Available online at www.sciencedirect.com

ScienceDirect

Applied Mathematical Modelling 31 (2007) 1351–1361

APPLIED MATHEMATICAL MODELLING

www.elsevier.com/locate/apm

Minimizing total tardiness in a scheduling problem with a learning effect

Tamer Eren [a], Ertan Güner [b,*]

[a] *Kırıkkale University, Faculty of Engineering, Department of Industrial Engineering 71450 Kırıkkale, Turkey*
[b] *Gazi University, Faculty of Engineering and Architecture, Department of Industrial Engineering 06570 Maltepe, Ankara, Turkey*

Received 1 March 2005; received in revised form 1 December 2005; accepted 23 March 2006
Available online 14 June 2006

Abstract

Minimizing of total tardiness is one of the most studied topics on single machine problems. Researchers develop a number of optimizing and heuristic methods to solve this NP-hard problem. In this paper, the problem of minimizing total tardiness is examined in a learning effect situation. The concept of learning effects describes the reduction of processing times arising from process repetition. A 0–1 integer programming model is developed to solve the problem. Also, a random search, the tabu search and the simulated annealing-based methods are proposed for the problem and the solutions of the large size problems with up to 1000 jobs are found by these methods. To the best of our knowledge, no works exists on the total tardiness problem with a learning effect tackled in this paper.
© 2006 Elsevier Inc. All rights reserved.

Keywords: Single machine scheduling; Tardiness; Learning effect; 0–1 Integer programming; Heuristic methods; Tabu search

1. Introduction

In traditional scheduling problems, most research assumes that all jobs have constant processing times; in particular, job processing times are independent of their position in the scheduling sequence. However, this assumption is impractical in many situations. Recent empirical studies in several industries have verified that unit costs decline as firms produce more of a product and gain knowledge or experience [1–4]. This phenomenon is known as the “learning effect” in the literature [5]. The learning effect has been widely employed in management science since its discovery by Wright [6] over half a century ago. However, there have been very few studies in the general context of production scheduling. These studies were about on single [5,7–14] and multi-machines [15–19].

The production of a family of similar components or operations using similar equipment and tooling is tenet of group technology [20]. In a group production and/or cellular manufacturing system, a group or a cell will process only a limited number of different, but similar jobs in a finite production cycle repeatedly.

[*] Corresponding author. Tel.: +90 312 2317400 2855; fax: +90 312 2308434.
E-mail addresses: teren@kku.edu.tr (T. Eren), erguner@gazi.edu.tr (E. Güner).

0307-904X/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2006.03.030

Therefore the machine operators will repetitively perform a set of similar operations. Because of this, operators will gain experience, move upwards on the learning curve and produce parts faster with a higher quality level over time [21]. As a result group technology and cellular manufacturing are highly prevalent in small batch and job production systems. The practical importance of taking learning effects into consideration in scheduling similar jobs for production is quite evident.

In this paper, we consider the total tardiness problem with a learning effect in a single machine. Single machine scheduling does not necessarily involve one machine. It can also involve a group of machines or a system that can be treated as one machine.

Minimizing the total tardiness is one of the most important criteria in single machine scheduling and it is a NP-hard problem [22]. Many implicit enumeration techniques such as branch-and-bound [23–28], dynamic programming [29–37], dominance rules [38,39] and decomposition [28,30,33,35,40–42] have been proposed for the static total tardiness problem. Table 1 provides a brief summary of the important advancements in research towards solving the static total tardiness problem to optimality. However, high computational requirements of branch and bound procedures and high memory requirements of dynamic programming techniques make them inapplicable for large scale problems. For heuristics, see the literature review by Russell and Holsenback [43]. On the other hand, the dynamic version of the total tardiness problem has usually been attacked by referring to priority rules, see, for example, Baker and Bertrand [44] and Raghu and Rajendran [45]. Biskup and Simons [46] have considered a dynamic total tardiness problem arising out of a decentralized job acquisition by several agents and presented two cost allocation schemes.

Since the total tardiness problem without learning effect in a single machine is a NP-hard problem; the problem tackled here is also a NP-hard problem. To the best of our knowledge, no works exists on the total tardiness with a learning effect.

The rest of the study is organized as follows. In Section 2, the problem and the proposed 0–1 integer programming model are described. The heuristics that are used to solve large size problems are presented in Section 3. The experimental results are given in Section 4. Finally, Section 5 provides conclusions and evaluations of the study and suggests some directions for future researches.

