeghieh, and L. Emami (19) presented a new multi-objective job shop scheduling hybrid genetic algorithm (HGA) with sequence-dependent setup times. The objectives are to minimize the makespan and sum of the earliness and tardiness of jobs in a time window. To validate the efficiency of our proposed HGA, a number of test problems are solved.

Liji Shen (20) in this paper addresses the classic job shop scheduling where sequence dependent setup times. Used a tabu search algorithm with a sophisticated neighborhood

structure for solve this problem. Based on modified disjunctive graph, he further investigates and generalizes structural properties for the problem under study.

3. Proposed of Simulated Annealing (SA)

Simulated annealing (SA) is a local-search-based meta-heuristic which has exhibited some promise when it is applied to NP-hard problems. A typical SA starts from an initial solution and proceeds sequentially and slowly toward the area that might be far from the search area of the initial solution. The following steps, description of proposed simulated annealing.

3.1 Encoding scheme and initialization.

Encoding schemes are used to make a candidate solution recognizable for algorithms. A proper encoding scheme plays a key role in maintaining the search effectiveness of any algorithms. Since there are precedence constraints among the operations of each job, not all the permutations of the operations give feasible solutions. In the proposed algorithm generate feasible random solution, and encoding by jagged array (array of arrays). For example two machines and three jobs, show the initial solution by arrays.

Array of the Sequences processing times of Jobs on Machines J [m/c]

Job1 1 [1] 2 [2]

Job2 1 [2] 3 [1]

Job3 1 [1] 2 [2]

Initial Solution

-List of jobs needing each machine, Makespan = 11 [start: end]

M/C1: 1 [0:1] 2[1:6] 3 [6:8]

M/C2: 2[0:1] 1[1:5] 3[8:11]

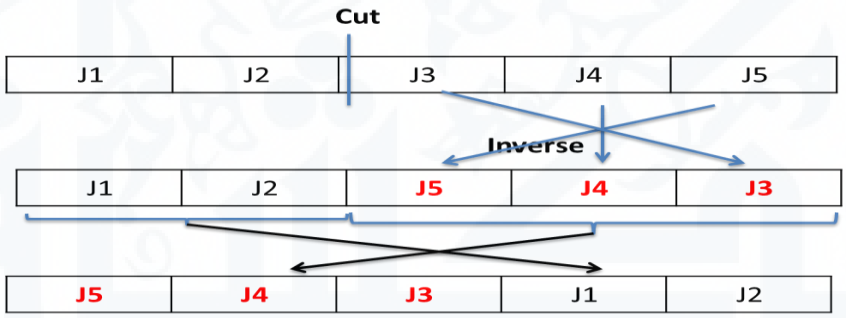
3.2 Neighborhood search structure (NSS).

Neighborhood search structure generates a new solution from current candidate solution by making a slight change in it. Many different NSSs have been applied to scheduling problems. These NSSs must work in a way that they avoid generating infeasible solutions. It seems to be often overlooked that performance of the SA depends critically on precise calibration of its neighborhood search structure. Five different NSSs are considered:

1. ***SWAP*** operator in which the random keys (RKs) of two randomly selected operations are swapped. In the algorithm we using the swap neighborhood structure simply consists of exchanging the position of two jobs in the sequence, for example, of a sequence of 5 jobs, namely j1, j2, j3, j4 and j5.slect tow random jobs j4 and j2 then swapped it, new

sequence j1, j4, j3, j2 and j5. Provided that the new solution is feasible, and repeat the previous step until proved the solution when the counting of iterations.

