Two Robust Meta-Heuristic Approaches for a New Modeling of Single Machine Scheduling Problem with Multiple Criteria

Kiarash Poursalik [1], Sina Miri-Nargesi [2*]

[1.] Department of Industrial Engineering, Islamic Azad University, Najafabad branch, Najafabad, Iran
[2.] Young Researchers Club, Qazvin branch, Islamic Azad University, Qazvin, Iran
*Corresponding Author: mirinargesy@gmail.com

Abstract: The aim of this paper is to propose a new model for a single machine-scheduling problem. According to just in time (JIT) approach, production managers should consider more than one criterion in scheduling problems. However, three criteria, including minimizing the number of tardy jobs, total weighted earliness and total weighted tardiness, are considered in this proposed model. To solve the model, firstly, branch and bound (BBA) method is applied, because it can solve the small size problems. Furthermore, the results obtained from this algorithm, are good measurement to test effectiveness of proposed meta-heuristic algorithms. In the literature, this problem is classified in the NP-hard class. Hence, two well-known meta-heuristic methods, including genetic algorithm (GA), simulated annealing (SA), are applied to tackle large scale problems. Finally, heuristic solutions were compared with the global optimum which is obtained by (BBA) method. Computational results showed that both the heuristic algorithms yield good quality solutions using reasonable computation time.
[Kiarash Poursalik, Sina Miri-Nargesi. Two Robust Meta-Heuristic Approaches for a New Modeling of Single Machine Scheduling Problem with Multiple Criteria. Journal of American Science 2011;7(7):818-825]. (ISSN: 1545-1003). http://www.americanscience.org.

Keywords: Single machine; branch and bound algorithm; genetic algorithm; simulated annealing

1. Introduction

Scheduling is the allocation of limited resources to perform a set of tasks over a period of time. Many real scheduling problems in the manufacturing industries are quite complex and very difficult to solve by conventional optimization techniques (Watanabe et al., 2005). The complexity of these problems has a direct dependence on constraints and shop environments upon which these problems are defined. One of the most well known problems in this area is single machine scheduling problem, since a complex system can be reduced to a single-machine problem, especially if there is a bottleneck machine in the system. This scheduling problem is NP-hard, but very simple to describe (Rinnooy, 1976). This type of problem became important with the advent of the just-in-time (JIT) concept. The just-in-time production philosophy has lead to a growing interest in scheduling problems considering both earliness and tardiness penalties (Monden, 1993).

Many researchers have investigated the several studies about single machine scheduling problem. A large body of literature on scheduling models with earliness and tardiness has emerged in the last two decades. There is little work in the literature handling single machine scheduling problems with different users and different criteria. First, Kanet (1981) introduced the single-machine E/T problem. Since many researchers have been worked on various extensions of the problem. Kanet (1981) examined the E/T problem with equal penalties and unrestricted common due date. As a result, Hall (2006) extended Kanet's work and developed an algorithm that finds a set of optimal solutions for the problem based on some optimality conditions. Baker and Scudder (1990) published a comprehensive state-of-the-art review for different versions of the E/T problem. Much research has been directed to scheduling problems with multiple criteria. To the best of our knowledge, Smith (1956) was the first who deal with multiple criteria in single-machine scheduling. In his paper, the total weighted flowtime and maximum tardiness were considered. However, most of the research focuses on JIT scheduling models with objective of minimizing total (weighted) costs of early and tardy jobs. Vairaktarakis and Lee (1995), Duffuaa et al. (1997) studied a single machine-scheduling problem to minimize total tardiness subject to minimal number of tardy jobs, independently.

