Scientia Iranica E (2018) 25(1), 370–385

Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
http://scientiairanica.sharif.edu

Single-machine scheduling to minimize the maximum tardiness under piecewise linear deteriorating jobs

A.A. Jafari and M.M. Lotfi*

Department of Industrial Engineering, Faculty of Engineering, Yazd University, Yazd, Iran.

Received 9 December 2015; received in revised form 16 October 2016; accepted 25 December 2016

KEYWORDS
Scheduling;
Piecewise linear deteriorating jobs;
Single machine;
Tardiness;
Branch and Bound;
Heuristic.

Abstract. In many realistic production environments, jobs will take longer time if they begin later. This phenomenon is known as deteriorating jobs which have widely been studied. In this paper, the piecewise linear deterioration is discussed in a single-machine scheduling problem of minimizing the maximum tardiness. After proving the *NP-hardness* of problem, a Branch and Bound and a heuristic algorithm with $O(n^2)$ are proposed to solve the large-scale problems by near-optimal solutions. The heuristic approach is also used to determine an upper bound on the solution of B&B algorithm. The computational results of evaluating performance of the two algorithms confirm the excellent performance of B&B algorithm as it is able to solve the problems with at least 32 jobs within a reasonable time. Notably, the heuristic approach is quite accurate and efficient with an average error percentage of less than 0.3%.

1. Introduction

Nowadays, scheduling problems are applied in different production and service systems. In traditional scheduling problems, it is assumed that the job processing times are known and constant. This assumption may not be true for all the cases; there are many situations in which a job consumes more time when processed later. In fact, when a given job delays its starting time or waits for process, its processing time may be increased [1]. In usual, the increment in processing time is a function of starting time or position in sequence [2]. These kinds of job are known as deteriorating jobs, and there is a growing interest to study them in the literature.

*. *Corresponding author. Tel.: +98 035 31232409;*
Fax: +98 353 8210699
E-mail addresses: a.jafari@stu.yazd.ac.ir (A.A. Jafari); lotfi@yazd.ac.ir (M.M. Lotfi)

doi: 10.24200/sci.2017.4408

Applications of deteriorating jobs can be found in the fire-fighting, maintenance planning and scheduling, medical procedures, and searching for an object under worsening weather or growing darkness [1]. Rolling process in the steel industries is another well-known case of deteriorating jobs. Steel ingots must be heated up to a predetermined temperature in preheating stage; each ingot rolled earlier has less heat exchange with environment and its preheating time will be shorter. Another important application of the above situation is lathing process, in which the needed lathing time will be increased due to gradual exhaustion of tools [3].

In this paper, Graham symbols [4], in the form of $\alpha|\beta|\gamma$, is used where α, β, and γ demonstrate machine environment, problem specification, and objective function, respectively. The variables and parameters used in this paper are described in Table 1.

There is a growing interest in the literature to study the scheduling problems with deteriorating jobs [5-6]. Alidaee and Womer [7] classified the deterioration functions into three different kinds: linear, piecewise linear, and non-linear. Most authors assumed

Table 1. Variables and parameters.

Description	Notation	Description	Notation
Number of jobs	n	Maximum lateness	$L_{\max} = \max_{t \leq i \leq n} \{L_t\}$
ith job	$j_i;\ i = 1, 2, ..., n$	Tardiness of j_i	$T_i = \max\{0, C_i - d_i\}$
Set of all jobs to be scheduled	$N = \{j_1, j_2, ..., j_n\}$	Maximum Tardiness	$T_{\max} = \max_{l \leq i \leq n} \{T_i\}$
Actual processing time of j_i	$P_i;\ i = 1, 2, ..., n$	Partial sequence of scheduled jobs	δ
Normal processing time of j_i	$a_i;\ i = 1, 2, ..., n$	Set of unscheduled jobs (complementary of δ)	δ'
Deterioration rate of j_i	$b_i\ i = 1, 2, ..., n$	Partial sequence with scheduling j_i after δ	δ_i
Starting time of each job	S	Maximum tardiness of partial sequence δ	$T_{\max}(\delta)$
Due date of j_i	$d_i,\ i = 1, 2, ..., n$	Completion time of j_i	$C_i;\ i = 1, 2, ..., n$
$U_i = 0$ if $d_i \geq C_i$; otherwise $U_i = 1$	$U_i;\ i = 1, 2, ..., n$	Weight of j_i	$w_i;\ i = 1, 2, ...n$
Release time of j_i	$r_i;\ i = 1, 2, ..., n$	Maximum completion time	$c_{\max} = \max_{l \leq i \leq n} \{C_i\}$
Total completion time	$\sum_{i=1}^{n} C_i$	Weighted total completion time	$\sum_{i=1}^{n} w_i C_i$
Number of tardy jobs	$N_T = \sum_{i=1}^{n} U_i$	Number of weighted tardy jobs	$\sum_{i=1}^{n} w_i U_i$
Lateness of j_i	$L_i = C_i - d_i$		

a linear or piecewise linear deterioration function. The problem of deteriorating jobs was reviewed by Cheng et al. [8]. They assumed that the processing time of a job is a linear function of its starting time. In the linear functions, the deterioration rates may be similar or different. In different deterioration rates, normal processing times might be zero or positive. Therefore, the linear deterioration functions are as follows:

$$P_i = a_i + b_i S, \quad P_i = a_i + bS, \quad P_i = b_i S.$$

Browne and Yechiali [9] showed that the optimal sequence in problem $1|P_i = a_i + b_i S|_{C_{\max}}$ is based on non-decreasing rate of a_i/b_i. Bachman and Janiak [10] proved that problem $1|P_i = a_i + b_i S|L_{\max}$ is NP-complete and presented two heuristics with complexities of O (n.log n) and O(n^2). The problem was also investigated by Hsu and Lin [11] and a Branch and Bound (B&B) algorithm was proposed which was able to solve 100 jobs. Ng et al. [12] proposed a B&B algorithm for problem $F2|P_i = a_i + b_i S|\sum C_i$ which was able to handle problems with 15 jobs. Also, they involved a heuristic in the proposed B&B algorithm as an upper bound. Lee et al. [13] considered problem $Fm|P_{ij} = a_{ij} + b_i t|\sum T_i$ and developed a B&B and two metaheuristic algorithms. Yin et al. [14] studied parallel machine scheduling of deteriorating jobs with disruption and presented pseudo-polynomial time solution algorithms. Luo and Ji [15] considered single-machine scheduling with variable maintenance under deteriorating jobs as $1|VM, P_i = a_i + b_i S|C_{\max}$, and proved that the problem is NP - hard.

Lee et al. [16] proposed a heuristic and B&B algorithm to minimize makespan in a Linear Deteriorating Jobs Scheduling Problem (LDJSP) with release time, i.e. $1|P_i = a_i + bS, r_i\,|C_{\max}$; the proposed algorithm solved problems with 28 jobs. Wu and Lee [17] presented a B&B and several heuristics for problem $F2|P_i = a_i + b_i S|\bar{F}$. The paper was extended by Lee et al. [18] and a B&B and several heuristics were developed to minimize makespan. Jafari and Moslehi [1] proved that problem $1|P_i = a_i + bS\,|\sum U_i$ is NP-hard; hence, a B&B procedure and a heuristic with O(n^2) as an upper bound were proposed. Wang and Wang [19] studied problem $F3|P_{ij} = a_{ij} + bS|C_{\max}$ and derived several dominance properties, some lower bounds and two heuristic algorithms and applied them in a proposed B&B algorithm to find the optimal solution. Yin et al. [20] considered some two-agent single-machine scheduling problems with increasing linear job deterioration and proved their complexity.

The LDJSP with zero normal processing time ($P_i = b_i S$) on a single-machine was investigated by Mosheiov [21]; he presented the optimal solutions using simple rules for performance criteria $C_{\max}$, $\sum C_i$, $\sum W_i C_i$, $T_{\max}$, $L_{\max}$, and $\sum U_i$. Wang et al. [22] showed that problem $F2|P_i = b_i S\,|\sum C_i$ is NP-hard; they proposed a B&B algorithm able to handle 14 jobs. Yang and Wang [23] developed a B&B and heuristic algorithm for problem $F2|P_i = b_i S\,|\sum W_i C_i$. This kind of deterioration function was considered by the others. Some assumed that the machines are not available at any time due to preventive maintenance or breakdown. For instance, Woo and Lee [24] studied the availability constraints on a single machine in two resumable and non-resumable cases. They proposed an integer programming model and a heuristic algorithm, respectively, for two problems:

$$1|r - a, P_i = b_i S\,|C_{\max},$$

and:

$$1|nr - a, P_i = b_i S\,|\sum C_i.$$

Some authors supposed that setup time of each job is not constant; it is a simple linear function of its starting time similar to processing time. Cheng et al. [25] presented a B&B algorithm for problem $1|P_i = b_iS, S_i = b_i'S\,|T_{\max}$. Lee et al. [26] proposed a B&B algorithm for problem $1|P_i = b_iS, S_i = b_i'S\,|\sum U_i$ which could solve the instances up to 1000 jobs in a reasonable time. Lee and Lu [2] provided a B&B algorithm for problem $1|P_i = b_iS, S_i = b_i'S\,|\sum W_iU_i$.

Some authors assumed that the deterioration function is piecewise linear in which the actual processing time of each job is a function of two or more than two constant or linear criteria. Kubiak and Velde [27] studied problem $1|P_i\,|C_{\max}$ in which P_i is considered as a non-decreasing three-criteria function as follows:

$$P_i = \begin{cases} a_i & \text{if} \quad S \le y_1 \\ a_i + b_i(S - y_1) & \text{if} \quad y_1 < S < y_2 \\ a_i + b_i(y_2 - y_1) & \text{if} \quad S \ge y_2 \end{cases} \tag{1}$$

where y_1 and y_2 are the input variables. They showed that the problem in special case, $y_2 = \infty$ and $y_1 > 0$, is NP-hard and proposed a binary B&B algorithm. Moslehi and Jafari [3] surveyed the piecewise linear deteriorating jobs scheduling problem where the deterioration function is similar to Eq. (1) and the objective is to minimize the number of tardy jobs. They proved that problem:

$$1|P_i = a_i + b_i(S - y_1),\ y_1 > 0,\ y_2 > y_1|\sum_{i=1}^{n} U_i,$$

is NP-hard and developed a B&B procedure and a heuristic algorithm. Lalla Ruiz and Vob [28] considered problem $Pm|P_i = a_i$ or $a_i + b_i|\sum C_i$ and presented two mathematical models.