2. ***SHIFT*** operator in which the RKs of one randomly picked operation and put in first position. We select random a number, then put it in first position. For example, of a sequence of 5 jobs, namely j1, j2, j3, j4 and j5. Select RK from 1 to 5, e.g. 3 the new sequence is j3, j1, j2, j4 and j5.
3. ***INVERSION*** operator in which the RKs between two randomly selected cut points are reversed. Select RK between 1 to 5 then cut point and reverses. E.g. Select RK 2 the new sequence is j1, j5, j4, j3 and j2.
4. ***Migration Mechanism*** (MM) a new farther neighbor is generated from the current solution by relocating two randomly selected operations into two new randomly selected positions (i.e. The RKs of two randomly selected operations are randomly regenerated). Select two random. E.g. 5, 3 then randomly regenerated 2, 1 the new sequence j3, j5, j1, j4 and j2.
5. ***The proposed neighborhood search structure*** (PNS) is implemented in two steps; the first step is to generate a random number to cut the sequence at a random position. So, the sequence is splinted into a tail part. Then, the tail part is inverted and replaced by the head part. And the head part is replaced at the tail part. The best generated farther neighbor is accepted to move, whether it has the better objective function than the current solution or not. In the example above select random number e.g. in the first step select cut point 3, then inversed and replaced by the head part, new sequence is j5, j4, j3, j1 and j2.



Steps for generating a new neighbor in each temperature by steps:

1. We use of the SWAP operator (small neighbors). During each temperature i.
2. If the best ever visited makespan (x_{best}) is not promoted, a counter increases by one unit. This procedure is repeated until the counter reaches the number 20.
3. If during temperature i, x_{best} is improved and the counter shows a number less than 20, the counter restarts from zero.

4. If the counter becomes greater than 20, it is expected that the algorithm gets stuck in a local optimum or a loop.
5. We therefore need to generate farther neighbors than just changing the position of one operation. In new PNS we using two steps, show below:
    A. 50 new farther neighbors are generated from current solution by proposed neighborhood search PNS.
    B. The best generated farther neighbor is accepted to move whether it has the better objective function than the current solution or not.

3.3 Cooling schedule.

Neighborhood search structure is not the only aspect of the SA that is free to be chosen in a way that it improves the performance of the algorithm, the form of the energy function (cooling schedule) may also influence the behavior of the algorithm. As previously explained, to avoid local minima, solutions with worse objective values are probably accepted depending on the temperature. As the procedure proceeds, the temperature is gradually lowered under a certain mechanism called cooling schedule. Generally, there are three types of cooling schedule in the literature (see more details in [2]):

A. Linear cooling rate: $Ti = T0 - i \times ((T0 - Tf)/N)$, $i = 1, 2,,,,,,, N$.
B. Exponential cooling rate : $T_i = (A/(i+1)) + B$, $A = ((T_0 - T_f)(N+1))/N$, $B = T_0 - A$, $i = 1, 2, ,,,,,, N$.
C. Hyperbolic cooling rate: $Ti = \frac{1}{2} (T_0 - T_f) (1 - tgh((10i/N - 5))) + T_f$, $i=1, 2,,,, N$.

Where T_0, T_f and N: are initial temperature, final (stopping) temperature and desired number of temperature levels between T_0 and T_f, respectively.

```
Using C# program.
Procedure simulated annealing
t= T0
X = initialization                                  // initial solution by Random
X best = X
Counter =0
While stopping criterion is not met do
        For iter =1 to max do
                S= move  x by SWAP operator        // generating a neighbor solution from x
                If  f(s) <  f(x) then              //Acceptance criterion
                        X = s
                        Xbest = Update
        else
            if random < exp {- (f(s)- f(x))/t} then
                x = s
            endif
     endi
endfor
if Xbest is not improved then                      //  Proposed neighbor search
   counter = counter +1
   if counter > 20 then
        counter =0
      for r := 1 to 50 do
        s(r) = Generate anew father neighbors
    endfor
    x = sbest
   endif
elseif
     Counter = 0
endif
 Update Xbest
   Reduce t                                        // By cooling schedule
endwhile
```

Figure (1): Outline of the Proposed Simulated Annealing (PSA).

4. Proposed of Simulated annealing factors

In this section, we aim at analyzing the behavior of the proposed algorithm considering the above-mentioned operators and parameters. In this study, the PSA factors are: initial solution (A), combination of number of desired temperature and number of neighborhood searches in each temperature (B), initial temperature (C), cooling schedule (D) and neighborhood search structure (E).