Table 1
Optimizing techniques for the total tardiness problems [47]

Year	Method	Complexity	Reference
1961	BB[a]		Schild and Fredman [23]
1962	DP[b]		Held and Karp [29]
1964	DP	$O(n2^n)$	Lawler [30]
1968	Network model		Elmaghraby [48]
1969	Dominance rules		Emmons [38]
1971	DP-based hybrid		Srinivasan [31]
1972	BB	30 jobs, $\leqslant 4.13$ s	Shwimer [24]
1975	BB	20 jobs, >300 s	Rinooy Kan et al. [25]
1976	BB/ Lagragian relaxtion		Fisher [26]
1977	Decompozition/DP	$O(n^4 \sum p_i)$	Lawler [32]
1978	Decompozition/DP	50 jobs	Baker and Schrage [33]
	BB	20 jobs, $\leqslant 12.8$ s	Picard and Queyranne [27]
	DP	50 jobs, 0.844 s	Schrage and Baker [34]
1982	Decompozition/DP	$O(n^7/\varepsilon)$	Lawler [35]
	Lawler decompositions extension		Potts and Van Wassenhove [40]
1983	MINIT		Sen et al. [49]
1987	DP	$\leqslant 100$ jobs	Potts and Van Wassenhove [36]
1991	DP		Sen and Borah [37]
1995	Decomposition extension		Chang et al. [41]
1996	Extension of Emmons' dominance		Yu [39]
	Decomposition/BB	100–150 jobs	Szwarc and Mukhopadhyay [28]
1997	Decomposition		Tansel and Sabuncuoglu [42]

[a] BB: Branch-and-bound.
[b] DP: Dynamic programming.

2. Problem description

In considered problem we assume that a set $N = \{1, 2, \dots, n\}$ of n simultaneously available jobs is to be processed on a single machine. p_j denotes the processing time of job j, d_j denotes the due date of the job j. A job once started on the machine must be completed on it without interruption (i.e., no preemption is allowed). The machine may not process more than one operation at a time. When the completion time of any job j is greater than its due date, the tardiness occurs for this job j. That is $\sum T = \sum_{j=1}^{n} \max\{C_j - d_j, 0\}$. C_j is the completion time, T_j is the tardiness of the job respectively. That is if p_{jr} is the actual processing times of job j scheduled in position r in a sequence, then $p_{jr} = p_j r^a$ (where $a \leqslant 0$ is the learning index, given is the logarithm to the base 2 of the learning rate) the objective is to find a schedule that minimizes the total tardiness on single machine (the problem is denoted as $1/\mathrm{LE}/\sum T$, where LE is learning effect).

3. A 0–1 integer programming model

In this proposed model, there are $n^2 + 5n$ variables and $6n$ constraints, where n denotes the number of jobs. The parameters and variables in the model are described below and then the proposed model is given.

Parameters:

j	the number of jobs, $j = 1, 2, \dots, n$
p_j	the processing time of job j, $j = 1, 2, \dots, n$
d_j	the due date of job j, $j = 1, 2, \dots, n$
a	learning index

Decision variables:

Z_{jr}	If job j is scheduled at the rth position to be processed, $Z_{jr} = 1$ otherwise $Z_{jr} = 0$, $j = 1, 2, \dots, n$, $r = 1, 2, \dots, n$
T_r	the tardiness for the rth position job $r = 1, 2, \dots, n$

Auxiliary variables:

p_r the learning effect processing time of the rth ranked job

$$p_r = \sum_{j=1}^{n} r^a Z_{jr} p_j, \quad r = 1, 2, \dots, n. \tag{1}$$

C_r the completion time of the rth ranked job $r = 1, 2, \dots, n$

d_r^* the due date of the rth ranked job

$$d_r^* = \sum\nolimits_{j=1}^{n} Z_{jr} d_j, \quad r = 1, 2, \dots, n. \tag{2}$$

0–1 integer programming model:

Objective function :

$$\mathrm{Min} = \sum_{r=1}^{n} T_r$$

Constraints :

$$\sum_{j=1}^{n} Z_{jr} = 1, \quad r = 1, 2, \dots, n, \tag{3}$$

$$\sum_{r=1}^{n} Z_{jr} = 1, \quad r = 1, 2, \dots, n, \tag{4}$$

$$C_1 \geqslant A_1,$$
$$C_r \geqslant C_r - 1 + p_r, \quad r = 1, 2, \ldots, n, \tag{5}$$
$$T_r \geqslant C_r - d_r^*, \quad r = 1, 2, \ldots, n. \tag{6}$$

Z_{jr}: 0–1 and all variables are positive and integer.

Constraint (3) specifies that only one job be scheduled at the rth job priority. Constraint (4) defines that each job be scheduled only once. Constraint (5) represents that the beginning processing time of the rth ranked job be greater than or equal to the previous jobs completion time at the machine. Constraint (6) stipulates that the tardiness of the rth ranked job be greater than or equal to difference between (C_r) and (d_r^*).

4. Heuristic methods

Most of the scheduling problems are NP-hard problems. Because of this, only small size problems can be solved optimally by the branch-and-bound, the dynamic programming and the integer programming models. But, for the real applications in industry to deal with large size problems can be necessary. The researchers have studied to develop heuristic methods for solving these type problems.

The considered single machine scheduling with learning effect problem can be solved optimally for small size problems with up to 25 jobs by the proposed integer programming model. That is why we applied meta heuristic methods to solve large size problems. These heuristics are three tabu search based methods, three simulated annealing based methods and a random search method. These methods are explained in the following sections.