Jafari and Moslehi [29] proved that problem:

$$1|P_i = a_i + b_i(S - y_1),\ y_1 > 0,\ y_2 > y_1|\sum_{i=1}^{n} W_i\,U_i,$$

is NP-hard and provided a B&B algorithm able to handle 28 jobs. Lai et al. [30] presented the optimal solutions to a single-machine problem with non-linear deterioration function. Lee and Yu [31] provided pseudo-polynomial time algorithms to optimize the parallel machine scheduling under potential disruption.

In Table 2, we present a review on the studies directed onto the deteriorating jobs scheduling problems where dispatching rule, heuristics, and integer

Table 2. Researches on the deteriorating jobs scheduling problems.

Ref. no.	Deterioration function	Objective	Problem	Solution approach
[9]	Linear ($P_i = a_i + b_iS$)	$C_{\max}$	$1\|P_i = a_i + b_iS\|C_{\max}$	DR
[10]	Linear ($P_i = a_i + b_iS$)	$L_{\max}$	$1\|P_i = a_i + b_iS\|L_{\max}$	Heu
[11]	Linear ($P_i = a_i + b_iS$)	$L_{\max}$	$1\|P_i = a_i + b_iS\|L_{\max}$	B&B
[12]	Linear ($P_i = a_i + b_iS$)	$\sum C_i$	$F2\|P_i = a_i + b_iS\|\sum C_i$	B&B
[17]	Linear ($P_i = a_i + b_iS$)	$\bar{F}$	$F2\|P_i = a_i + b_iS\|\bar{F}$	B&B, Heu
[18]	Linear ($P_i = a_i + b_iS$)	$C_{\max}$	$F2\|P_i = a_i + b_iS\|C_{\max}$	B&B, Heu
[16]	Linear ($P_i = a_i + b_iS$)	$C_{\max}$	$1\|P_i = a_i + bS, r_i\|C_{\max}$	B&B, Heu
[1]	Linear ($P_i = a_i + b_iS$)	$\sum U_i$	$1\|P_i = a_i + bS\|\sum U_i$	B&B, Heu
[21]	Linear ($P_i = b_iS$)	$C_{\max}$, $\sum C_i$, $\sum W_iC_i$, $T_{\max}$, $L_{\max}$, $\sum U_i$	$1\|P_i = b_iS\,\|C_{\max}$ $1\|P_i = b_iS\,\|\sum C_i$ $1\|P_i = b_iS\,\|\sum W_iC_i$ $1\|P_i = b_iS\,\|T_{\max}$ $1\|P_i = b_iS\,\|L_{\max}$ $1\|P_i = b_iS\,\|\sum U_i$	DR
[22]	Linear ($P_i = b_iS$)	$\sum C_i$	$F2\|P_i = b_iS\,\|\sum C_i$	B&B
[23]	Linear ($P_i = b_iS$)	$\sum W_iC_i$	$F2\|P_i = b_iS\,\|\sum W_iC_i$	B&B, Heu
[24]	Linear ($P_i = b_iS$)	$C_{\max}$, $\sum C_i$	$1\|r-a, P_i = b_iS\,\|C_{\max}$ $1\|nr-a, P_i = b_iS\,\|\sum C_i$	Heu, IP
[25]	Linear ($P_i = b_iS$)	$T_{\max}$	$1\|P_i = b_iS, S_i = b_i'S\,\|T_{\max}$	B&B
[26]	Linear ($P_i = b_iS$)	$\sum U_i$	$1\|P_i = b_iS, S_i = b_i'S\,\|\sum U_i$	B&B
[2]	Linear ($P_i = b_iS$)	$\sum W_iU_i$	$1\|P_i = b_iS, S_i = b_i'S\,\|\sum W_iU_i$	B&B
[27]	Piecewise Linear	$C_{\max}$	$1\|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 = \infty\|C_{\max}$	B&B
[3]	Piecewise Linear	$\sum U_i$	$1\|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1\|\sum_{i=1}^{n} U_i$	B&B, Heu
[29]	Piecewise Linear	$\sum W_iU_i$	$1\|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1\|\sum_{i=1}^{n} W_i\,U_i$	B&B
This study	Piecewise Linear	$T_{\max}$	$1\|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1\|T_{\max}$	B&B, Heu

programming are shown as DR, HE, and IP, respectively.

As can be seen, maximum tardiness as a performance measure has only been considered in a specific case of liner deterioration function with zero normal processing time ($P_i = b_i S$) [25]; the actual processing time of each job in real applications is as a piecewise linear function as in Figure 1 so that the deterioration happens in a period of time after the starting process leading to an increase in the actual processing time. This increment will not, however, continue to the end; in fact, after a specific time, the value of deterioration will be constant until the end of the process. It is noteworthy to mention that the piecewise linear deterioration function, addressed in this paper, may cover all the possible forms of linear deterioration functions. Finally, no research can be found in the scheduling problems with piecewise linear deterioration function and minimization of the maximum tardiness. The problem is focused in our paper.

The rest of the paper is organized as follows. In Section 2 the problem definition and its complexity are presented. In Section 3 a heuristic algorithm and Section 4 a B&B procedure in are proposed to solve the problem. In Section 5, computational experiments are developed in order to test the performance of algorithms. Conclusions and directions for future research are presented in Section 6.

2. Problem

According to Table 1, we describe and formulate our considered problem. There are n jobs in set N, $N = \{j_1, j_2, ..., j_n\}$, to be processed on a single machine. All the jobs are available at time 0 and will be processed without interruption or preemption. The machine is available all the time, and it can handle no more than one job at a time. We assume that each job has a specific deterioration rate, and the actual processing time of j_i is based on a piecewise linear function of its starting time, S, as in Eq. (2) and Figure 1, where y_1 and y_2 are considered as parameters:

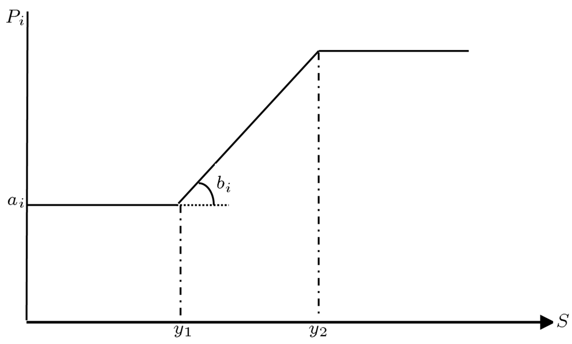

Figure 1. Actual processing time of j_i.

$$P_i = \begin{cases} a_i & \text{if } S \leq y_1 \\ a_i + b_i(S - y_1) & \text{if } y_1 < S < y_2 \\ a_i + b_i(y_2 - y_1) & \text{if } S \geq y_2 \end{cases} \tag{2}$$

The objective is to find an admissible schedule, such that the maximum tardiness is minimized. The problem is demonstrated as:

$$1|P_i = a_i + b_i(S - y_1),\ y_1 > 0,\ y_2 > y_1|T_{\max}.$$

We analyze the problem complexity at first. If any problem P reduces to problem Q and problem P is NP-hard, then problem Q will also be NP-hard [32]. Cheng et al. [25] showed that problem $1|P_i = b_i S, S_i = b_i' S|T_{\max}$ is NP-hard. The problem is reducible to:

$$1|P_i = a_i + b_i(S - y_1),\ y_1 > 0,\ y_2 > y_1|T_{\max},$$

therefore, the latter is also NP-hard. Accordingly, it is reasonable to utilize general procedures, such as B&B, to find only the optimal solution and heuristic algorithm to find a near-optimal solution.

3. Heuristic algorithm

In this section, a heuristic algorithm with $O(n^2)$ is developed to solve the problem. In any iteration, one job among n jobs is chosen and scheduled. The steps of algorithm for each job are repeated n times; therefore, it can be solved in $O(n^2)$. The sequence obtained by heuristic algorithm, H, is denoted as δ_h and its objective function is defined by $T_{\max}(h)$. $P_i[k]$ and $C_i[k]$ are the actual processing time and completion time of j_i in position k, respectively, which are obtained using relations (3) and (4):

$$C_i[k] = \begin{cases} S + a_i & \text{if} \quad S \leq y_1 \\ a_i + (1 + b_i)S - b_i y_1 & \text{if} \quad y_1 < S < y_2 \\ a_i + b_i(y_2 - y_1) + S & \text{if} \quad y_2 \leq S \end{cases} \tag{3}$$

$$P_i[k] = C_i[k] - S. \tag{4}$$

In this algorithm, choosing jobs for scheduling is totally based on the shortest processing times (a_i), the largest deterioration rates (b_i), and the shortest due dates (d_i). In Steps 1 and 2, all the jobs are considered and eligible ones are located at the beginning of sequence and are deleted from non-scheduled jobs set. Through Steps 3-11, if the completion time of the last scheduled job (S) is before y_1, set A is formed containing jobs like j_i in which:

$$a_i \leq \bar{a},\quad b_i \leq \bar{b},\quad d_i \leq \bar{d},$$

where:

$$\bar{a} = \sum_{i=1}^{n} a_i \Big/ n,\ \bar{b} = \sum_{i=1}^{n} b_i \Big/ n,\ \text{and}\ \ \bar{d} = \sum_{i=1}^{n} d_i \Big/ n.$$

If set A is empty or S is between y_1 and y_2, eligible jobs are chosen from the set of unscheduled jobs according to Steps 12 and 14. In Step 16, if S is greater than y_2, then non-scheduled jobs are arranged based on the EDD rule and scheduled at the end of scheduled jobs. The steps of algorithm H are as follows:

- **Step 0.** Start. Set $k = 1$, $L = \{j_1, j_2, ..., j_n\}$, $A = \phi$, $S = 0$, $T_{\max} = 0$.
- **Step 1.** If:

$$j_i \in L,$$

$$a_i = \min_{j\in L}\{a_j\}, \qquad b_i = \max_{j\in L}\{b_j\},$$

$$d_i = \min_{j\in L}\{d_j\} \qquad S + a_i \le y_1,$$

then go to Step 2; else, go to Step 3.
- **Step 2.** Schedule j_i in position k and calculate $C_i[k]$. Set $S = C_i[k]$, $L = L - j_i$, and $k = k+1$. If $k = n+1$, then go to Step 16; else, go to Step 1.
- **Step 3.** Calculate:

$$\overline{a} = \sum_{i=1}^{n} a_i/n, \quad \overline{b} = \sum_{i=1}^{n} b_i/n$$

$$\text{and} \quad \overline{d} = \sum_{i=1}^{n} d_i/n.$$

- **Step 4.** If $S < y_1$, then set A is updated based on the following condition:

$$\text{For} \quad j_i \in L:$$

$$\text{if} \quad a_i \le \overline{a}, \ \ b_i \ge \overline{b}, \ \text{ and } \ d_i \le \overline{d},$$

then

$$A = A + j_i.$$

- **Step 5.** If $A = \phi$, go to Step 6; else, go to Step 9.
- **Step 6.** If $S < y_1$, then set A is updated based on the following condition:

$$\text{For} \quad j_i \in L:$$

$$\text{if} \quad a_i \le \overline{a} \quad \text{and} \quad b_i \ge \overline{b}$$

then

$$A = A + j_i.$$

If $A = \phi$, go to Step 7; else, go to Step 9.
- **Step 7.** If $S < y_1$, then set A is updated based on the following condition:

$$\text{For} \quad j_i \in L:$$

$$\text{if} \quad a_i \le \overline{a} \quad \text{and} \quad d_i \le \overline{d}$$

then

$$A = A + j_j$$

If $A = \phi$, go to Step 8; else, go to Step 9.
- **Step 8.** If $S < y_1$, then set A is updated based on the following condition:

$$\text{For} \quad j_i \in L:$$

$$\text{if} \quad b_i \ge \overline{b} \quad \text{and} \quad d_i \le \overline{d}$$

then

$$A = A + j_i.$$

If $A = \phi$, go to Step 10; else, go to Step 9.
- **Step 9.** If $A = \phi$ and $S < y_1$, then go to Step 6. If $S \ge y_1$, go to Step 14; else, choose j_i with the smallest a_i from A and schedule it in the kth position. Set $S = C_i[k]$, $L = L - j_i$, $A = A - j_i$, and $k = k + 1$. If $k = n + 1$, then go to Step 16; else, repeat Step 9.
- **Step 10.** If $S \ge y_1$, then go to Step 14; else, for all the jobs $j_i \in L$, if $d_i \le \bar{d}$, then set $A = A + j_i$.
- **Step 11.** If $A = \phi$, then go to Step 12; else, choose j_i with the smallest $\frac{a_i}{b_i}$ and the largest b_i from A, set $A = A - j_i$ and go to Step 13. If there is no such a job, then choose j_i with the largest b_i from A, set $A = A - j_i$, and go to Step 13.
- **Step 12.** Choose j_i with the smallest $\frac{a_i}{b_i}$ and largest b_i from L and go to Step 13. If there is no such a job, then choose j_i with the largest b_i from L and go to Step 13.
- **Step 13.** Schedule j_i in position k. Set $S = C_i[k]$, $L = L - j_i$, and $k = k + 1$. If $k = n + 1$, then go to Step 16; if $S < y_1$, then go to Step 11; else, go to Step 14.
- **Step 14.** If $S \ge y_2$, then go to Step 15; else, choose $j_i \in L$ with one of the following conditions and go to Step 13:
  - j_i has the smallest a_i/b_i and smallest d_i;
  - j_i has the largest b_i and smallest d_i;
  - j_i has the smallest a_i/b_i and largest b_i;
  - j_i has the smallest a_i and smallest d_i;
  - j_i has the smallest b_i and smallest d_i;
  - j_i has the smallest a_i;
  - j_i has the largest b_i.
- **Step 15.** Sequence unscheduled jobs by the EDD

rule from k to n.

1. **Step 16.** Denote the final sequence as δ_h and its objective function as $T_{\max}(h)$, respectively.

4. B&B algorithm

In this section, a B&B algorithm using the back-tracking strategy is proposed to search for the optimal solution where upper bound, lower bounds, and dominance rules are used in an efficient manner. At first, we establish several dominance rules to fathom the searching tree, and then present a property to determine the ordering of remaining jobs (set δ'). In addition, three lower bounds are provided in Subsection 4.3. In the proposed B&B algorithm, when a job from set δ' is selected for scheduling, its involvement in set δ is checked by dominance rules and lower bounds. If it is not fathomed, then it will be added to the end of set δ.

4.1. Upper bound

In this paper, heuristic algorithm H is considered as the upper bound of problem and its final sequence (δ_h) will be a basis for generating the searching tree.

4.2. Dominance rules

Dominance rules are important in solving the scheduling problems. In this subsection, some dominance properties are given to be employed in the B&B algorithm. It is assumed that partial sequence, δ, with completion time, S, and maximum tardiness, $T_{\max}(\delta)$, is in hand. If j_i is processed immediately after partial sequence, δ, then the resulted sequence is shown as δi and if j_j is located after δi, then the sequence will be shown by δij. Notably, partial sequence δji is the result of pairwise interchange of j_i and j_j in partial sequence δij. To show that δij dominates δji, one should prove that $T_{\max}(\delta ij) \leq T_{\max}(\delta ji)$ and $C_j(\delta ij) \leq C_i(\delta ji)$.

Completion times of j_i and j_j in partial sequence δij are as follows:

$$C_i(\delta ij) = \begin{cases} S + a_i & \text{if} \quad S \leq y_1 \\ a_i + (1+b_i)S - b_i y_1 & \text{if} \quad y_1 < S < y_2 \\ a_i + b_i(y_2 - y_1) + S & \text{if} \quad y_2 \leq S \end{cases} \tag{5}$$

$$C_j(\delta ij) =$$

$$\begin{cases} S + a_j + a_i & \text{if} \quad C_i(\delta ij) \leq y_1 \\ a_j + (1+b_j)C_i(\delta ij) - b_j y_1 & \text{if} \quad y_1 < C_i(\delta ij) < y_2 \\ a_j + b_j(y_2 - y_1) + C_i(\delta ij) & \text{if} \quad y_2 \leq C_i(\delta ij) \end{cases} \tag{6}$$

Lemma 1. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, the jobs completed before y_1 will be arranged by the EDD rule.

Proof. The actual processing time of jobs completed before y_1 is constant. Hence, the problem would be like the basic form $1|T_{\max}$ where sequence based on the EDD rule is optimal; therefore, completed jobs before y_1 will be arranged by the EDD rule.

Lemma 2. In problem $1|P_i = a_i + b_i(S - y_1),\ y_1 > 0,\ y_2 > y_1|T_{\max}$, if $S > y_1$ and the following relations hold, then there exists an optimal sequence in which j_i must be processed before j_j:

$$y_2 \geq a_i + S(1+b_i) - b_i y_1, \tag{7}$$

$$y_2 \geq a_j + S(1+b_j) - b_j y_1, \tag{8}$$

$$a_i + a_j + a_i b_j + (S - y_1)(b_i + b_j + b_i b_j) + S - d_j$$
$$\geq T_{\max}(\delta), \tag{9}$$

$$a_j + a_i b_j + (S - y_1)(b_j + b_i b_j) + d_i \geq d_j, \tag{10}$$

$$a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S - d_i$$
$$\geq T_{\max}(\delta), \tag{11}$$

$$a_i + a_j b_i + (S - y_1)(b_i + b_i b_j) + d_j \geq d_i, \tag{12}$$

$$d_j \geq a_i b_j - a_j b_i + d_i, \tag{13}$$

$$a_j/b_j \geq a_i/b_i. \tag{14}$$

Proof. Since $S > y_1$, $y_2 \geq a_i + S(1+b_i) - b_i y_1$, and $y_2 \geq a_j + S(1+b_j) - b_j y_1$, the completion times of j_i and j_j in δij and δji are as follows:

$$C_i(\delta ij) = a_i + (1+b_i)S - b_i y_1, \tag{15}$$

$$C_j(\delta ij) = a_i + a_j + a_i b_j + (S - y_1)(b_i + b_j + b_i b_j) + S, \tag{16}$$

$$C_j(\delta ji) = a_j + (1+b_j)S - b_j y_1, \tag{17}$$

$$C_i(\delta ji) = a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S. \tag{18}$$

From $a_j/b_j \geq a_i/b_i$, it implies that:

$$C_i(\delta ji) \geq C_j(\delta ij). \tag{19}$$

According to the definition of tardiness and maximum tardiness, we have the following relations:

$$T_i(\delta ij) = \max\{0, a_i + (1+b_i)S - b_i y_1 - d_i\}, \tag{20}$$

$$T_j(\delta ij) = \max\{0, a_i + a_j + a_i b_j$$
$$+ (S - y_1)(b_i + b_j + b_i b_j) + S - d_j\}, \tag{21}$$

$$T_j(\delta ji) = \max\{0, a_j + (1+b_j)S - b_j y_1 - d_j\}, \tag{22}$$

$$T_i(\delta ji) = \max\{0, a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S - d_i\}, \quad (23)$$

$$T_{\max}(\delta ij) = \max\{T_{\max}(\delta), T_i(\delta ij), T_j(\delta ij)\}, \quad (24)$$

$$T_{\max}(\delta ji) = \max\{T_{\max}(\delta), T_i(\delta ji), T_j(\delta ji)\}. \quad (25)$$

Due to Relations (9), (10), and (24), the following relation is satisfied:

$$T_{\max}(\delta ij) = T_j(\delta ij). \quad (26)$$

Also, Relations (11), (12), and (25) express that the following relation is valid:

$$T_{\max}(\delta ji) = T_i(\delta ji). \quad (27)$$

Relation (13) shows that $T_i(\delta ji) \geq T_j(\delta ij)$. So, we have the following relation:

$$T_{\max}(\delta ji) \geq T_{\max}(\delta ij). \quad (28)$$

Based on Relations (19) and (28), we can confirm that sequence δij dominates sequence δji, and thus the proof is completed.