Table (1): Factors used in Proposed Simulated Annealing

Factor	Symbol	Type
Initial solution.	A	Random
Combination number of temperatures between T_0 and T_f and number of Neighbors visited at each temperature.	B	100, 100
	B	200,200
	B	300,300
	B	1000,300
	B	200,1000
Initial temperature.	C	40

	C	30	
Cooling schedule type.	D	Exponential	
	D	Linear	
	D	Hyperbolic	
Neighborhood search structure (NSS).	E	SWAP	
	E	PNS	
	E	SHIFT	
	E	MM	

5. Design of Experiment

A set of benchmark problems is generated based on Taillard's instances. Data required for a problem consist of the number of jobs (n), number of machines (m), range of processing times (P) and range of the sequence-dependent setup times (SDST). The benchmark contains different combinations of the number of jobs n and the number of machines m. The (n, m) combinations are: ({20, 30 and 50}. (15, 20), (15 x 15) and (100 x 20) summing up 8 combinations of (n x m). The processing times in Taillard's, instances [3] with number of problems (ta01, ta12, ta24, ta32, ta41, ta51, ta61 and ta71). We used SDSTs generated from uniform distributions over U (1, 50). The algorithm is coded in program C# and run on a PC with 2.50 GHz Intel Core (TM) i5 and 4 GB of RAM.

We will make a test of simulated annealing algorithm comparison the make sure the efficiency of the program. That is done by comparing the SA with the best solution from OR- library. We will assume assumptions, as shown below:

1. Each job has its own processing route; that is, jobs visit machines in different orders.
2. Each job might need to be performed only on a fraction of m machines, not all of them.
3. Each job can be processed by at most one machine at a time and each machine can process at most one job at a time.
4. The jobs are independent; that is, there are no precedence constraints among the jobs and they can be operated in any sequence.
5. The jobs are available for their process at time 0.
6. There is no machine breakdown (i.e. Machines are continuously available).
7. The sequences of jobs in machines are independent setup times, $S_{ij} = 0$.
8. The objective function when solving or optimizing a JSS is to determine the processing order of all jobs on each machine that minimizes the makespan.

In table 3, results the job shop scheduling problem with independent setup times or setup times equal zero. Has been compared the best solution from the OR- library Taillard's [3], so make sure that the proposed program works well. The proposed SA

used factor shown in table (2). We'll name program depending on different levels from temperatures between T_0 and T_f and No. of neighbors visited at each temperature the three types PSA (1), PSA (2), PSA (3), PSA (4) and PSA (5) respectively with install the other factors in table below.

Table (2): PSA with different levels of temperature and neighbors

PSA	Cooling schedule	T_0,T_f	No. of t between T_0andT_f	No. of NS visited t
PSA (1)	Exponential	30,1	100	100
PSA (2)	Exponential	30,1	200	200
PSA (3)	Exponential	30,1	300	300
PSA (4)	Exponential	30,1	1000	300
PSA (5)	Exponential	30,1	200	1000

Table (3): Results of JSSP independent setup times by five types PSA (1), PSA (2) and PSA (3), PSA (4) and PSA (5) respectively

Problems	PSA(1)		PSA (2)		PSA (3)		PSA (4)		PSA (5)	
	Cmax	CPU sec	Cmax	CPU sec	Cmax	CPU sec	Cmax	CPU sec	Cmax	CPU sec
3x4	21	0.89	21	1	20	0.58	20	1.45	20	0.58
5x3	24	0.98	23	1.02	23	1.01	23	2.6	23	1.01
6x6	61	0.99	61	1.02	61	1.2	61	3.02	61	1.2
10x10	1359	0.52	1289	1.67	1282	1.77	1282	20.5	1282	1.77
15x15	1437	3.06	1334	4.91	1367	4.81	1200	30.2	1322	24.21
20x15	1633	3.38	1593	5.6	1548	4.25	1496	33.21	1500	29.21
20x20	1672	5.1	1844	6.65	1865	6.15	1652	39.89	1828	35.01
30x15	2194	6.2	2008	5.15	2065	4.15	1805	41.21	2001	40.09
30x20	2208	6.6	2300	7.8	2093	7.4	2000	45.26	2105	43.12
50x15	3247	8.15	3307	9.24	3246	5.24	3046	50.01	3246	50.4
50x20	3532	9..98	3437	12.98	3387	15.98	3305	53.1	3387	55.21
100x20	6246	15.56	6278	18.23	6089	17.23	6089	70.21	60089	69.52