4.1. Tabu search heuristic

Tabu search has been used widely in combinatorial optimization. The basic idea is to slightly alter a known (current) solution in a certain manner (called neighborhood structure) and take the best alteration as the new current solution. Such altered solutions are called neighbors of the current solution. An operation that yields a neighbor is called a move. To avoid being trapped at a local optima, the best neighbor that is worse than the current solution is allowed to become the new current solution. To avoid cycling, certain moves are marked as tabu. A tabu move may be allowed if an aspiration criterion is satisfied. This procedure continues until a criterion is met [50].

In this study, an experimental design was made for determining suitable tabu search parameters. To the results, the chosen tabu search parameters are given in Table 2. Additionally, according to the used initial solutions, the tabu search methods were considered into three groups (TS-I, TS-II, TS-III). These are the random, EDD, E&G solutions. The methodology of E&G heuristic is shortly given below.

E&G heuristic

The methodology of E–G heuristic is similar to NEH [51] procedure. Jobs are ordered in descending order of their due dates. The best partial sequence of first two jobs is found about minimizing total tardiness with learning effect processing times and the third job is inserted into the best two-job partial sequence and the best three-job partial sequence is chosen. In this way the complete sequence is constructed.

Table 2
Parameters of the tabu search

Parameters	TS-I	TS-II	TS-III
Initial solutions	Random	EDD	E&G
Tabu list length	$2\sqrt{n}$	$2\sqrt{n}$	$2\sqrt{n}$
Neighborhood search strategy:	API	API	API
Stopping criterion	n Iteration no improvement	n Iteration no improvement	n Iteration no improvement

4.2. Simulated annealing

Simulated annealing (SA) is a local search algorithm (meta-heuristic) capable of escaping from local optima. Its ease of implementation, convergence properties and its use of hill-climbing moves to escape local optima have made it a popular technique over the past two decades [52].

SA is a neighborhood search technique that has produced good results for combinatorial problems. A standard SA procedure begins by generating an initial solution at random. At each stage, the new solution taken from the neighborhood of the current solution is accepted as the new current solution if it has a lower or equal cost (in the our study objective function value); if it has a higher cost it is accepted with a probability that decreases as the difference in the costs increases, and as the temperature of the method decreases. This temperature, which is simply a positive number, is periodically reduced by some temperature scheme, so that it moves gradually from a relatively high value to near zero as the method progresses. Thus at the start of SA, most worsening moves are accepted, but at the end only improving ones are likely to be accepted. The method converges to a local optimum as the temperature approaches zero, but because SA has performed many perturbations at a high temperature which have pushed the search path into new areas, a better local optimum solution should hopefully be reached [53]. A standard SA algorithm can be stated in Fig. 1 [54].

Some parameters and rules must be set to solve any problem with the SA algorithm. These are: (1) the initial value of the temperature parameter, (2) the number of iterations to be performed at each temperature, (3) a cooling function, (4) a stopping criterion to terminate the algorithm. According to experimental design results, the chosen simulated annealing parameters in the problem are given in Table 3. Additionally, according to the used initial solutions, the simulated annealing methods were considered into three groups (SA-I, SA-II, SA-III). These were the random, EDD, E&G solutions.

```
Step 1.
        Given an initial solution $i \in S$;
        Given simulation temperature values $T_0 > T_F > 0$;
        Setting an iterative counter $k = 0$.
Step 2.
        Generating an initial solution $j \in N_i$;
        Evaluating $\Delta f = f(j) - f(i)$.
Step 3.
        If 'Metropolis criterion' is satisfied, i.e., $\min\{1, \exp(-\Delta f / T_k)\} > \eta \in [0,1)$,
        then $i = j$.
Step 4.
        If 'Metropolis equilibrium' under $T_k$ is realized, then go to Step 5; Else go to
        Step 2.
Step 5.
        If 'stop criterion' is not satisfied, i.e., $T_k > T_f$, then reducing the temperature
        $T_{k+1} = T_k - \Delta T_k, T_k > 0$, and $k = k + 1$, go to Step 2; Else output $i_{opt} = i$.
End.
```

Fig. 1. A standard SA algorithm.

Table 3
Parameters of the simulated annealing

Parameters	TS-I	TS-II	TS-III
Initial solutions	Random	EDD	E&G
Neighborhood search strategy:	API	API	API
Initial temperature	500	500	500
Final temperature	1	1	1
Cooling rate	0.995	0.995	0.995
Maximum iteration length at a particular temperature	1	1	1

4.3. Random search

Random search is a method that selects a specific number of solution point (a sample size) randomly from the solution space. Random search evaluates the selected points (sequences) about their objective function values and identifies the best sequence in the sample. The best sequence is stored at the memory and then this process is repeated. When the new best sequence is better than the previous one, the previous sequence is updated with the new sequence otherwise it does not change. The search is stopped at a prespecified number of iterations. There are two parameters in random search method: The first is the specification of the sample size and the other is the stopping rule. Stopping can be realized either at a certain number of iterations or when there is no improvement a given number of consecutive iterations [55,56].

In the experimental study we have considered $(n-1)$ as the sample size, and no improvement for n repetitions as the stopping rule for the random search of the problem.