For bicriteria scheduling models related to early and tardy costs, Chen et al. (1997) considered a single machine scheduling problem of minimizing total weighted earliness subject to maximum tardiness. They developed a heuristic and branch and bound algorithms based on the properties they derived to solve the problem. Baker and Scudder (1990) pointed out that the single-machine scheduling problem minimizing the summation of weighted earliness and tardiness with a restricted due date has not yet been addressed. Chand and

Schneeberger (1988) considered the problem of weighted earliness with no tardy jobs. The given weights and due dates were all job dependent. Guner et al. (1998) considered one machine scheduling to minimize the maximum earliness with minimum number of tardy jobs. Karasakal and Koksalan (2000) developed a simulated annealing approach to two single machine bicriteria scheduling problems: one to minimize total flowtime and maximum earliness while the other to minimize total flowtime and number of tardy jobs. Later, Koksalan and Keha (2003) also developed genetic algorithms for the two problems. Azizoglu et al. (2003a) considered a single machine scheduling problem with maximum earliness and number of tardy and no inserted idle time. They developed procedures to solve the problem optimally. Azizoglu et al. (2003b) further studied the same problem, but with allowed idle time. Chen and Sheen (2007) in their research have considered a single-machine scheduling problem with the objective of minimizing the summation of the weighted earliness and tardiness, subject to the number of tardy jobs. Wan and Yen (2009) studied a single machine scheduling problem with dual criteria, i.e., to minimize total weighted earliness subject to minimum number of tardy jobs.

To the best of our knowledge, so far no solution procedure has been proposed to this problem. In this paper, we will deal with a broad spectrum of objective functions and hence our results will apply to many different problems occurring in manufacturing. In principle, these objective functions include at least two of three different types of penalties, which can be motivated as follows:

1. Penalties arising from exceeding the contractually allotted delivery date are the most common (Explicit contract penalties, consumer dissatisfaction).
2. A penalty for an early completion of a job is appropriate for modeling capital intensive manufacturing processes, where the costs for bounded capital are an important part of the overall costs.
3. Intermediate storage costs contribute considerably to overall costs in chemical industry, if the intermediate is not stable.

The complexity of the problem necessitates meta-heuristic methods for solving large-scale problems. Our contribution is the first attempt in applying two meta-heuristics named genetic algorithm (GA) and simulated annealing (SA) to optimize this problem. Next, we compare the performance of these meta-heuristics to show the effect of the population-based and local search methods on optimization of the problem under consideration.

The remaining part of this paper is structured as follows: in Section 2 the notation needed is introduced and the problem under study is formulated; We describe our proposed methodology including a branch and bound algorithm, GA, and SA in Section 3 and 4; in Section 5 we present computational experiments, In the last section, we give our final conclusions.

2. Problem description

The simplest scheduling problem is the one in which there is a single processor or machine. Nevertheless, some features make it complicated and put it in the NP-hard class of problems. The single machine-scheduling problem is to organize the execution of n jobs on one machine. In this problem, there is a machine and a set of jobs $J = \{J_1, J_2, ..., J_n\}$. Consider a scheduling problem with n jobs to be processed on one machine with the following assumptions:

(1) Jobs are independent from each other. There are no precedence constraints among the operations of different jobs. Nonetheless, there are precedence constraints among the operations of the same job.

(2) All jobs are available at time zero.

(3) Setup time of machine and move time between operations are negligible.

(4) At a given time, a machine can only execute one operation.

(5) Pre-emption is not allowed. That is, each operation must be completed without interruption, once it starts.

In this section, we formally define the considered problem and give some useful properties. The notations are used throughout the paper are shown in Table 1.

Table 1. Nomenclature

n	Number of jobs
n_T	Number of tardy job
d_j	Common due date of job j
p_j	Processing time of job j
α_j	Weight of the earliness for job j
β_j	Weight of the tardiness for job j
γ_j	Weight of being tardy for job j
C_j	Completion time of job j
θ_1	Importance of earliness
θ_2	Importance of tardiness
θ_3	importance of number tardy job

Based on the above notations, we have the following mathematical formulation for the scheduling problem:

$Min\ Z:$

$$\theta_1 \cdot \sum_{j=1}^{n} \alpha_j * \max\{0,(d_j - C_j)\} + \theta_2 \cdot \sum_{j=1}^{n} \beta_j * \max\{0,(C_j - d_j)\} + \theta_3 \cdot \sum_{j=1}^{n} n_{Tj} \quad (1)$$