Has been calculated the Ratio from best solution by this equation $(ORC_{\max} - PSAC_{\max})/ORC_{\max}$. Table (4) explains extent of deviation between best solutions.

Table (4): Ratio proposed PSA algorithm with different factors about best solution

Problems	Best Solution (OR)	PSA (1)	Ratio(1)	PSA (2)	Ratio(2)	PSA (3)	Ratio(3)	PSA (4)	Ratio(4)	PSA (5)	Ratio(5)
3x4	20	21	0.05	20	0	20	0	20	0	20	0
5x3	23	24	0.04	23	0	23	0	23	0	23	0
6x6	61	64	0.05	61	0	61	0	61	0	61	0
10x10	1282	1359	0.06	1289	0.01	1282	0	1282	0	1282	0
15x15	1231	1437	0.2	1334	0.08	1367	0.1	1300	0.06	1322	0.07
20x15	1376	1633	0.2	1593	0.2	1548	0.1	1496	0.09	1500	0.09
20x20	1663	1672	0.01	1893	0.1	1865	0.1	1664	0.001	1828	0.1
30x15	1770	2194	0.2	1844	0.04	2065	0.2	1805	0.02	2001	0.1
30x20	1850	2208	0.2	2008	0.09	2008	0.1	2000	0.08	2105	0.1
50x15	2760	3247	0.2	3307	0.2	3246	0.2	3046	0.1	3246	0.2
50x20	2868	3532	0.2	3437	0.2	3387	0.2	3305	0.2	3387	0.2
100x20	5464	6246	0.1	6278	0.1	6089	0.1	6089	0.1	6089	0.1