Lemma 3. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, if $S > y_1$ and the following relations hold, then there exists an optimal sequence in which j_i must be processed before j_j:

$$y_2 \geq a_i + S(1 + b_i) - b_i y_1, \quad (29)$$

$$y_2 \geq a_j + S(1 + b_j) - b_j y_1, \quad (30)$$

$$T_{\max}(\delta) \geq a_i + S(1 + b_i) - b_i y_1 - d_i, \quad (31)$$

$$T_{\max}(\delta) \geq a_i + a_j + a_i b_j + (S - y_1)(b_i + b_j + b_i b_j) + S - d_j, \quad (32)$$

$$T_{\max}(\delta) \geq a_j + S(1 + b_j) - b_j y_1 - d_j, \quad (33)$$

$$T_{\max}(\delta) \geq a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S - d_i, \quad (34)$$

$$a_j/b_j \geq a_i/b_i. \quad (35)$$

Lemma 4. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, if $S > y_1$ and the following relations hold, then there exists an optimal sequence in which j_i must be processed before j_j:

$$y_2 \geq a_i + S(1 + b_i) - b_i y_1, \quad (36)$$

$$y_2 \geq a_j + S(1 + b_j) - b_j y_1, \quad (37)$$

$$T_{\max}(\delta) \geq a_i + S(1 + b_i) - b_i y_1 - d_i, \quad (38)$$

$$T_{\max}(\delta) \geq a_i + a_j + a_i b_j + (S - y_1)(b_i + b_j + b_i b_j) + S - d_j, \quad (39)$$

$$a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S - d_i \geq T_{\max}(\delta), \quad (40)$$

$$a_i + a_j b_i + (S - y_1)(b_i + b_i b_j) + d_j \geq d_i, \quad (41)$$

$$a_j/b_j \geq a_i/b_i. \quad (42)$$

Lemma 5. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, if $S > y_1$ and the following relations hold, then there exists an optimal sequence in which j_i must be processed before j_j:

$$y_2 \geq a_i + S(1 + b_i) - b_i y_1, \quad (43)$$

$$y_2 \geq a_j + S(1 + b_j) - b_j y_1, \quad (44)$$

$$a_i + S(1 + b_i) - b_i y_1 - d_i \geq T_{\max}(\delta), \quad (45)$$

$$d_j \geq a_j + a_i b_j + (S - y_1)(b_j + b_i b_j) + d_i, \quad (46)$$

$$a_i + a_j + a_j b_i + (S - y_1)(b_i + b_j + b_i b_j) + S - d_i \geq T_{\max}(\delta), \quad (47)$$

$$a_i + a_j b_i + (S - y_1)(b_i + b_i b_j) + d_j \geq d_i, \quad (48)$$

$$a_j/b_j \geq a_i/b_i. \quad (49)$$

Notably, the proofs of Lemmas 3 to 5 are omitted since they are similar to that of Lemma 2. However, they are available upon the request of the interested readers. We present Lemma 6 to determine the ordering of jobs in set δ' and to further speed up searching process.

Lemma 6. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, if $S \geq y_2$, then the optimal sequence after y_2 will be obtained by using EDD rule on set δ' as follows.

Proof. After y_2, actual processing time of jobs in set δ' is known and constant so that the problem is equivalent to problem $1||T_{\max}$ in which optimal sequence is obtained via EDD rule. So, jobs started after y_2 should be arranged by EDD rule.

Lemma 7. In problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|\sum_{i=1}^{n} U_i$, if there exists j_i so that relations

$a_i = \min_{j\in\delta'}\{a_j\}$, $b_i = \max_{j\in\delta'}\{b_j\}$, $d_i = \min_{j\in\delta'}\{d_j\}$, and $S + a_i \leq y_1$ hold, then there always exists an optimal sequence in which j_i is scheduled at time S.

Proof. As j_i has the least normal process time, it will have shortest completion time and starting and processing times of the following jobs will become shorter. Also, the selection of j_i makes the job with the largest deterioration rate be scheduled in a condition that there would not be any deterioration for it. It would make the next jobs have less deterioration and the shortest completion time. On the other hand, j_i has the least due date; so, scheduling it on time S will not increase the maximum tardiness.

As a result, employing the above lemma at the start of each algorithm may lead to scheduling the jobs at the beginning and omit them from set N whose search space of problem is reduced. In addition, implementing the lemma in depth search process of B&B algorithm leads to the selection of the best job from set δ'; so, there is no need to search the other branches.

4.3. Lower bounds

Lower bounds can further enhance the efficiency of B&B algorithm. In each node, the objective function of partial sequence δ is $T_{\max}(\delta)$ and its lower bound is shown by LB^*. In order to obtain LB^*, the following theorems are presented.

Theorem 1. In partial sequence δ for problem $1|P_i = a_i+b_i(S-y_1), y_1 > 0, y_2 > y_1|T_{\max}$, if all the jobs in set δ' are scheduled one by one at time S, then lower bound LB_1 is obtained according to the following relation:

$$LB_1 = \max\{T_{\max}(\delta), \max_{\forall i\in\delta'}\{0, C_i(\delta i) - d_i\}\}. \tag{50}$$

Proof. Obviously, any different sequence of jobs in set δ' will have no effect on $T_{\max}(\delta)$. Also, scheduling a given job in set δ' at time S leads to one of the following two cases for that job. If the job becomes tardy at time S, then its tardiness will increase after S; in contrast, if the job does not become tardy at time S, then its tardiness will not decrease after S. Hence, maximum tardiness of set δ' will never be less than $\max_{\forall i\in\delta'}\{0, C_i(\delta i) - d_i\}$. Therefore, the objective function of each complete sequence would not be less than LB_1.

Theorem 2. In partial sequence δ for problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, lower bound LB_2 is calculated as follows:

$$LB_2 = \max\{T_{\max}(\delta), T_{\max}(\delta'_{EDD})\}, \tag{51}$$

where $T_{\max}(\delta'_{EDD})$ is obtained according to EDD rule for the jobs in set δ' assuming deterioration at time S for each job.

Proof. Any arbitrary sequence of jobs in set δ' will have no impact on $T_{\max}(\delta)$. Furthermore, it is apparent that the actual processing time and completion time of each job assuming the deterioration at time S is not higher than the real deterioration. Since the maximum tardiness in basic form $1||T_{\max}$ is optimized via EDD rule, by relaxing the assumption of real deterioration and using the deterioration at time S for all the jobs in set δ', the maximum tardiness will never be less than $T_{\max}(\delta'_{EDD})$. Hence, the objective function of each complete sequence will not be less than LB_2.

Theorem 3. In partial sequence δ for problem $1|P_i = a_i + b_i(S - y_1), y_1 > 0, y_2 > y_1|T_{\max}$, lower bound LB_3 is calculated by the following relation:

$$LB_3 = \max\{T_{\max}(\delta), T_{\max}(\delta'_{LB})\}, \tag{52}$$

where $T_{\max}(\delta'_{LB})$ is calculated by algorithm LB.

Algorithm *LB*

- **Step 0.** Set $T_{\max} = 0$, $C = S$, $M = 1$, $k = 1$, $\delta' = \{j_1, j_2, ..., j_u\}$, $N_T(\delta') = 0$, $B_{\delta'} = \{b^{\delta'}_{[1]}, b^{\delta'}_{[2]}, ..., b^{\delta'}_{[u]}\}$, and $D_{\delta'} = \{d^{\delta'}_{[1]}, d^{\delta'}_{[2]}, ..., d^{\delta'}_{[u]}\}$, such that $b^{\delta'}_{[1]} \leq b^{\delta'}_{[2]} \leq ... \leq b^{\delta'}_{[u]}$ and $d^{\delta'}_{[1]} \leq d^{\delta'}_{[2]} \leq ... \leq d^{\delta'}_{[u]}\}$ where u is the number of jobs in set δ'.
- **Step 1.** Choose a job with the least a_j from set δ' and schedule it at time C. If $C \leq y_1$, then set $C = C + a_j$, $M = M + 1$, $\delta' = \delta' - j_j$ and go to Step 2; else, select M deterioration rates from set $D_{\delta'}$ and assign them to M scheduled jobs at the end of sequence in a non-increasing order; then, calculate their completion times. Set $C = C_j$, $M = M + 1$, and $\delta' = \delta' - j_j$.
- **Step 2.** If $C - d_{[k]} \leq T_{\max}$, then $k = k + 1$ and go to step 3; else, set $T_{\max} = C - d_{[k]}$, $k = k + 1$ and go to Step 1.
- **Step 3.** If $k \leq u$, then go to Step 1; else, set $T_{\max}(\delta'_{LB}) = T_{\max}$.

Proof. The proof of Theorem 3 is presented in the Appendix.

The lower bound for every node of B&B tree is calculated by the following relation:

$$LB^* = \max\{LB_1, LB_2, LB_3\}. \tag{53}$$

5. Computational experiments

In this section, a set of random generated test problems is considered in order to evaluate the performance of B&B and heuristic algorithms. The test problems were solved on a Pentium 4 PC with 2.53 GHz CPU and 3G RAM under Windows XP. In the following subsections, the problem generation procedure and analysis of the results are described. Notably, the intervals of uniform distribution for the test problem parameters are similar to those of [3].