Subject to:

$$\sum_{i=1}^{3} \theta_i = 1 \quad (2)$$

$$n_{Tj} = \begin{cases} 1 & \text{if } d_j > C_j \\ 0 & \text{otherwise} \end{cases} \quad j=1,...,n. \quad (3)$$

$$C_j > 0, \quad j=1,...,n. \quad (4)$$

The objective of the problem is to find a schedule that minimizes the total weighted earliness and tardiness subject to the minimization of the number of tardy jobs. When a job j is completed before its due date, its earliness is given by $E_j = (0,\ d_j - C_j)$, where C_j, is the completion time of the job j. Conversely, if the job is finished after the due date, its tardiness is given by $T_j = (C_j - d_j,\ 0)$. Constants a_j and b_j represent job earliness and tardiness penalties, respectively. If the job j is finished after its due date, n_{Ti} will be *j*th job that will be tardy.

3. The Branch-and-Bound Algorithm

In any B&B, three major procedures are involved:

Initialization, Branching and Bounding:

During initialization, fast heuristics are usually employed to find a good initial solution. This solution serves as an upper bound (UB) for the problem until a better solution is found. This helps in eliminating (or fathoming) any nodes that have a lower bound (LB) worse than that UB.

Branching partitions the problem into smaller sub-problems. Each sub-problem represents a partial solution and is represented by a node. A search strategy must be associated with the branching scheme. This strategy decides which node to branch next.

The *bounding* procedure is used to calculate a LB at each node considered for branching to help in eliminating nodes. If the LB is worse (higher in the case of a minimization problem) than the best solution obtained so far, the node is eliminated because a better complete solution can never be reached in that case. In other words, exhaustive pursuit of the branching tree would be equivalent to complete enumeration of all sequences. The function of the bounding process is to provide a means for curtailing this enumeration (Baker and Trietsch, 2009). To find lower bound estimation we use Moore-Hodgson algorithm (Moore, 1968). There are two important reasons for using this method, which are described as follow:

1. To find minimum number of tardy jobs one of the important criteria in our model that it can be achieve whit using this algorithm.
2. In most cases the objective function, which is obtained by Moor algorithm, is an appropriate approximation of global optimum. The steps of this method can elucidate as follows:

3.1. Moore–Hodgson algorithm

Input: a job set J.

Output: the minimum number of tardy jobs for job set J.

Algorithm:

Step 0. Reindex the jobs in non-decreasing order of their due dates.

Step 1. $\sigma \leftarrow \phi, \sigma' \leftarrow \phi$.

Step2.

$If\ d_{j^*} = \min_{j \in J}\{d_j\}, \sigma \leftarrow \sigma \cup \{j^*\}, J \leftarrow J - \{j^*\}.$

Step 3. *If* $\sum_{j \in \sigma} p_j > d_{j^*}$, let k^* denote the job satisfying $p_{k^*} = \max_{j \in \sigma}\{p_j\}$ (break ties with large due date),then $\sigma' \leftarrow \sigma' \cup \{k^*\}, \sigma \leftarrow \sigma - \{k^*\}$.

Step 4. If $J = \phi$, stop; otherwise go to Step 2.

In this algorithm, r and r0 denote the non-tardy job set and tardy job set, respectively. In the sequel, we assume the jobs in $\sigma = \{1,...,|\sigma|\}$ are in non-decreasing order of their due dates and denoted as $\sigma = (1,...,|\sigma|)$, where $|\sigma|$ denotes the cardinality of set σ, i.e., the number of non-tardy jobs.

4. Two robust meta-heuristics

As mentioned before, because of computational complexity of the problem, meta-heuristic methods are applied to solve large scale problems of this type. In the following subsection, two well-known meta-heuristics that are proliferated to solve the studying issue of this paper.

4.1. Genetic algorithm

A genetic algorithm is an optimal search method motivated by natural selection and natural evolution. It maintains a population where each individual is characterized by its chromosome.