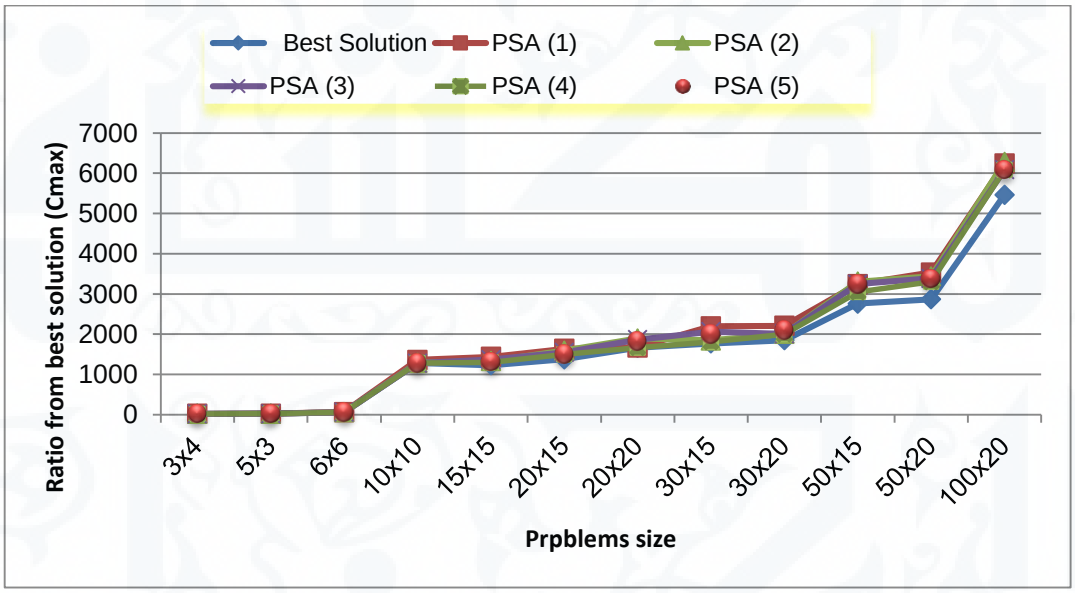


Figure (2): Ratio Approach Proposed PSA algorithm from best solution

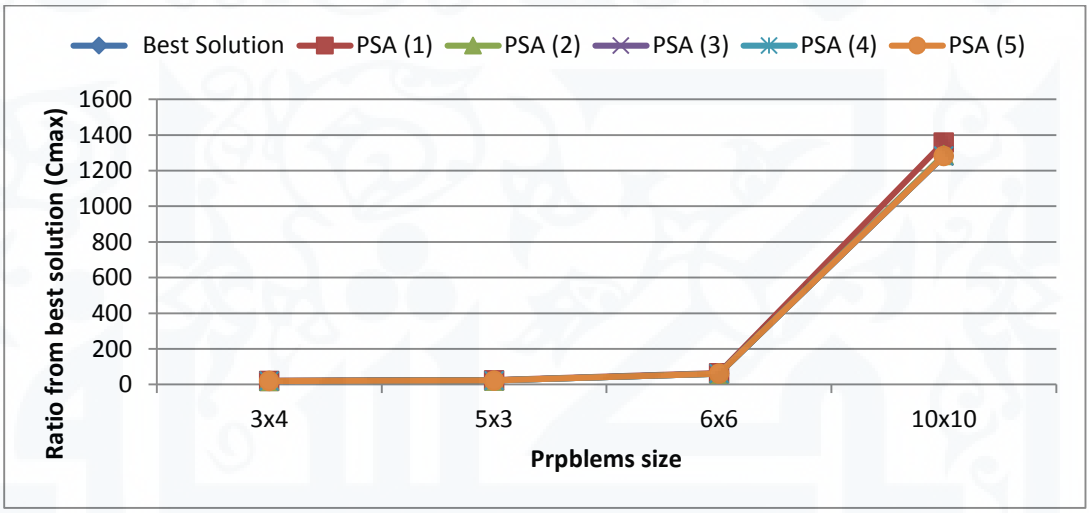


Figure (3): Ratio Approach Proposed PSA algorithm from best solution with small problems

In this table, we solve the proposed SA with different types of cooling scheduling are Exponential cooling rate, Linear cooling rate and Hyperbolic cooling rate, the results indicate that the first method gives better results in general compared by both methods.

Table (5): Comparison results the proposed PSA algorithm with different cooling schedule

Problems	Exponential cooling rate		Linear cooling rate		Hyperbolic cooling rate	
	Cmax	CPU time	Cmax	CPU time	Cmax	CPU time
15x15	1437	3.27	1370	1.27	1418	1.99
20x15	1633	4.06	1666	1.54	1683	5.3
20x20	1672	7.38	1845	3.38	1898	9.02
30x15	2194	15.25	2034	9.25	2191	10.95
30x20	2508	23.99	2476	10.99	2445	30.25
50x15	3247	30.5	3356	19.5	3356	22.14
50x20	3532	34.83	3571	24.83	3571	30.05