5. Experimental results

In this study, all experimental tests were conducted on a personal computer with Pentium IV/2 512 Ram. The integer programming model is used to find the optimal solutions of the considered problem using Hyper LINDO/PC 6.01 software package [57]. The heuristic methods used in this paper were coded on C++ Builder 5. The exact solutions for four different job sizes as $n = 10$, 15, 20 and 25 were found. Ten repetitions were made for each problem size.

Heizer and Render, [4] indicate that different products have different learning curves, and that the rate of learning varies depending upon the potential of the process and product. They provide that the learning rates vary between 70% and 90% for different manufacturing industries in the US between 1920 and 1988. For examples; 79%, and 80% were in the steel industry and aircraft assembly, respectively. So, in this study we assumed 80% as a learning rate.

In this study, experimental analyses are realized at two stages:

1. Finding exact results for small size problems, and solving the small size problems with heuristic methods, and comparing the solutions with the exact results.
2. Solving the large size problems with the heuristic methods.

5.1. Best results and errors of heuristic methods

The experimental design was similar to Hsu and Lin [58] and processing times on the single machine was generated from a uniform distribution in the ranges [1, 100]. The due dates were randomly generated from another uniform distribution on the integers between $[0 - C_{\max}/2]$, $[0 - C_{\max}]$, $[C_{\max}/2 - C_{\max}]$ and $[0 - 3C_{\max}/2]$. Here, $C_{\max}$ was obtained by an SPT sequence with taking into account the learning effect processing times (that is; $C_{\max} = \sum_{r=1}^{n} p_{[r]} r^{a}\,\text{rank} : \min_{r=1}^{n} p_{[r]}$). The experimental set is given in Table 4. As seen from Table 4, totally 160 small size problems were solved.

Table 4
The parameters for small size problem ($n \leqslant 25$)

Parameter	Alternatives	Values
Processing time (p_j)	1	$\sim U[1, 100]$
Number of jobs (n)	4	10, 15, 20, 25
Due date range (DDR)	4	$\sim U[0 - C_{\max}/2]$
		$\sim U[0 - C_{\max}]$
		$\sim U[C_{\max}/2 - C_{\max}]$
		$\sim U[0 - 3C_{\max}/2]$
Number of solution problem	10	
Total problem		$1 \times 4 \times 4 \times 10 = 160$

The average CPU times of the integer programming model (in terms of seconds) in relation to the number of jobs are shown in Fig. 2. Four different due date range were used for the problems. As seen in Fig. 2, the longest CPU times belong to the problems with $U[0 - 3C_{\max}/2]$-due date range. On the other hand, the shortest CPU times belong to problems with $U[0 - C_{\max}]$-due date range.

The optimal solutions of the considered problem were found up to 25 job size. To solve large size problems in a short time and to find the optimal or near optimal solutions, heuristics should be used. For this reason, seven heuristic methods that are three tabu search methods, three simulated annealing methods and a random search method were used. Heuristic error is calculated as follows:

$$\text{Error} = \frac{\text{Heuristic solution} - \text{Optimal solution}}{\text{Optimal solution}}.$$

The results of the heuristic methods were compared with the results of the optimal solutions obtained by the integer model, and the errors of these heuristics are given in Fig. 3. As seen in Fig. 3, firstly the TS-III and

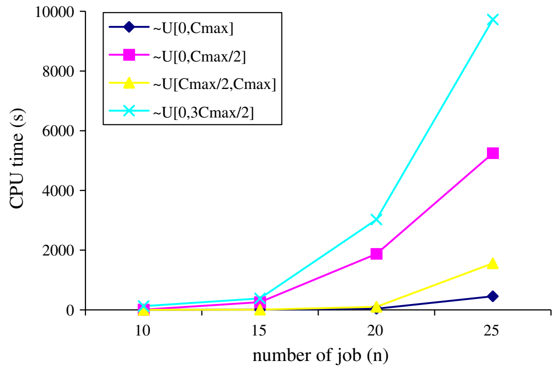

Fig. 2. CPU times of the integer programming.