5.1. Test problems

The normal processing times (a_i) and deterioration rates (b_i) are randomly generated from discrete uniform distribution over $(0, 10]$ and continuous uniform distribution over $(0, 1]$, respectively. y_1 and y_2 are also generated from continuous uniform distributions over intervals $[0, A/3]$ and $[0, 2A/3]$ for y_1 and intervals $[A/3, 2A/3]$ and $[2A/3, A]$ for y_2, where A is assumed to be determined as in:

$$A = \sum_{i=1}^{n} a_i.$$

As $y_2 > y_1$, in order to generate values of y_1 and y_2, three distinctive conditions for different combinations of y_1 and y_2 would be logical. Due dates were also generated randomly from continuous uniform distributions over $(0, 0.5C_{\max}]$, $[0.5C_{\max}, C_{\max}]$, $(0, C_{\max}]$, and $(0, 1.5C_{\max}]$ where $C_{\max}$ is makespan of the obtained sequence based on the non-decreasing ratio of a_i/b_i. For the number of jobs (n), values of 8, 12, 16, 20, 24, 28, 32, 36, 40, and 44 were utilized [3]. Considering the intervals of y_1, y_2, and due date, 12 groups, S_{111} to S_{224}, were formed whose definitions and specifications are briefly given in Table 3. For any possible combination of S_{111} to S_{224} and n, 20 test problems were randomly generated. Accordingly, 2400 (i.e., $12 \times 10 \times 20$) sample problems were generated and solved totally.

5.2. Computational results

Heuristic procedure H and B&B algorithm were coded in C++ and sample problems were solved. In our B&B method, a time limit equal to 4000 seconds for each problem was considered; if a problem does not get the optimal solution in this limitation, then B&B procedure will be stopped. In Table 4, computational results for 12 groups of problems are presented. As observed in Table 4, the optimal solution is achieved for all the sample problems with at least 32 jobs. Moreover, some sample problems with more jobs are also solved.

In order to study the performance of heuristic approach, the error percentages based on the following equation are recorded:

Table 3. Specifications of the different groups of problems.

No	Range of deterioration function variables: y_2	y_1	Range of due dates
1	$[A/3, 2A/3]$	$[0, A/3]$	$(0, .5C_{\max}]$
2	$[A/3, 2A/3]$	$[0, A/3]$	$[.5C_{\max}, C_{\max}]$
3	$[A/3, 2A/3]$	$[0, A/3]$	$(0, C_{\max}]$
4	$[A/3, 2A/3]$	$[0, A/3]$	$(0, 1.5C_{\max}]$
5	$[2A/3, A]$	$[0, A/3]$	$(0, .5C_{\max}]$
6	$[2A/3, A]$	$[0, A/3]$	$[.5C_{\max}, C_{\max}]$
7	$[2A/3, A]$	$[0, A/3]$	$(0, C_{\max}]$
8	$[2A/3, A]$	$[0, A/3]$	$(0, 1.5C_{\max}]$
9	$[2A/3, A]$	$[0, 2A/3]$	$(0, .5C_{\max}]$
10	$[2A/3, A]$	$[0, 2A/3]$	$[.5C_{\max}, C_{\max}]$
11	$[2A/3, A]$	$[0, 2A/3]$	$(0, C_{\max}]$
12	$[2A/3, A]$	$[0, 2A/3]$	$(0, 1.5C_{\max}]$

$$\%\text{Error} = {(Z - Z^*)}/{Z^*} \times 100\ \%,$$

where Z and Z^* are $T_{\max}$ obtained from heuristic algorithm and optimal schedule, respectively.

In Table 4, average and maximum values of %Error are presented. The corresponding column shows that the average %Error is less than 0.3% which proves that the proposed heuristic algorithm is highly accurate; therefore, solving the large-scale problems is recommended. Notably, the computation time of heuristic algorithm is not recorded since it is almost finished in zero time.

As can be seen in Table 4, the performance of B&B algorithm is different for 12 groups; it significantly depends upon the values of due dates, y_1 and y_2. As the maximum tardiness in the problems with large due dates is lower than the maximum tardiness in those with short due dates, a great decrease in the number of nodes happens, which results in the easy problems. Generally, large values of y_1 cause a decrease in the completion times of jobs because jobs do not have any deterioration up to time y_1. Decreasing the completion times of jobs leads to an increase in the number of utilization of the dominance rules and lower bounds, especially LB_3; so, it makes the problems hard to solve. Large values of y_2 also bring about an increase in the quantity of employing the lemmas and theorems. According to Lemma 8, on the other hand, obtaining the optimal solution for small values of y_2 is easier than that for large values; hence, large values of y_2 make the problems difficult.

The results given in Table 4 indicate that the maximum job size, whose B&B method is able to solve, is 44 belonging to groups S_{114}, S_{124}, and S_{224}. Figure 2 demonstrates the minimum average of CPU times for

Table 4. Performance of B&B and heuristic algorithms.

Group	n	# of optimum samples		% Error		Avg CPU time of B&B (s)	%Avg of fathomed nodes by										%Avg of all fathomed nodes
		B&B	H	Avg	Max		Lem[a] 6	Lem 1	Lem 7	Lem 2	Lem 3	Lem 4	Lem 5	LB_3	LB_1	LB_2	
S_{111}	8	20	15	0.09	0.43	0.00	4.49	1.30	1.27	4.37	0.21	1.50	3.19	76.02	2.91	4.74	97.04
	12	20	12	0.11	0.56	0.02	8.63	1.26	2.61	2.13	2.97	1.77	2.64	65.88	4.70	7.41	97.89
	16	20	12	0.09	0.60	0.12	13.59	0.48	1.86	2.55	3.49	2.94	4.19	59.13	4.45	7.32	99.07
	20	20	10	0.05	0.34	1.42	14.45	2.62	4.12	1.95	2.92	1.62	2.30	58.07	5.14	6.81	91.81
	24	20	6	0.08	0.70	32.82	15.52	0.35	3.64	0.95	1.85	1.32	3.22	59.86	4.70	8.59	91.84
	28	20	7	0.05	0.12	293.03	12.41	0.32	4.37	1.12	1.65	3.03	2.01	63.67	6.88	4.53	95.81
	32	20	6	0.07	0.68	1072.02	16.82	0.47	5.41	2.25	8.49	2.08	2.55	50.59	5.18	6.17	92.12
	36	12	5	0.04	0.19	1000.25	11.21	0.58	1.37	1.61	2.81	1.43	3.14	66.13	6.18	5.54	94.27
S_{112}	8	20	16	0.05	0.35	0.00	10.98	1.00	3.06	0.76	0.32	0.99	0.78	76.53	1.72	3.85	98.66
	12	20	13	0.24	1.24	0.00	13.85	1.98	1.43	0.45	0.14	0.41	3.11	74.78	0.24	3.62	96.23
	16	20	12	0.13	0.59	0.04	11.64	3.01	2.17	1.33	0.01	0.75	0.59	76.66	1.66	2.17	88.26
	20	20	7	0.15	1.09	0.37	21.36	2.15	1.73	0.54	0.08	2.28	1.66	64.33	1.59	4.29	94.41
	24	20	8	0.11	0.98	2.64	30.54	1.51	0.74	1.14	0.81	1.65	1.36	59.76	1.18	1.30	90.52
	28	20	6	0.04	0.44	28.93	26.17	2.35	1.32	0.14	1.36	0.00	0.94	66.38	0.49	0.85	84.63
	32	20	4	0.03	0.37	100.04	34.24	5.39	4.51	0.09	0.21	0.05	1.48	51.83	0.09	2.11	95.72
	36	20	5	0.09	1.03	725.82	22.07	3.68	2.96	0.85	1.91	1.24	2.66	61.43	1.27	1.93	89.61
	40	17	4	0.06	0.36	2154.78	14.71	3.65	0.92	1.45	0.56	1.37	2.51	72.73	0.94	1.16	93.53
S_{113}	8	20	13	0.08	0.34	0.00	7.31	1.77	2.10	0.79	0.35	1.81	1.86	69.44	4.04	10.52	97.40
	12	20	18	0.01	0.12	0.01	8.50	1.52	1.16	1.21	1.45	4.86	2.84	66.33	4.40	7.73	96.06
	16	20	14	0.11	1.75	0.09	8.14	2.15	2.50	1.76	2.26	0.70	3.50	64.54	6.33	8.13	90.93
	20	20	9	0.29	4.67	0.54	21.05	2.23	5.19	0.53	2.31	0.55	2.70	55.06	5.59	4.80	92.54
	24	20	7	0.02	0.22	6.27	16.25	1.35	4.25	0.69	0.85	0.87	1.29	61.91	4.35	8.19	91.84
	28	20	4	0.07	1.16	45.18	25.24	4.21	6.12	1.16	1.20	2.25	3.41	46.61	5.12	4.68	93.35
	32	20	1	0.06	0.91	116.26	28.25	2.65	3.84	0.44	0.94	1.69	2.45	50.48	4.12	5.14	92.68
	36	20	2	0.04	0.24	650.33	24.38	1.24	4.45	1.96	2.21	1.78	2.84	50.75	6.55	3.84	94.74
	40	11	1	0.18	1.22	1327.30	18.96	1.09	3.75	1.66	1.95	1.09	2.85	59.80	4.43	4.42	95.60
S_{114}	8	20	18	0.04	0.62	0.00	2.45	0.84	1.25	0.65	0.12	0.26	1.26	86.54	4.24	2.38	91.24
	12	20	15	0.18	1.66	0.00	3.45	2.28	2.20	1.21	2.14	0.05	0.61	73.65	5.42	8.99	90.76
	16	20	10	0.06	0.48	0.11	8.65	1.24	0.65	0.23	0.72	0.42	1.45	76.78	3.11	6.75	90.14
	20	20	11	0.00	0.05	0.86	6.24	0.92	0.89	0.76	0.64	0.13	0.83	80.34	6.87	2.38	94.56
	24	20	6	0.03	0.12	1.27	10.65	0.00	1.40	1.62	1.14	0.42	1.14	76.84	4.24	2.55	93.37
	28	20	5	0.01	0.08	3.94	12.65	0.68	0.36	0.85	0.81	0.92	0.35	70.23	3.51	9.64	90.27
	32	20	3	0.02	0.11	15.13	6.84	1.21	1.02	1.32	0.41	0.19	0.65	78.21	2.54	7.61	89.77
	36	20	4	0.02	0.06	95.58	8.54	2.12	1.86	0.65	0.67	1.81	1.32	68.58	5.51	8.94	92.76
	40	20	1	0.00	0.06	435.41	7.79	3.24	2.47	0.34	1.42	0.68	1.55	72.12	3.61	6.78	94.78
	44	20	2	0.04	0.09	1650.96	7.12	1.04	0.95	0.00	0.89	0.44	0.59	77.34	4.42	7.21	94.24
S_{121}	8	20	16	0.04	0.49	0.00	2.24	2.43	2.35	1.34	1.47	2.93	1.62	74.86	4.21	6.55	94.43
	12	20	16	0.06	0.97	0.00	3.82	0.65	4.57	2.61	2.73	4.61	2.05	67.22	3.43	8.31	95.67
	16	20	11	0.05	0.35	1.12	8.56	2.54	3.49	2.02	2.12	2.14	1.68	62.37	4.34	10.74	86.34
	20	20	8	0.05	0.27	8.44	7.44	1.14	6.73	1.33	2.37	3.08	2.94	60.64	6.94	7.39	96.72
	24	20	4	0.00	0.04	35.82	9.32	1.36	8.41	3.76	1.43	2.59	2.57	59.44	4.30	6.82	95.07
	28	20	5	0.01	0.11	455.67	10.65	0.85	0.24	2.45	3.74	1.68	1.44	66.18	5.86	6.91	90.51
	32	20	3	0.02	0.21	2890.83	7.74	1.73	16.37	2.76	2.47	2.34	3.51	46.52	8.23	8.33	94.92
	36	16	1	0.00	0.01	1850.19	6.96	2.38	2.61	3.27	4.70	1.11	2.73	66.19	4.63	5.42	84.20
	8	20	15	0.08	0.74	0.00	3.29	2.41	1.81	0.04	1.45	0.98	0.57	88.58	0.41	0.46	92.42