100x20	6246	40.12	6552	35.12	6552	40.25

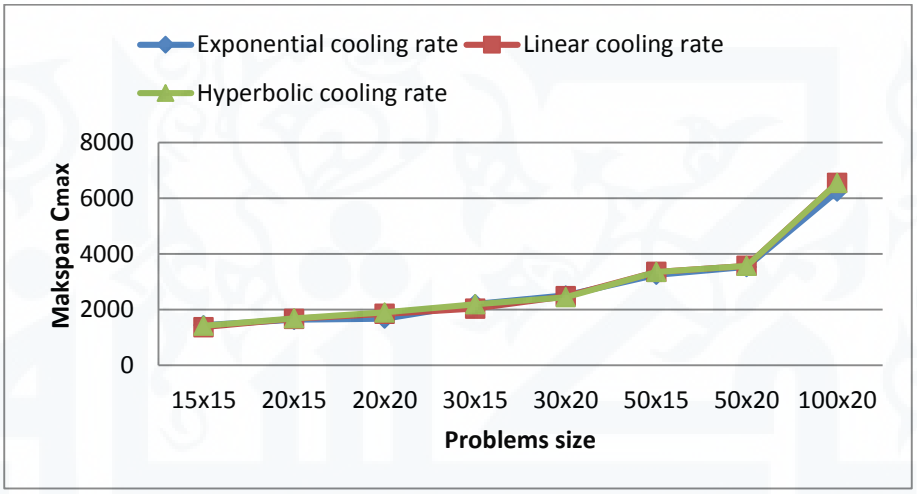


Figure (4): Makespan with different types of cooling schedule

After ascertaining that the program gives correct results, we'll same assumptions described in above, but we will change constraint number 7 to sequence dependent setup times. The sequences of jobs in machines are dependent setup times, the setup time between i and j S_{ij}.

We will comparison the proposed SA algorithm with other SA algorithm, for literature .The different between algorithms, in the SA located in the search was used factor: initial temperature T_0 =40, final temperature T_f =1, No. of temperatures between T_0 and T_f = 200. No. of neighbors visited at each temperature =200, and neighborhood search structure: MM and SHIFT. The Proposed PSA we use factor: initial temperature T_0 =30, final temperature T_f =1, No. of temperatures between T_0 and T_f =100,200, 300 and 1000. No. of neighbors visited at each temperature =100,200, 300 and 100. And use the neighborhood search structure Proposed neighborhood (PNS) and SWAP. Generate sequence dependent setup times by U (1, 50). In first time compared SA from literature (A) and PSA (B) with same factors and then used PSA (C), (D), (E), and (F) with change in factors as it is shown above.

Table (6): The SA and PSAs with different factors

PSA	**Cooling schedule**	**T_0,T_f**	**NSS**	**No. of (t) between T_0andT_f, No. of(NS) visited t**

SA (A)	Exponential	40,1	MM, Shift	200,200
PSA (B)	Exponential	40,1	PNS,Swap	200,200
PSA (C)	Exponential	30,1	PNS,Swap	100,100
PSA (D)	Exponential	30,1	PNS,Swap	300,300
PSA (E)	Exponential	30,1	PNS,Swap	1000,200
PSA (F)	Exponential	30,1	PNS,Swap	300,1000

Table (7): Results of problems with SA and PSA algorithms

Problems	SA(A)		PSA (B)		PSA (C)		PSA (D)		PSA (E)		PSA (F)	
	Cmax	CPU time	Cmax	CPU time	Cmax	CPU time	Cmax	CPU time	Cmax	CPU time	Cmax	CPU time
15x15	1987	1.56	1914	1.55	1905	1.52	1839	3.21	1808	40.21	1829	48.21
20x15	2417	2.3	2291	3.36	2250	2.52	2296	4.88	2190	45.09	2296	51.03
20x20	2706	3.9	2591	5.34	2596	2.46	2563	6.25	2483	50.9	2503	58.45
30x15	3160	10.31	2996	3.35	2979	3.2	2962	7.15	2890	59.4	2905	62.14
30x20	3661	20.85	3608	6.96	3580	3.76	3501	8.95	3479	70.65	3711	70.15
50x15	4827	30.38	4620	4.02	4643	3.11	4625	15	4600	76.65	4700	78.86
50x20	5014	35.36	4803	12.09	4968	6.01	4867	20.2	4857	89.24	4950	95.14
100x20	9436	47.58	9197	30.04	9525	9.92	9064	30.21	9064	100.05	9564	112.04