Genetic algorithms (GAs) are adaptive searching procedures for solving optimization problems based on the mechanics of natural genetics and natural selection. Since developed in the 1960s and 1970s Holland (1975), they have been applied to a wide variety of problems. Much work has been done on exploring new applications of GA and on improving their performance through genetic operator selection, parameter setting, etc., to suit the problems better. Basically, the GA procedure includes chromosome reproduction, chromosome crossover, gene mutation, chromosome fitness, and natural selection. The reproduction operator will reproduce the next generation based on their fitness value. The crossover operator, the most important step in a GA, exchanges a pair of sub-strings of their parents to generate offspring chromosomes. The mutation operator randomly selects some of the genes of each chromosome and changes their values. In this problem, the GA is implemented by steps which are described in Figure 1. The general steps of GA are as follows:

Begin
1. Initialize crossover rate (P_c), mutation rate (P_m), population size (Pop.size) and generation number (Gen. No.);
2. Generate random numbers for initial population (chromosomes);
3. Evaluate the fitness function for population;
4. Repeat the following steps until stopping condition is satisfied.
5. Select two parents from population, and apply the crossover operator over the parent chromosomes and produce two offspring chromosomes for Pc ×Pop.size times;
6. Select an individual for Pm×Pop.size times and apply mutation to the random selected chromosomes;
7. Apply reproduction operation for (1- Pc - Pm)×Pop.size times;
8. Apply heuristic method to each new individual and replace the worst chromosome by the best chromosome found so far.
9. **end**
End

Fig 1. Algorithmic skeleton of GA

4.2. Simulated annealing

Simulated annealing (SA) algorithm is based on ideas from physical annealing of solids and has proven to be a good technique for many difficult combinatorial optimization problems. Metropolis et al. (1953) was first described its main principle, SA has been applied successfully to solve a variety of problems (Kirkpatrick et al., 1983; Černỳ, 1985). SA is a local search algorithm. In a simple local search algorithm, such as a descent algorithm, an initial solution is chosen at random and then a neighboring solution is generated based on some mechanism. If the neighboring solution is better than the current solution it replaces the current solution, otherwise the current solution is retained. The process is repeated until no improving neighbor is found for the current solution.

SA has received considerable attention in the recent past and has been widely used to solve difficult combinatorial optimization problems. Many researchers have applied SA successfully to various problems with slight variations in the cooling scheme and strategies for neighbor selection. Simulated annealing has been applied to single criterion scheduling problems in the past. Potts and Van Wassenhove (1991) and Ben-Daya and Al-Fawzan (1996) were proposed SA approaches to solve single machine tardiness problems. Tan and Narasimhan (1997) addressed the problem of minimizing tardiness on a single machine with sequence-dependent set-up times. Here, we meticulously elucidate the proposed SA algorithm to optimize our problem. Following the explanation of initialization of the algorithm, the main algorithm is illuminated.

The algorithmic steps of this method are as follows:

Begin
1. Generate the initial solution, S_0;
2. Get, an initial temperature, T_0;
3. Set counter = 0 and repeat until the stopping criterion is satisfied:
4. Select a random solution $S \in N(S_0)$;
5. Set $\delta = f(s) - f(s_0)$;
6. If $\delta \prec 0$, set $S_0 = S$, otherwise select a random number between (0, 1); go to step 7;
7. If $X \prec e^{\frac{-\Delta\delta}{T}}$, set $S_0 = S$,
8. Set counter = counter + 1;
9. **While** iterations of algorithm equal iteration, set $t = \alpha(t)$,
10. Local optimum is S_0;
11. **end**
End

Fig 2. Algorithmic skeleton of SA

5. Experiment and results

In our experiments, to solve this problem we use BBA, SA, GA algorithms. The problem instances were generated in a manner similar to the one which is used in Potts and van Wassenhove (1985) and Kim and Yano (1994) also Koksalan et al. (1998) that are now a standard methods to generate single machine

scheduling problems with due dates. The integer processing times p_j are drawn from a uniform distribution in the range [1, 10]. The integer earliness penalties α_j and tardiness penalty β_j are drawn from uniform distributions in the range [1, 10] and [1, 15], respectively. Also the due date for every scheduling problem is drawn from a uniform distribution in the range [1, 10]. As well as θ_i are drown from uniform distribution in range [0, 1] respect to $\sum_{i=1}^{3} \theta_i = 1$. To test the efficiency of heuristic SA and the branch and bound algorithm (BBA), the algorithm is coded in Visual Studio C# and run on a Pentium 4 2.2, 2.1 GB RAM personal computer. In the following, we use example to illustrate two heuristics GA and SA as well as the branch and bound algorithm BBA.