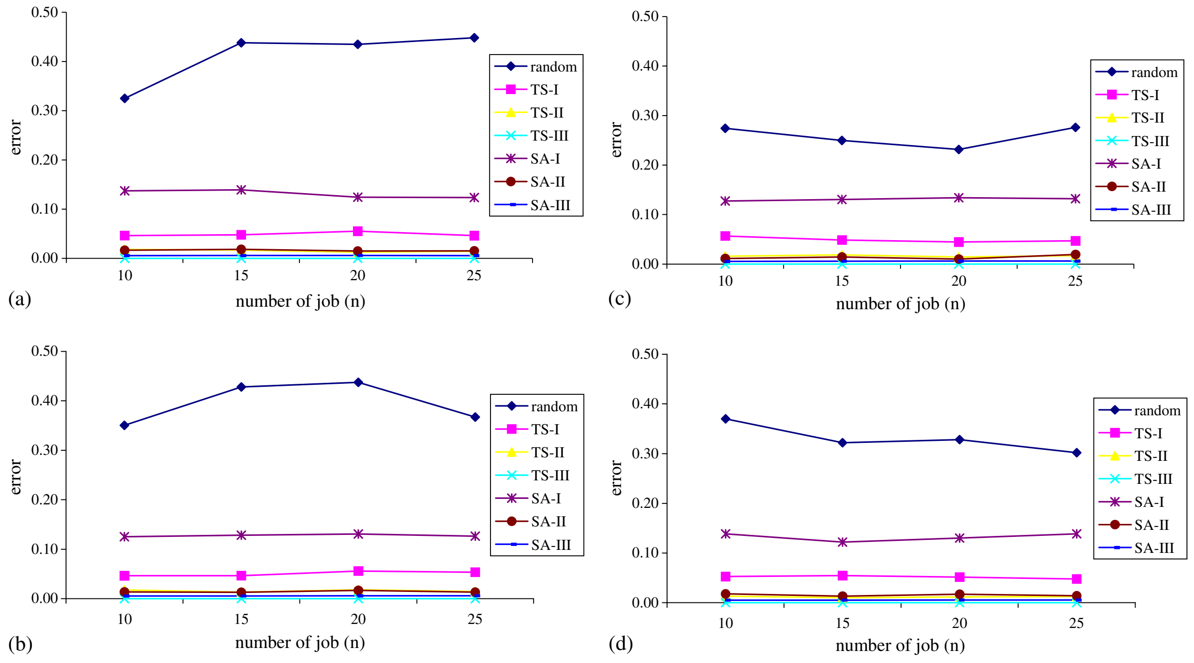

Fig. 3. Average heuristic errors for the different due date ranges ($n \leqslant 25$). (a) $[0 - C_{\max}/2]$, (b) $[0 - C_{\max}]$, (c) $[C_{\max}/2 - C_{\max}]$, (d) $[0 - 3C_{\max}/2]$.

secondly SA-II give fairly good results in terms of error for allude date combinations and these methods are quite effective for the considered small size problems. On the other hand, the random search gives the worst results for all due date combinations.

5.2. The large size problem results

The results of the large size problems ($100 \leqslant n \leqslant 1000$) were also computed using heuristic methods. Since the optimal results of these problems have not been known, the results of the heuristics were compared with the best result for defining their performances. In this comparison, the error, formulated below, was used as a performance measure:

$$\text{Error} = \frac{\text{Heuristic solution} - \text{Best heuristic solution}}{\text{Best heuristic solution}}.$$

The experimental set of the large size problems is given in Table 5. As seen in Table 5, totally 400 problems were solved.

Table 5
The parameters for larger size problem ($n \leqslant 25$)

Parameter	Alternatives	Values
Processing time (p_j)	1	$\sim U[1, 100]$
Number of jobs (n)	10	$100, 200, \ldots, 1000$
Due date range (DDR)	4	$\sim U[0 - C_{\max}/2]$
		$\sim U[0 - C_{\max}]$
		$\sim U[C_{\max}/2 - C_{\max}]$
		$\sim U[0 - 3C_{\max}/2]$
Number of solution problem	10	
Total problem		$1 \times 10 \times 4 \times 10 = 400$