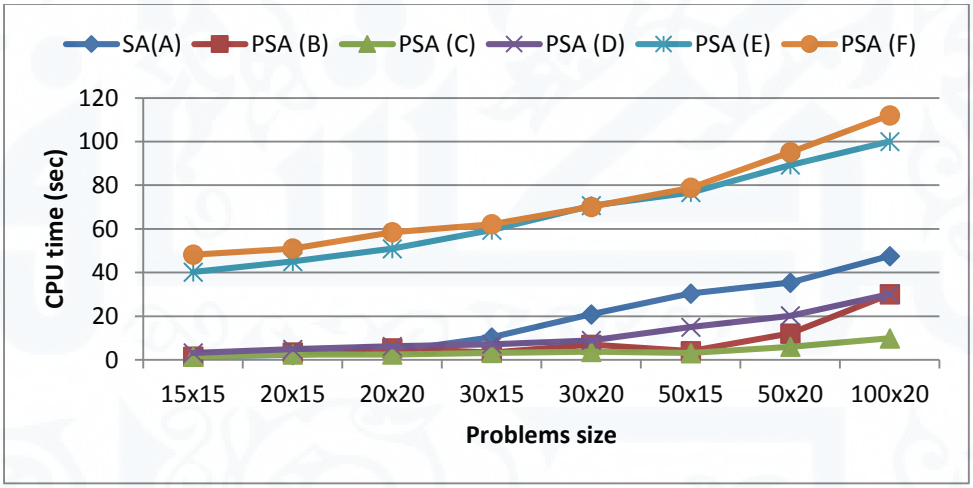


Figure (5): CPU Times of algorithms as regards increasing problem size.

6. Conclusion

In this research examined job shop scheduling with the sequence dependent setup times. The considering scheduling problems with setup times is an interest among researchers. Nowadays companies produce multiple goods on common resources and this ends up with the need for setup time activities. Setup time activities usually play a costly role in production processes; therefore planners should consider the principles of setup time activities in their schedules. To tackle the problem we apply meta-heuristic, in the framework of simulated annealing, by studying the effect of a novel neighborhood search structure with different factors. An investigation impacts of many operators and parameters (i.e. such as initial solution, cooling schedule, initial temperature and neighborhood search structure and, etc.) on the performance of SA. The numerical experiment included a set of instances generated based on Taillard's benchmark. We used sequence dependent setup times with U (1, 50). It is compared the results obtained from PSA with SA taken of literature with same factors; the results shown that the proposed PSA outperformed the other SA taken from the literature. Then using proposed simulated annealing PSA with change in factors, the results indicated the PSA with different parameters and factors, given the minimum makespan and minimum of CPU time. Used PSA with independent setup times, we compared the result with the best solution from OR- library, and have been results indicates the proposed simulated annealing algorithm near from best solution, with large problems, and in small problem size given best solution. When it increased No. of temperatures between T_0 and T_f, the makespam improved, but when it increased the number of neighbors in the solution did not improve for large. It can say when the No. of temperatures between T_0 and T_f increases with the proposed neighbors will improve the solution, but not continuously.

7. Future work

In future work use the Disjunctive graph mathematical method for solving small and medium problems of job shop scheduling with sequence dependent setup times.

Solve the Flexible job shop scheduling with sequence dependent setup times (FJSSP/SDT) by proposing simulated annealing. Solve simulated annealing using other methods for initial solution and change objective function.

References

1.Garey, Michael R.; Johnson, David S.and A W. H. Freeman, "Guide to the Theory of NP-Completeness". Computers and Intractability ISBN 0-7167-1045-5, 1979.

2. M. Lundy, A. Mees, Convergence of an annealing algorithm, Mathematical Programming 34 (1986) 111–124.

3.Taillard," Benchmarks for basic scheduling problems", European Journal of Operational Research 64 (1993) 278–285.