5.1. Calibration of parameters

In the process of acquiring appropriate and desirable parameters, to obtain high quality answer, in this section we try using a systematic method to find effective parameters on algorithm performance. In algorithm based on (SA) method there are three factors including: initial temperature (T) cooling rate (r) and the number of iteration in a temperature, on the other hands, for algorithm based on (GA) method, considering three important factors include: pop-size, percent of crossover (P_c), percentage of mutation (P_m). In table below the value of different parameters that have tested to gain best parameters are shown in Table 2.

With regard to the desired values for each algorithm on each level of the parameters in above Table, due to the nature of meta-heuristics methods was solved 10 times, so the number of running each algorithm on each level of the parameters were 100 times. After running experiments, for algorithm's performance analysis and to determine the suitable parameters, we use the relative percentage deviation (RPD), which is defined as follows:

$$RPD = \frac{Local\ optimum_{(GA/SA)} - Global\ optimum_{(BBA)}}{Global\ optimum_{(BBA)}} \times 100 \quad (5)$$

$Global\ optimum_{(BBA)}$: Global optimum obtained from branch and bound

$Local\ optimum_{(GA/SA)}$: Local optimum obtained heuristic algorithm

After the implementation of programs, RPD is calculated for the objective function of all problems and finally, mean RPD in each level is calculated, we use RPD to normalizing outputs in each level to compare whit each other. The best parameters, show in the Table 3.

Table 2. Deviation from optimum for different parameters of algorithms

	Value of Parameters			
Algorithm	Initial Temperature	Cooling Schedule	Number Of Iterations	Deviation From Global Optimum (SA)
SA	30	0.81	100	32%
	35	0.82	150	30%
	35	0.83	200	25%
	40	0.84	250	23%
	40	0.85	300	11%
	40	0.86	350	18%
	50	0.87	400	20%
	50	0.88	450	19%
	50	0.89	500	22%
	60	0.90	500	21%
	Pop Size	Crossover	Mutation	Deviation From Global Optimum (GA)
GA	80	0/70	0/15	33%
	85	0/75	0/15	26%
	90	0/75	0/10	25%
	90	80	0/10	23%
	95	0/80	0/05	22%
	100	0/85	0/10	17%
	100	0/90	0/05	9%
	110	0/90	0/05	15%
	120	0/90	0/03	17%
	120	0/90	0/02	18%

Table 3. Best parameters for heuristics algorithms

	Best Parameters		
Algorithm	Initial Temperature	Cooling Schedule	Number Of Iterations
SA	4	0.85	300
	Pop Size	Crossover	Mutation
GA	100	0.90	0.05

5.2. Branch and bound algorithm

To illustrate the problem and the solution method we present the following example.

Example:

Suppose the problem has the job data shown in table 4.

Table 4. Job data in Example

j	1	2	3	4	5
p_j	3	7	5	6	10
d_j	6	8	10	12	25
α_j	1	1	1	1	1
β_j	1	1	1	1	1
γ_j	1	1	1	1	1

In order to find suitable lower bound we used Moore-Hadgson algorithm that coded in C#.
By the Branch and Bound algorithm (BBA), the optimal schedule is {1(0), 3(3), 4(8), 5(14), 2(24)} with two tardy jobs and global optimum for this example is obtained 3.8. In which task #{1, 3, 4} are in early sets and tasks # {2, 5} are in tardy sets.