[a]Lem: Lemma.

Table 4. Performance of B&B and heuristic algorithms (continued).

Group	n	# of optimum samples		% Error		Avg CPU time of B&B (s)	%Avg of fathomed nodes by										%Avg of all fathomed nodes
		B&B	H	Avg	Max		Lem[a] 6	Lem 1	Lem 7	Lem 2	Lem 3	Lem 4	Lem 5	LB_3	LB_1	LB_2	
	12	20	12	0.05	0.61	0.08	2.43	3.63	1.75	0.69	0.75	2.37	1.65	84.10	1.69	0.94	94.61
	16	20	13	0.12	1.05	0.48	6.74	1.54	2.59	0.88	1.70	1.67	2.37	80.49	0.75	1.27	96.45
	20	20	10	0.19	1.16	1.22	3.81	0.98	1.97	0.96	0.42	1.76	1.93	86.18	0.60	1.39	88.26
S_{122}	24	20	5	0.06	0.91	14.48	8.56	2.12	1.24	0.41	1.34	2.58	0.71	79.22	1.37	2.45	93.20
	28	20	5	0.11	1.35	110.33	8.12	3.58	1.48	0.65	0.97	0.64	0.95	81.36	0.00	2.25	92.42
	32	20	2	0.21	0.45	866.89	4.97	2.91	1.13	0.29	1.94	2.37	1.84	82.15	1.43	0.97	94.18
	36	20	3	0.09	0.81	2154.78	5.93	0.41	0.92	1.27	1.02	1.13	2.08	84.95	0.82	1.47	90.04
	8	20	16	0.06	0.90	0.00	6.35	0.96	0.24	1.21	2.51	1.55	1.20	75.18	4.60	6.20	94.18
	12	20	12	0.11	1.58	0.00	4.57	2.64	0.83	0.84	1.57	0.95	2.67	77.82	3.99	4.12	96.28
	16	20	13	0.18	2.13	0.03	5.14	1.43	0.92	1.70	2.63	1.82	1.13	77.08	2.19	5.96	90.59
	20	20	8	0.02	0.30	0.67	3.68	1.76	0.81	0.36	0.97	0.80	2.19	79.99	5.86	3.58	95.66
S_{123}	24	20	3	0.09	0.45	44.15	2.27	2.39	0.81	1.22	1.84	1.02	0.87	78.45	4.77	6.36	88.74
	28	20	3	0.03	0.06	180.55	5.69	3.72	0.60	1.95	1.22	0.93	2.55	74.04	3.51	5.79	93.62
	32	20	1	0.02	0.12	428.91	6.12	1.87	0.51	0.72	0.97	1.24	0.72	78.09	4.84	4.92	94.93
	36	20	2	0.08	0.10	712.25	4.71	2.44	0.88	2.42	1.41	2.79	0.42	71.48	5.26	8.19	89.67
	40	12	2	0.04	0.08	950.56	6.91	3.26	0.69	0.63	1.22	1.75	1.94	75.19	4.69	3.72	92.25
	8	20	16	0.02	0.11	0.00	3.47	1.32	0.69	0.98	1.41	1.27	0.94	80.05	3.11	6.76	90.99
	12	20	15	0.03	0.16	0.00	4.66	1.45	0.46	1.38	1.18	2.84	0.65	74.44	2.56	10.38	94.24
	16	20	13	0.04	0.12	0.00	3.19	2.34	0.04	1.75	0.94	0.73	1.28	77.06	3.96	8.71	92.35
	20	20	9	0.00	0.01	0.02	4.21	1.54	0.42	0.84	0.00	0.49	0.78	80.45	4.81	6.46	96.03
S_{124}	24	20	11	0.00	0.02	0.09	7.96	2.34	0.00	2.76	1.09	1.65	2.34	69.69	4.62	7.55	88.27
	28	20	7	0.00	0.00	6.94	2.35	1.84	0.09	0.54	1.54	0.49	0.71	83.36	5.89	3.19	93.14
	32	20	5	0.00	0.03	34.43	8.58	0.97	0.01	1.89	0.43	1.38	1.22	78.55	4.21	2.76	87.42
	36	20	5	0.01	0.08	147.81	6.41	1.23	0.00	1.19	1.89	0.91	1.07	71.99	5.37	9.94	92.86
	40	20	6	0.00	0.04	675.92	7.80	2.43	0.87	0.97	2.01	1.00	2.63	73.93	4.54	3.82	96.24
	44	14	2	0.01	0.12	1348.76	8.28	1.89	0.93	1.65	2.18	1.11	2.78	76.16	2.90	2.12	93.64
	8	20	14	0.09	1.44	0.00	5.44	2.65	0.75	1.94	2.66	1.35	2.58	75.32	4.93	2.38	93.72
	12	20	10	0.04	0.63	0.05	6.17	4.78	1.32	1.47	0.91	2.51	3.24	64.06	6.55	8.99	97.39
	16	20	11	0.07	1.09	0.67	3.30	2.40	1.54	2.05	1.23	0.97	2.85	73.67	5.24	6.75	92.03
	20	20	5	0.08	1.39	4.66	3.86	3.13	0.68	2.69	2.12	6.62	2.62	66.46	4.93	6.90	88.55
S_{221}	24	20	9	0.04	0.65	25.72	9.64	2.12	2.95	1.27	1.46	3.06	3.20	55.74	8.63	11.93	97.23
	28	20	6	0.12	0.21	185.93	4.79	8.89	1.49	1.38	0.00	1.28	2.77	66.03	3.18	10.18	88.58
	32	20	4	0.01	0.04	620.55	5.29	4.72	0.70	2.53	2.75	2.40	6.89	57.26	5.58	11.88	93.08
	36	18	3	0.01	0.18	1497.07	5.94	6.55	3.88	1.40	2.08	2.58	2.50	60.35	4.30	10.43	84.56
	40	9	3	0.00	0.01	675.66	2.25	7.63	4.27	2.72	1.67	3.67	1.65	66.27	3.45	6.42	92.61
	8	20	12	0.12	1.48	0.00	1.45	1.92	2.85	0.82	0.96	1.16	0.89	83.44	5.36	1.15	98.76
	12	20	10	0.09	1.36	0.00	5.81	4.68	2.48	1.04	1.63	2.42	0.64	72.20	3.14	5.96	94.92
	16	20	7	0.06	0.97	0.21	6.91	3.61	3.47	1.61	1.55	1.45	1.60	69.68	7.31	2.81	94.81
S_{222}	20	20	8	0.11	1.72	3.51	4.34	7.36	0.92	0.94	2.84	1.59	2.32	75.73	1.92	2.04	96.57
	24	20	5	0.14	0.98	68.35	8.57	4.71	1.55	1.21	0.60	1.71	2.60	69.64	2.74	6.67	89.29
	28	20	2	0.16	1.50	1243.76	2.70	3.12	2.67	0.74	1.54	0.54	1.77	80.20	3.55	3.17	90.67
	32	20	3	0.11	0.76	2875.88	6.32	4.07	5.09	1.35	1.01	1.92	1.53	68.12	5.14	5.45	88.25
	36	10	0	0.09	0.56	1562.93	4.18	3.71	2.42	1.55	0.96	1.38	2.50	74.47	4.65	4.18	91.73
	8	20	13	0.00	0.03	0.00	3.47	2.47	1.27	0.88	1.33	1.34	1.38	74.60	7.94	5.32	94.48
	12	20	14	0.13	0.90	0.01	7.96	3.54	1.46	0.73	4.34	1.05	2.80	65.29	10.23	2.59	96.85

[a]Lem: Lemma.

Table 4. Performance of B&B and heuristic algorithms (continued).