4.M. Gen, Y. Tsujimura, E. Kubota, Solving job shop scheduling problem using genetic algorithms, in: Proceedings of the 16th International Conference on Computer and Industrial Engineering, Ashikaga, Japan, (1994), pp. 576–579.

5.Daniel sipper and Robert L. Bulfin "production planning, control and integration", McGraw-Hill, 1998.

6.Xiaoqing sun, and James S. Noble "An approach to job shop scheduling with sequence dependent setups", Journal of manufacturing systems, volume 18, No; 6, 1999.

7.R. Cheng, M. Gen, Y. Tsujimura, A tutorial survey of job-shop scheduling problem using genetic algorithms, part II: hybrid genetic search strategies, Computers and Industrial Engineering 36 (1999) 343-364.

8.Ali Allahverdi, C.T. Ng, T.C.E. Cheng, and Mikhail Y. Kovalyov "A survey of scheduling problems with setup times or costs", European Journal of Operational Research 187 (2008) 985–1032.

9.R. Moghaddas , and M.Houshmand "Job-Shop Scheduling Problem With Sequence Dependent Setup Times" Proceedings of the International MultiConference of Engineers and Computer Scientists 2008 Vol II, IMECS 2008, 19-21 March, 2008, Hong Kong.

10.B. Naderi, M. Zandieh, A. Khaleghi Ghoshe Balagh, V. Roshanaei, An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness, Expert Systems with Applications 36 (2009) 9625–9633.

11.B. Naderi, S.M.T. Fatemi Ghomi, and M. Aminnayeri " A high performing metaheuristic for job shop scheduling with sequence-dependent setup times", Applied Soft Computing 10 (2010) 703–710.

12.A.Tamilarasi and T. Anantha kumar "An enhanced genetic algorithm with simulated annealing for job-shop scheduling" International Journal of Engineering, Science and Technology, Vol. 2, No. 1, 2010, pp. 144-151.

13.Mehrzad Abdi Khalife, Babak Abbasi, and Amir Hossein Kamali Dolat Abadia " A Simulated Annealing Algorithm for Multi Objective Flexible Job Shop Scheduling with Overlapping in Operations "Journal of Industrial Engineering, volume 5(2010) 17-28.

14.A.Bagheria, and M. Zandiehb, " Bi-criteria flexible job-shop scheduling with sequence-dependent setup times—Variable neighborhood search approach" Journal of Manufacturing Systems 30 (2011) 8–15.

15.Saeid Nourali , Narges Imanipour, and Mohammad Reza Shahriari "A Mathematical Model for Integrated Process Planning and Scheduling in Flexible Assembly Job Shop Environment with Sequence Dependent Setup Times", Int. Journal of Math. Analysis, Vol. 6, 2012, no. 43, 2117 – 2132.

16.Mohamed A. Shalaby , Tamer F. Abdelmaguid, and Zakaria Y. Abdelrasol "New Routing Rules for Dynamic Flexible Job Shop Scheduling with Sequence-Dependent Setup Times", Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management, Istanbul, Turkey, July 3 – 6, 2012

17.Jung-Hyeon Park, and Dong-Ho Lee " Job Shop Scheduling with Job Families and Sequence dependent Setups: Minimizing the Total Family Flow Time" Proceedings of the Asia Pacific Industrial Engineering & Management Systems Conference 2012.

18.Cemal Ozguven, Yasemin Yavuz, and Lale Ozbakır ,"Mixed integer goal programming models for the flexible job-shop scheduling problems with separable and non-separable sequence dependent setup times", Applied Mathematical Modelling , volume 36 (2012) 846–858.

19.M. B. Fakhrzad, A. Sadeghieh, and L. Emami "A New Multi-objective Job Shop Scheduling with Setup Times Using a Hybrid Genetic Algorithm", IJE TRANSACTIONS B: Applications Vol. 26, No. 2, (February 2013) 207-218.

20.Liji Shen "atabu search algorithm for the job shop scheduling problem with sequence dependent setup time" computer and industrial engineering, reference CAIE 3796. 2 September 2014.