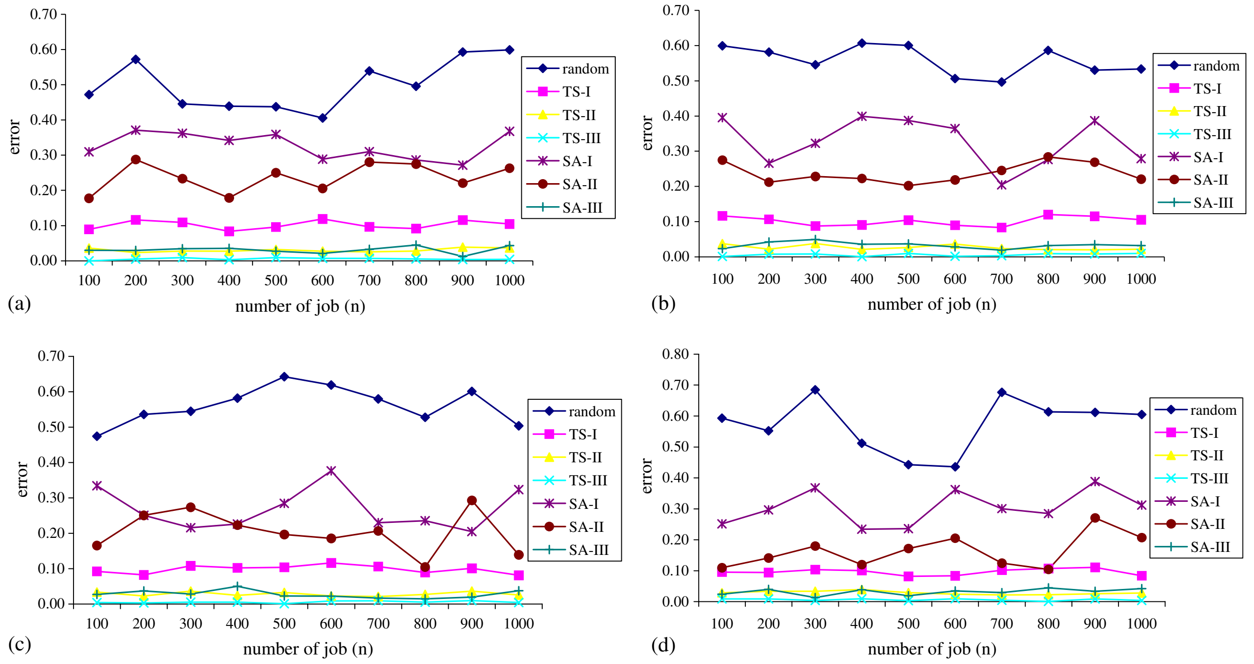

Fig. 4. Average heuristic errors for the different due date ranges ($100 \leqslant n \leqslant 1000$). (a) $[0 - C_{\max}/2]$, (b) $[0 - C_{\max}]$, (c) $[C_{\max}/2 - C_{\max}]$, (d) $[0 - 3C_{\max}/2]$.

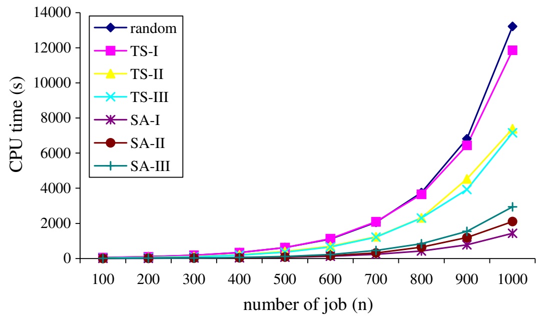

Fig. 5. The average CPU times of heuristics for$100 \leqslant n \leqslant 1000$.

Fig. 4 presents, the solution qualities of heuristics for large size problems according to the due date range values. The TS-III gives the best results of the problem for all due date range values. Again the random search gives the worst results for large size problems for all due date values. According to Fig. 4, mostly tabu search based methods gave better results than the simulated annealing methods, except for SA-III. SA-III also gave fairly good results for all due date ranges.

The execution times of heuristics for small size problems were so small that they were neglected. However, the average execution times of the heuristics for large size problems cannot be neglected as shown in Fig. 5. It is clear from this Fig. 5 that the longest execution times belong to the random search and TS-I follows it.

6. Conclusion

In this paper, we consider the single machine total tardiness problem with the consideration of the learning effect. An integer programming model was developed for the problem and the solution results were obtained up to 25 job size. Due date data of the problem were generated from four distinct ranges. Three heuristic methods that are the tabu search based, simulated annealing based and the random search and their variations were proposed to get the solutions of the problem with up to 1000 jobs. According to the experimental comparisons, tabu search based methods have given better solution values about solution quality than the results of simulated annealing and random search for all due date ranges except for SA-III. SA-III has given also good results as TS-III. The two methods used the same initial solution taken from EG heuristic. As a result, it can be stated that there is a virtual superior impact of EG heuristic on the effectiveness of SA-III and TS-III.

By this study, meta heuristic methods were firstly used for the learning effect scheduling problem and also the total tardiness that is one of the well known and used criterion in scheduling literature was considered under the learning effect.

In scheduling the multi-machine or multi-criteria cases under the learning effects can be also interesting issues for future studies.

References

[1] L.E. Yelle, The learning curve: historical review and comprehensive survey, Decision Sciences 10 (1984) 302–327.

[2] E.S. Buffa, Meeting the Competitive Challenge, Irwin, Homewood, 1984.

[3] C. Li, T.C.E. Cheng, The parallel machine min–max weighted absolute lateness scheduling problem, Naval Research Logistics 41 (1994) 33–46.

[4] J. Heizer, B. Render, Operations Management, sixth ed., Prentice-Hall, 2001.

[5] D. Biskup, Single machine scheduling with learning considerations, European Journal of Operational Research 115 (1999) 173–178.

[6] T.P. Wright, Factors affecting the cost of airplanes, Journal of the Aeronautical Sciences 3 (1936) 122–128.

[7] G. Mosheiov, Scheduling problems with learning effect, European Journal of Operational Research 132 (2001) 687–693.

[8] G. Mosheiov, J.B. Sidney, Scheduling with general job-dependent learning curves, European Journal of Operational Research 147 (2003) 665–670.

[9] W.-C. Lee, C.-C. Wu, H.-J. Sung, A bi-criterion single-machine scheduling problem with learning considerations, Acta Informatica 40 (2004) 303–315.