Group	n	# of optimum samples		% Error		Avg CPU time of B&B (s)	%Avg of fathomed nodes by										%Avg of all fathomed nodes
		B&B	H	Avg	Max		Lem[a] 6	Lem 1	Lem 7	Lem 2	Lem 3	Lem 4	Lem 5	LB_3	LB_1	LB_2	
S_{223}	16	20	8	0.09	1.03	0.53	5.27	1.38	0.58	1.31	1.10	1.48	1.32	77.08	3.78	6.69	94.46
	20	20	4	0.06	0.61	3.98	4.52	4.10	2.73	0.00	2.75	2.00	0.93	69.20	6.65	7.11	90.13
	24	20	3	0.00	0.02	84.62	5.29	6.21	1.29	0.61	4.23	0.95	2.03	68.73	8.39	2.27	85.89
	28	20	5	0.16	0.86	198.94	6.15	5.97	2.11	1.54	3.12	2.63	2.83	63.09	3.64	8.92	95.31
	32	20	0	0.08	0.34	822.50	3.88	2.02	0.93	0.84	2.68	3.54	3.20	74.96	4.50	3.45	91.43
	36	15	2	0.03	0.28	1100.50	4.24	2.96	0.67	0.62	3.71	2.01	1.79	74.86	4.96	4.18	93.59
S_{224}	8	20	17	0.02	0.38	0.00	6.10	1.25	1.07	0.84	0.99	2.38	2.29	78.34	5.62	1.12	98.86
	12	20	11	0.01	0.09	0.07	4.22	2.55	0.93	1.23	1.16	1.57	1.41	76.37	5.89	4.67	89.21
	16	20	13	0.00	0.05	0.15	3.67	1.47	2.49	0.66	1.64	1.16	0.00	83.05	3.45	2.41	96.58
	20	20	9	0.00	0.01	1.45	7.51	4.18	2.13	1.05	1.28	3.83	2.40	68.48	2.90	6.24	93.78
	24	20	5	0.00	0.01	5.63	4.13	3.09	1.37	0.76	0.86	1.40	5.08	73.07	3.69	6.55	93.07
	28	20	7	0.00	0.00	3.49	10.27	7.62	2.55	1.47	0.89	2.07	0.95	64.06	6.22	3.90	84.96
	32	20	4	0.00	0.00	2.28	7.38	3.48	0.78	2.04	3.05	0.95	2.67	71.18	4.31	4.16	85.58
	36	20	1	0.00	0.02	146.10	4.80	5.67	1.63	0.66	2.01	0.65	1.22	74.38	6.93	2.05	91.45
	40	20	1	0.01	0.11	910.59	6.58	2.19	1.79	0.43	1.39	2.93	0.67	78.20	4.21	1.61	88.78
	44	20	3	0.00	0.02	1985.68	5.16	1.49	2.02	1.23	1.60	2.62	1.84	77.53	4.64	1.87	93.63

[a]Lem: Lemma.

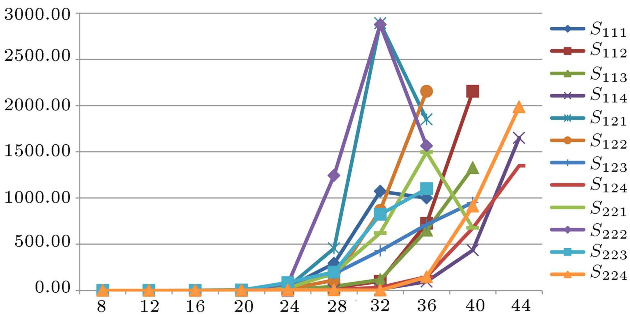

Figure 2. Average CPU time of B&B algorithm.

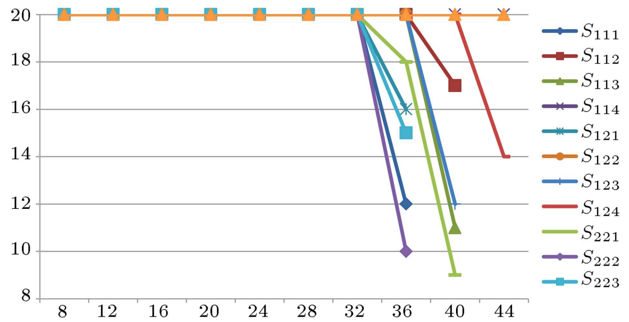

Figure 3. Number of optimal samples obtained by B&B algorithm.

solving the three groups in which the values of due date are generated over the wide interval $(0, 1.5C_{\max}]$, although values of y_1 and y_2 in S_{224} are large. Also, groups S_{111}, S_{121}, and S_{221} have large CPU times and small solved job sizes; due dates of those groups were obtained over small range $(0, 0.5C_{\max}]$. In addition, because of the longest interval of y_1 and y_2 values in groups S_{221} and S_{223}, the average CPU times of solving the generated hard problems are rather high. Since y_1 and y_2 in group S_{222} were generated over the longest interval and due dates were obtained over the short interval $(0.5C_{\max}, C_{\max}]$, this group is strongly hard to solve; the maximum CPU time and minimum number of optimal samples belong to this group as given in Table 4 and Figure 3.

In Table 4, the efficiency of all the lemmas and lower bounds is also demonstrated by the average percentage of fathomed nodes presented according to the order of accomplishment in B&B algorithm. Due to having the shortest interval of y_2 over $[A/3, 2A/3]$ in the groups S_{111}, S_{112}, and S_{113}, the number of utilizing Lemma 6 is increased and the performance of this lemma is great. Also, in groups S_{221}, S_{222}, S_{223}, and S_{224}, where y_1 is obtained from the longest interval $[0, 2A/3]$, Lemma 1 is highly efficient.

According to Table 4, the efficiency of LB_3 is so excellent in all the groups; in many problems, it fathoms the initial nodes of B&B tree so that the numerous branches of searching tree, and thus a great percentage of entire nodes are omitted. As it can be seen, due to having large due dates over intervals $(0.5C_{\max}, C_{\max}]$ and $(0, 1.5C_{\max}]$ in S_{112}, S_{114}, S_{122}, S_{124}, S_{222}, and S_{224}, the efficiency of LB_1 and LB_2 is decreased. The average percentage of fathomed nodes is at least 84% which proves a fantastic performance of the proposed B&B method.

6. Conclusion and future research

In this paper, the single-machine scheduling problem under piecewise linear deteriorating jobs was investigated whose objective is to minimize the maximum tardiness. It was assumed that the processing time of jobs is an increasing function of their starting time according to a piecewise linear function. The problem is known to be NP-hard; therefore, a B&B algorithm with several dominance rules and lower bounds was established to solve the problem optimally. A heuristic method was also proposed to derive the near-optimal solutions. The experimental results showed a high performance of the proposed B&B algorithm as it could solve the problems with at least 32 jobs in 12 different groups. Furthermore, it was shown that the average percentage error of heuristic approach is less than 0.3% which demonstrates its great capabilities to solve the large-scale problems. Scheduling problem under deteriorating jobs is an interesting topic for research studies. The future studies may focus on multiple machines or the other objective functions. Furthermore, some practical assumptions, such as the machine availability constraint or release times, might be added. Also, the other types of deterioration function, such as the exponential form with the assumption of learning or forgetting effects, can be investigated.

References

1. Jafari, A. and Moslehi, G. "Scheduling linear deteriorating jobs to minimize the number of tardy jobs", *Journal of Global Optimization*, **54**(2), pp. 389-404 (2012).
2. Lee, WC. and Lu, Z.S. "Group scheduling with deteriorating jobs to minimize the total weighted number of late jobs", *Applied Mathematics and Computation*, **218**(17), pp. 8750-8757 (2012).
3. Moslehi, G. and Jafari, A. "Minimizing the number of tardy jobs under piecewise-linear deterioration", *Computers & Industrial Engineering*, **59**(4), pp. 573-584 (2010).
4. Pinedo, M., Scheduling: *Theory, Algorithms, and Systems*, 3th Ed., Upper Saddle River, Prentice Hall (2002).
5. Yin, Y., Cheng, T.C.E. and Wu, C.C. "Scheduling with time-dependent processing times 2015", *Mathematical Problems in Engineering*, Article ID 367585, 2 pages (2015).
6. Yin, Y., Cheng, T.C.E. and Wu, C.C. "Scheduling with time-dependent processing times", *Mathematical Problems in Engineering*, Article ID 201421, 2 pages (2014).
7. Alidaee, B. and Womer, N.K. "Scheduling with time dependent processing times: Review and extensions", *Journal of the Operational Research Society*, **50**(7), pp. 711-720 (1999).
8. Cheng, T.C.E., Ding, Q. and Lin, B.M.T. "A concise survey of scheduling with time-dependent processing times", *European Journal of Operational Research*, **152**(1), pp. 1-13 (2004).
9. Browne, S. and Yechiali, U. "Scheduling deteriorating jobs on a single processor", *Computers & Operations Research*, **38**(3), pp. 495-498 (1990).
10. Bachman, A. and Janiak, A. "Minimizing maximum lateness under linear deterioration theory and methodology", *European Journal of Operational Research*, **126**(3), pp. 557-566 (2000).
11. Hsu, Y.S. and Lin, B.M.T. "Minimization of maximum lateness under linear deterioration", *Omega*, **31**(6), pp. 459-469 (2003).
12. Ng, C.T., Wang, J.B., Cheng, T.C.E. and Liu, L.L. "A branch-and-bound algorithm for solving a two-machine flow shop problem with deteriorating jobs", *Computers & Operation Research*, **37**(1), pp. 83-90 (2010).
13. Lee, W.C., Yeh, W.C. and Chung, Y.H. "Total tardiness minimization in permutation flowshop with deterioration consideration", *Applied Mathematical Modelling*, **38**(13), pp. 3081-3092 (2014).
14. Yin, Y., Wang, Y., Cheng, T.C.E., Liu, W. and Li, J. "Parallel-machine scheduling of deteriorating jobs with potential machine disruptions", *Omega*, **69**, pp. 17-28 (2017).
15. Luo, W., and Ji, M. "Scheduling a variable maintenance and linear deteriorating jobs on a single machine", *Information Processing Letters*, **115**, pp. 33-39 (2015).
16. Lee, W.C., Wu, C.C. and Chung, Y.H. "Scheduling deteriorating jobs on a single machine with release times", *Computers & Industrial Engineering*, **54**(3), pp. 441-452 (2008).
17. Wu, C.C. and Lee, W.C. "Two-machine flowshop scheduling to minimize mean flow time under linear deterioration", *International Journal of Production Economics*, **103**(2), pp. 572-584 (2006).
18. Lee, W.C., Wu, C.C., Wen, C.C. and Chung, Y.H. "A two-machine flowshop makespan scheduling problem with deteriorating jobs", *Computers & Industrial Engineering*, **54**(4), pp. 737-749 (2008).
19. Wang, J.B. and Wang, M.Z. "Minimizing makespan in three-machine flow shops with deteriorating jobs", *Computers & Operation Research*, **40**(2), pp. 547-557 (2013).
20. Yin, Y., Cheng, T.C.E., Wan, L., Wu, C.C. and Liu, J. "Two-agent single-machine scheduling with deteriorating jobs", *Computers & Industrial Engineering*, **81**, pp. 177-185 (2015).
21. Mosheiov, G. "Scheduling jobs under simple linear de-

terioration", *Computers & Operation Research*, **21**(6), pp. 653-659 (1994).