On the other hand, about the number of tardy jobs this schedule is the same one obtaining by Moore-Hadgson algorithm. We solve our problem for n=3 up to n=9 with the Branch and Bound algorithm to gain global optimal solution. Computational results with different number of jobs in the following will be compared with two meta-heuristics.

5.3. Genetic algorithm

For proof efficiency of this meta-heuristic, GA algorithm was run 200 times. These results shown that more than 82% of solutions of GA algorithm are nearly the same as optimal solution gained with BBA algorithm. Percentage of reaches global optimums is depicted in Table 5.

Table 5. Percentage of reach to global optimum

Number Of Jobs	n=3	n=4	n=5	n=6	n=7	n=8	n=9
Number of reach to global optimum (%)	38	45	63	76	69	84	89

5.4. Simulated annealing

With the same approach for above example, SA algorithm was run 200 times. These results shown that more than 89% of solutions of SA algorithm are nearly the same as optimal solution gained with BBA algorithm. Percentage of reaches global optimums is shown in Table 6.

Table 6. Percentage of reach to global optimum

Number Of Jobs	n=3	n=4	n=5	n=6	n=7	n=8	n=9
Number of reach to global optimum (%)	42	51	68	77	68	82	87

5.5. Comparing SA and GA

After tuning the parameters of each algorithm, we found that the robust performance of GA and SA were happened when the parameters of algorithm were: initial temperature =40 cooling rate=0.85, iterations=300, Pop.size=100, P_c =0.90, P_m =0.05. Next, in the conditions in which these two algorithms act robustly, the results of both algorithms were compared with each others. To compare these results, each algorithm was run 10 times. Then, values of the objective functions were transformed into RPD values.

Computation times and optimum solution obtained from three algorithms are shown in Table 7.

Table 7. Computation time and optimum for three algorithms

Number of Jobs	Global Optimal With Branch and Bound		Average optimum (GA)		Average optimum(SA)	
	Optimum	Run Time(s)	Optimum	Run Time(s)	Optimum	Run Time(s)
n=3	1.31	0.17	1.836	0.332	1.772	0.328
n=4	2.41	0.34	2.9	0.66	2.796	0.688
n=5	3.92	0.94	4.362	0.922	4.318	1.04
n=6	5.28	3.44	5.682	1.118	5.556	1.394
n=7	8.13	9.12	8.558	1.94	8.388	2.226
n=8	11.72	28.29	12.144	3.422	12.052	3.666
n=9	16.88	170.43	17.36	5.514	17.556	7.1
n=10	21.47	612.88	21.754	7.112	22.226	8.758

A computational experiment was carried out in order to evaluate the performance of the proposed heuristic algorithms. To evaluate the quality of the heuristic solutions we use values of the percentage deviation of the heuristic algorithms from the global optimum, which is obtained by (BBA) method. We compare the performances of heuristic algorithms, using the Equation (5).

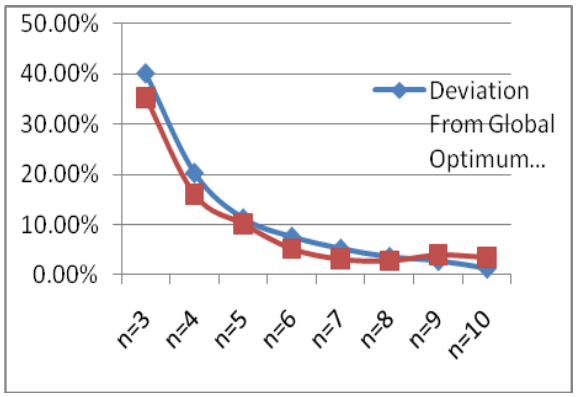

Figure 3. Deviation from global optimum

In addition, for large scale of number of jobs we solve our problem with GA and SA algorithms, and computational results for 100 Times run are shown in Table 8.

This table shows that our GA and SA algorithms for this problem are efficient. The computational results show that the heuristics performs preferably in most cases and the branch and

bound algorithm is suitable for medium-size problems. Computational results to compare two meta-heuristics which is used in this paper is depicted in Figure 4.