[10] T.C.E. Cheng, G. Wang, Single machine scheduling with learning effect considerations, Annals of Operations Research 98 (2000) 273–290.
[11] D. Biskup, D. Simons, Common due date scheduling with autonomous and induced learning, European Journal of Operational Research 159 (2004) 606–616.
[12] W.-C. Lee, A note on deteriorating jobs and learning in single-machine scheduling problems, International Journal of Business and Economics 3 (2004) 83–89.
[13] T. Eren, E. Güner, Application of learning effect on single machine scheduling problem, 10. Ergonomics Congress, Bursa, Turkey (7–9 October) 2004 (In Turkish).
[14] G. Mosheiov, J.B. Sidney, Note on scheduling with general learning curves to minimize the number of tardy jobs, Journal of the Operational Research Society Sidney 56 (2005) 110–112.
[15] G. Mosheiov, Parallel machine scheduling with learning effect, Journal of the Operational Research Society 52 (2001) 165–1169.
[16] T. Eren, E. Güner, Flowshop scheduling with general job-dependent learning effect, K.H.O. Journal of Defense Sciencesi 2 (2003) 1–11 (In Turkish).
[17] T. Eren, E. Güner, Minimizing mean flow time in a flowshop scheduling with learning effect, Journal of the Faculty of Engineering and Architecture of Gazi University 19 (2004) 119–124 (In Turkish).
[18] W.-C. Lee, C.-C. Wu, Minimizing total completion time in a two-machine flowshop with a learning effect, International Journal of Production Economics 88 (2004) 85–93.
[19] T. Eren, E. Güner, Application of learning effect on parallel machine bicriteria scheduling problem, YA/EM'2004, XXIV. National Congress 473-475, Gaziantep – Adana, Turkey, 15-18 June 2004 (In Turkish).
[20] E.K. Ivanov, Group Production Organization and Technology (E. Bishop, Trans.), Business Publications, London, 1968.
[21] T.J. Greene, R.P. Sadowski, A review of cellular manufacturing assumptions, advantages and design techniques, Journal of Operations Management 4 (1984) 85–97.
[22] J. Du, J.Y.T. Leung, Minimizing total tardiness on one machine is NP-hard, Mathematics of Operations Research 15 (1990) 483–495.
[23] L. Schild, K.R. Fredman, Scheduling tasks with linear loss functions, Management Science 7 (1961) 280–285.
[24] J. Shwimer, On the *n*-job one machine sequence independent scheduling problem with tardiness penalties: a branch-and-bound solution, Management Science 18 (1972) 301–313.
[25] A.H.G. Rinnooy Kan, B.J. Lageweg, J.K. Lenstra, Minimizing total costs in one machine scheduling, Operations Research 23 (1975) 908–927.
[26] M.L. Fisher, A dual problem for the one machine scheduling problem, Mathematics Programming 11 (1976) 229–251.
[27] J. Picard, M. Queyranne, The time-dependent traveling salesman problem and its application to the tardiness problem in one machine scheduling, Operations Research 26 (1978) 86–110.
[28] W. Szwarc, S. Mukhopadhyay, Decomposition of the single machine total tardiness problem, Operations Research Letters 19 (1996) 243–250.
[29] M. Held, R.M. Karp, A dynamic programming approach to sequencing problems, SIAM Journal 10 (1962) 196–210.
[30] E.L. Lawler, On scheduling problems with deferral costs, Management Science 11 (1964) 280–288.
[31] V. Srinivasan, A hybrid algorithm for the one machine sequencing problem to minimize total tardiness, Naval Research Logistics Quarterly 18 (1971) 317–327.
[32] E.L. Lawler, A pseudo polynomial algorithm for sequencing jobs to minimize total tardiness, Annals of Discrete Mathematics 1 (1977) 331–342.
[33] K.R. Baker, L. Schrage, Finding an optimal sequence by dynamic programming:an extension to precedence-related tasks, Operations Research 26 (1978) 111–120.
[34] L. Schrage, K.R. Baker, Dynamic programming solution of sequencing problems with precedence constraints, Operations Research 26 (1978) 444–449.
[35] E.L. Lawler, A fully polynomial approximation scheme for the total tardiness problem, Operations Research Letters 1 (1982) 207–208.
[36] C.N. Potts, L.N. Van Wassenhove, Dynamic programming and decomposition approaches for the single machine total tardiness problem, European Journal of Operational Research 32 (1987) 405–414.
[37] T. Sen, B.N. Borah, On the single machine scheduling problem with tardiness penalties, Journal of the Operational Research Society 42 (1991) 695–702.
[38] H. Emmons, One machine sequencing to minimize certain functions of job tardiness, Operations Research 17 (1969) 701–715.
[39] W. Yu, Augmentation of consistent partial orders for the one machine total tardiness problem, Discrete Applied Mathematics 68 (1996) 189–202.
[40] C.N. Potts, L.N. Van Wassenhove, A decomposition algorithm for the single machine tardiness problem, Operations Research Letters 1 (1982) 177–181.
[41] S. Chang, Q. Lu, G. Tang, W. Yu, On decomposition of the total tardiness problem, Operations Research Letters 17 (1995) 221–229.
[42] B.C. Tansel, I. Sabuncuoglu, New insights on the single machine total tardiness problem, Journal of the Operational Research Society 48 (1997) 82–89.
[43] R.M. Russell, J.E. Holsenback, Evaluation of leading heuristics for the single machine tardiness problem, European Journal of Operational Research 96 (1997) 538–545.
[44] K.R. Baker, J.W.M. Bertrand, A dynamic priority rule for scheduling against due-dates, Journal of Operations Management 3 (1982) 37–42.