Table 8. Computational results for large scale of number of job

Number of Jobs	Average for 100 Runes of SA		Average for 100 Runes of GA	
	Optimum	Run Time(s)	Optimum	Run Time(s)
n=20	49.54	8.11	47.44	8.55
n=40	79.56	9.12	86.54	10.11
n=60	147.54	11.33	144.56	12.32
n=80	237.65	13.66	230.87	14.1
n=100	387.7	18.76	385.6	19.76
n=200	459.63	24.65	472.44	25.27
n=500	686.56	34.67	679.54	37.12
n=1000	1065.56	38.49	1052.67	40.26

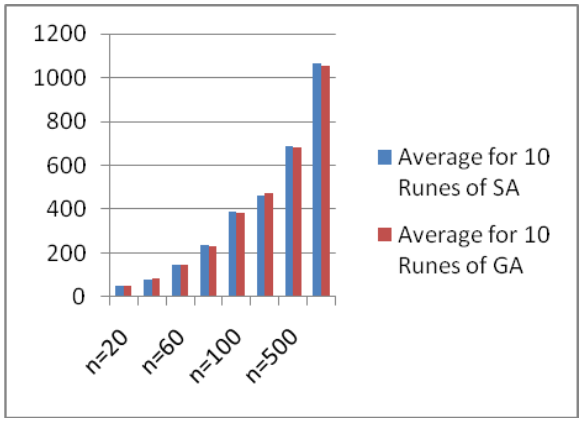

Figure 4. Compare two robust meta-heuristics

Consequently, it clearly demonstrated that in medium sized of jobs in medium sized of jobs SA has better performance than GA, but in large sized of jobs GA outperforms SA. In other words, when the total number of jobs is increased, using GA could be beneficial.

6. Conclusion

In this paper, a new model for a single machine-scheduling problem with three criteria was proposed. First, branch and bound (BBA) method was applied to solve the problem in small size. Scheduling jobs on a single machine consider specific due date for each job with respect to different earliness and tardiness penalties is an NP-hard combinatorial optimization problem. Hence, two well-known meta-heuristic methods, including genetic algorithm (GA), simulated annealing (SA), were improved to tackle large scale problems. In small size of problem, effectiveness of the heuristic algorithms GA and SA measured by means of the percentage deviation of the local optimum which was obtained by them from the lower bound on the global optimum was determined by (BBA). The computational results show that the heuristic methods work preferably in most cases and the branch and bound algorithm is suitable for medium-size problems. The computational results indicate that the proposed heuristic algorithms can be useful for scheduling in this new modeling with multiple criteria. Further research should include developing heuristics for scheduling problem with two or more parallel machines. Furthermore, the proposed approach will be applied on other scheduling problems that are well known and have similar approach to our problem.

Acknowledgements:

Authors are grateful to the Department of Industrial Engineering, Islamic Azad University, Najafabad branch, Najafabad, Iran for support to carry out this work.

Corresponding Author:
M.Sc. Sina Miri-Nargesi
Young Researchers Club, Qazvin branch,
Islamic Azad University, Qazvin, Iran
E-mail: Mirinargesy@gmail.com

References

1. Azizoglu, M., Koksalan, M. & Koksalan, S.K., 2003b. Scheduling to minimize maximum earliness and number of tardy jobs where machine idle time is allowed. Journal of the Operational Research Society, 54(6), 661–664.
2. Azizoglu, M., Kondakci, S. & Koksalan, M., 2003a. Single machine scheduling with maximum earliness and number tardy. *Computers & Industrial Engineering*, 45(2), 257–268.
3. Baker, K.R. & Scudder, G.D., 1990. Sequencing with earliness and tardiness penalties: a review. *Operations Research*, 38(1), 22–36.
4. Baker, K.R. & Trietsch, D., 2009. *Principles of sequencing and scheduling*, Wiley.
5. Ben-Daya, M. & Al-Fawzan, M., 1996. A simulated annealing approach for the one-machine mean tardiness scheduling problem. *European Journal of Operational Research*, 93(1), 61–67.
6. Černỳ, V. 1985. "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm." Journal of optimization theory and applications 45(1): 41–51.