[45] T.S. Raghu, C. Rajendran, An efficient dynamic dispatching rule for scheduling in a job shop, International Journal of Production Economics 32 (1993) 301–313.
[46] D. Biskup, D. Simons, Game theoretic approaches to cost allocation in the dynamic total tardiness problem, IIE Transactions 31 (9) (1999) 899–908.
[47] T. Sen, J.M. Sulek, P. Dileepan, Static scheduling research to minimize weighted and unweighted tardiness: a state-of-the-art survey, International Journal of Production Economics 83 (2003) 1–12.
[48] S.E. Elmaghraby, The one machine scheduling problem with delay costs, Journal of Industrial Engineering 19 (1968) 105–108.
[49] T. Sen, L.M. Austin, P. Ghandforoush, An algorithm for the single machine sequencing problem to minimize total tardiness, IIE Transactions 15 (1983) 363–366.
[50] F. Glover, Future paths for integer programming and links to artificial intelligence, Computers and Operations Research 5 (1986) 533–549.
[51] M. Nawaz, E.E. Enscore, I. Ham, A heuristic algorithm for the *m*-machine, *n*-job flow-shop sequencing problem, Omega 11 (1983) 91–95.
[52] D. Henderson, S.H. Jacobson, The theory and practice of simulated annealing, in: F. Glover, G. Kochenberger (Eds.), Handbook of Metaheuristics, Kluwer Academic Publishers, Norwell, MA, 2002, pp. 287–320.
[53] C. Low, J. Yeh, K. Huang, A robust simulated annealing heuristic for flow shop scheduling problems, International Journal of Advanced Manufacturing Technol 23 (2004) 762–767.
[54] P. Tian, J. Ma, D. Zhang, Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: an investigation of generation mechanism, European Journal of Operational Research 118 (1999) 81–94.
[55] J.–S.R. Jang, C.T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice-Hall, USA, 1997.
[56] R.L. Fox, Optimization Methods for Engineering Design, Addision Wesley Publishing Company, London, 1971.
[57] Lindo Systems, Inc., Hyper LINDO/PC Release 6.01, Chicago, USA, 1997.
[58] Y.S. Hsu, B.N.T. Lin, Minimizing of maximum lateness under linear deterioration, Omega 31 (2003) 459–469.

Update

Applied Mathematical Modelling

Volume 35, Issue 7, July 2011, Page 3652

DOI: https://doi.org/10.1016/j.apm.2011.01.032

Applied Mathematical Modelling 35 (2011) 3652

Contents lists available at ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Erratum

Erratum to “Minimizing total tardiness in a scheduling problem with a learning effect” [Appl. Math. Model. 31 (2007) 1351–1361]

Dehua Xu, Yunqiang Yin [*]

School of Mathematics and Information Sciences, East China Institute of Technology, Fuzhou, Jiangxi 344000, China

As we observe, there are some errors in the 0–1 integer programming model of the paper [1]. In this erratum, we point out these errors and correct them.

The objective function of the model should be “Min $\sum_{r=1}^{n} T_r$”.

The line just before constraint set (5) should be deleted since it has nothing to do with the model (the notation A_1 in that line is not defined).

Constraint set (5) should be replaced by

$$C_r \geqslant C_{r-1} + \sum_{j=1}^{n} r^a Z_{jr} p_j, \quad r = 1, 2, \ldots, n \tag{5}$$

and

$$C_0 = 0.$$

We believe that there is a typo in the original version of constraint set (5), i.e., “C_{r-1}” is misprinted as “$C_r - 1$”. And we believe that it is better we replace “p_r” with “$\sum_{j=1}^{n} r^a Z_{jr} p_j$” in order to avoid misunderstanding since p_r can also be understood as the processing time of job r according to the definition of the notation “p_j” in Section 2.

The line just after constraint set (6) should be “$Z_{jr} = 0$ or 1 ($j = 1,2,\ldots,n$; $r = 1,2,\ldots,n$) and all the other variables are non-negative integers.” since T_r may be zero.

Acknowledgements

This research was supported in part by the Natural Science Foundation of Jiangxi (2010GQS0003), the Science Foundation of Education Committee of Jiangxi for Young Scholars (GJJ11143, GJJ11144), and the Doctor Fundation of East China Institute of Technology.

Reference

[1] T. Eren, E. Güner, Minimizing total tardiness in a scheduling problem with a learning effect, Appl. Math. Model. 31 (2007) 1351–1361.

DOI of original article: 10.1016/j.apm.2006.03.030

[*] Corresponding author. Tel./fax: +86 794 8258307.
E-mail addresses: dhxu@ecit.edu.cn, schedulingresearch@gmail.com (D. Xu), yinyunqiang@126.com (Y. Yin).

0307-904X/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2011.01.032