7. Chand, S., Schneeberger, H., 1988. Single machine scheduling to minimize weighted earliness subject to no tardy jobs. *European Journal of Operations Research*, 34, 221–230.
8. Chen, T., Qi, X., Tu, F., 1997. Single machine scheduling to minimize weighted earliness subject to maximum tardiness. *Computers and Operations Research*, 24, 147–152.
9. Chen, W.Y. & Sheen, G.J., 2007. Single-machine scheduling with multiple performance measures: Minimizing job-dependent earliness and tardiness subject to the number of tardy jobs. *International Journal of Production Economics*, 109(1-2), 214–229.
10. Duffuaa, S.O. et al., 1997. One-machine scheduling to minimize mean tardiness with minimum number tardy. *Production Planning & Control*, 8(3), 226–230.
11. Guner, E., Erol, S. & Tani, K., 1998. One machine scheduling to minimize the maximum earliness with minimum number of tardy jobs. *International Journal of Production Economics*, 55(2), 213–219.
12. Hall, N.G., 2006. Single-and multiple-processor models for minimizing completion time variance. *Naval Research Logistics Quarterly*, 33(1), 49–54.
13. Holland, J. H., 1975. "*Adaption in natural and artificial systems*." University of Michigan Press, Ann Arbor.
14. Kanet, J.J., 1981. Minimizing the average deviation of job completion times about a common due date. *Naval Research Logistics Quarterly*, 28(4), 643–651.
15. Karasakal, E.K. & Koksalan, M., 2000. A simulated annealing approach to bicriteria scheduling problems on a single machine. Journal of Heuristics, 6(3), 311–327.
16. Kim, Y.D., Yano, C.A., 1994. "Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates." Naval Research Logistics, 41, 913–933.
17. Kirkpatrick, S., Gelatt, C., and Vecchi, M., (1983), "Optimization by Simulated Annealing", Science 20/4598, 671-680.
18. Koksalan, M., Azizoglu, M. & Kondakci, S.K., 1998. Minimizing flowtime and maximum earliness on a single machine. *IIE transactions*, 30(2), 192–200.
19. Koksalan, M., Keha, A.B, 2003. Using genetic algorithms for single-machine bicriteria scheduling problems. *European Journal of Operational Research*, 145(3), 543–556.
20. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. 1953. Equation of state calculations by fast computing machines. Journal of Chemical Physics, 21(6), 1087–1092.
21. Moore, J.M., 1968. An n job, one machine sequencing algorithm for minimizing the number of late jobs. Management Science, 102–109.
22. Potts, C.N. & Van Wassenhove, L.N., 1991. Single machine tardiness sequencing heuristics. *IIE transactions*, 23(4), 346–354.
23. Potts, C.N. & van Wassenhove, L.N., 1985. "A branch and bound algorithm for the total weighted tardiness problem." *Operations Research*, 33, 363–377.
24. Rinnooy Kan A. H. G., 1976. "Machine Scheduling Problems: Classification, Complexity, and Computations." The Hagues, Martinus Nijhoff,.
25. Smith, W.E., 1956. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2), 59–66.
26. Tan, K.C. & Narasimhan, R., 1997. Minimizing tardiness on a single processor with sequence-dependent setup times: a simulated annealing approach. *Omega*, 25(6), 619–634.
27. K Vairaktarakis, G.L. & Lee, C.Y., 1995. The single-machine scheduling problem to minimize total tardiness subject to minimum number of tardy jobs. *IIE transactions*, 27(2), 250–256.
28. Wan, G. & Yen, B.P., 2009. Single machine scheduling to minimize total weighted earliness subject to minimal number of tardy jobs. *European Journal of Operational Research*, 195(1), 89–97.
29. Watanabe, M., Ida, K. & Gen, M., 2005. A genetic algorithm with modified crossover operator and search area adaptation for the job-shop scheduling problem. *Computers & Industrial Engineering*, 48(4), 743–752.

3/5/2011