22. Wang, J.B., Ng, C.T.D., Chen, T.C.E. and Liu, L.L. "Minimizing total completion time in a two-machine flow shop with deteriorating jobs", *Applied Mathematics and Computation*, **180**(1), pp. 185-193 (2006).

23. Yang, S.H. and Wang, J.B. "Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration", *Applied Mathematics and Computation*, **217**(9), pp. 4819-4826 (2011).

24. Wu, C.C. and Lee, W.C. "Scheduling linear deteriorating jobs to minimize makespan with an availability constraint on a single machine", *Information Processing Letters*, **87**(2), pp. 89-93 (2003).

25. Cheng, T.C.E., Hsu, C.J., Huang, Y.C. and Lee, W.C. "Single-machine scheduling with deteriorating jobs and setup times to minimize the maximum tardiness", *Computers & Operation Research*, **38**(12), pp. 1760-1765 (2011).

26. Lee, W.C., Lin, J.B. and Shiau, Y.R. "Deteriorating job scheduling to minimize the number of late jobs with setup times", *Computers & Industrial Engineering*, **61**(3), pp. 782-787 (2011).

27. Kubiak, W. and Velde, S. "Scheduling deteriorating jobs to minimize makespan", *Naval Research Logistics*, **45**(5), pp. 511-523 (1998).

28. Lalla Ruiz, E. and Vob, S. "Modeling the parallel machine scheduling problem with step deteriorating jobs", *European Journal of Operational Research*, **255**(1), pp. 21-33 (2016).

29. Jafari, A. and Moslehi, G. "Minimizing the weighted number of tardy jobs on a single machine under piecewise linear deterioration", *9th International Industrial Engineering Conference*, Tehran, Iran (2013).

30. Lai, P., Wu, C. and Lee, W. "Single machine scheduling with logarithm deterioration", *Optimization Letters*, **6**(8), pp. 1719-1730 (2011).

31. Lee, C. and Yu, G. "Parallel machine scheduling under potential disruption", *Optimization Letters*, **2**(1), pp. 27-37 (2008).

32. Brucker, P. *Scheduling Algorithm*, 5th Ed., Heidelberg, Berlin (2006).

Appendix

Proof of Theorem 3: $T_{\max}(\delta'_{LB})$ in algorithm LB is based on comparing the least completion time for each position after partial sequence δ to the best possible due date (Steps 1 and 2). At first, we show that if non-decreasing ratio of a_i/a_b is observed for each position, M, after partial sequence, δ, then the shortest completion time for position M is obtained. Then, we prove that if this completion time is compared to the least existing due date, LB_3 is a lower bound for this problem.

Let $\delta_1 = (\delta, \pi, j_i, j_j)$ where δ and π are partial sequences with completion times S and C, and i and j are located in positions $M-1$ and M after δ. Sequence $\delta_2 = (\delta, \pi, j_j, j_i)$ is obtained from δ_1 by interchanging j_i and j_j. To show that the shortest completion time for position M is based on non-decreasing ratio of a_i/b_i, we must show that $a_i/b_i \leq a_j/b_j$ to have $C_j(\delta_1) \leq C_i(\delta_2)$. According to Relation (2), completion times of j_i and j_j in δ_1 are as follows:

$$C_i(\delta_1) = \begin{cases} C + a_i \quad \text{if} \quad C \leq y_1 \\ a_i + b_i(C - y_1) + C = a_i \\ \quad +(1+b_i)C - b_i y_1 \quad \text{if} \quad y_1 < C \leq y_2 \end{cases} \tag{A.1}$$

If $C_i(\delta_1) < y_2$, then completion time of j_j in δ_1 is as follows:

$$\begin{aligned} C_j(\delta_1) &= a_j + b_j(C_i(\delta_1) - y_1) + C_i(\delta_1) \\ &= a_j + b_j(a_i + (1+b_i)C - b_i y_1 - y_1) \\ &\quad + a_i + (1+b_i)C - b_i y_1 \\ &= a_j + a_i b_j + b_j(1+b_i)C - b_i b_j y_1 - b_j y_1 + a_i \\ &\quad + (1+b_i)C - b_i y_1. \end{aligned} \tag{A.2}$$

Now, if $C_i(\delta_1) > y_2$, then completion time of j_j in δ_1 is as follows:

$$\begin{aligned} C_j(\delta_1) &= a_j + b_j(y_2 - y_1) + C_i(\delta_1) \\ &= a_j + b_j(y_2 - y_1) + a_i + (1+b_i)C - b_i y_1. \end{aligned} \tag{A.3}$$

Also, completion times of j_i and j_j in δ_2 are as follows:

$$C_j(\delta_2) = \begin{cases} C + a_j \quad \text{if} \quad C \leq y_1 \\ a_j + b_j(C - y_1) + C = a_j + (1+b_j) \\ \quad C - b_j y_1 \quad \text{if} \quad y_1 < C \leq y_2 \end{cases} \tag{A.4}$$

If $C_j(\delta_2) < y_2$, then completion time of j_j in δ_1 is as follows:

$$\begin{aligned} C_i(\delta_2) &= a_i + b_i(C_j(\delta_2) - y_1) + C_j(\delta_2) \\ &= a_i + b_i(a_j + (1+b_j)C - b_j y_1 - y_1) \\ &\quad + a_j + (1+b_j)C - b_j y_1 \\ &= a_i + b_i a_j + b_i(1+b_j)C - b_i b_j y_1 - b_i y_1 + a_j \\ &\quad + (1+b_j)C - b_j y_1. \end{aligned} \tag{A.5}$$

Now, if $C_j(\delta_2) > y_2$, then completion time of j_j in δ_1 is as follows:

$$C_i(\delta_2) = a_i + b_i(y_2 - y_1) + C_j(\delta_2)$$

$$= a_i + b_i(y_2 - y_1) + a_j + (1 + b_j)C - b_j y_1. \quad (A.6)$$

According to the completion time of j_i and j_j, there are four cases to prove Theorem 3 that need to be checked one by one.

Case 1: $C_i(\delta_1) < y_2$ and $C_j(\delta_2) < y_2$:

$$\begin{aligned}
C_j(\delta_1) - C_i(\delta_2) &= (a_j + a_i b_j + b_j(1 + b_i)C \\
&- b_i b_j y_1 - b_j y_1 + a_i + (1 + b_i)C - b_i y_1) \\
&- (a_i + b_i a_j + b_i(1 + b_j)C - b_i b_j y_1 - b_i y_1 \\
&+ a_j + (1 + b_j)C - b_j y_1) \\
&= a_j + a_i b_j + b_j C + b_i b_j C - b_i b_j y_1 - b_j y_1 \\
&+ a_i + C + b_i C - b_i y_1 - a_i - b_i a_j - b_i C \\
&- b_i b_j C + b_i b_j y_1 + b_i y_1 - a_j - C - b_j C + b_j y_1 \\
&= a_i b_j - b_i a_j. \quad (A.7)
\end{aligned}$$

Since $C_j(\delta_1) - C_i(\delta_2) \leq 0$, we have $a_i b_j - b_i a_j \leq 0$. Therefore, it implies that $a_i/b_i \leq a_j/b_j$.

Case 2: $C_i(\delta_1) < y_2$ and $C_j(\delta_2) > y_2$:

$$\begin{aligned}
C_j(\delta_1) - C_i(\delta_2) &= (a_j + a_i b_j + b_j(1 + b_i)C - b_i b_j y_1 \\
&- b_j y_1 + a_i + (1 + b_i)C - b_i y_1) - (a_i \\
&+ b_i(y_2 - y_1) + a_j + (1 + b_j)C - b_j y_1) \\
&= a_j + a_i b_j + b_j C + b_i b_j C - b_i b_j y_1 - b_j y_1 \\
&+ a_i + C + b_i C - b_i y_1 - a_i - b_i y_2 + b_i y_1 - a_j \\
&- C - b_j C + b_j y_1 \\
&= a_i b_j + b_i b_j C - b_i b_j y_1 \\
&+ b_i C - b_i y_2. \quad (A.8)
\end{aligned}$$

From $C_i(\delta_1) < y_2$, we obtain the following relation:

$$\begin{aligned}
&a_i b_j + b_i b_j C - b_i b_j y_1 + b_i C - b_i y_2 \leq a_i b_j + b_i b_j C \\
&\quad - b_i b_j y_1 + b_i C - b_i C_i(\delta_1). \quad (A.9)
\end{aligned}$$

If the right-hand side of Inequality (A.9) is not positive, then the left-hand side will not be positive. Thus, the following relation is valid:

$$\begin{aligned}
&a_i b_j + b_i b_j C - b_i b_j y_1 + b_i C - b_i C_i(\delta_1) = a_i b_j \\
&\quad + b_i b_j C - b_i b_j y_1 + b_i C - b_i(a_i + (1 + b_i)C - b_i y_1) \\
&\quad \leq 0, \\
&a_i b_j + b_i b_j C - b_i b_j y_1 + b_i C - b_i a_i \\
&\quad - b_i C - {b_i}^2 C + {b_i}^2 y_1 \\
&\quad = b_j(a_i + b_i C - b_i y_1) \\
&\quad - b_i(a_i + b_i C - b_i y_1) \\
&\quad \leq 0, \\
&b_j(a_i + b_i C - b_i y_1) \\
&\quad \leq b_i(a_i + b_i C - b_i y_1), \\
&b_j \leq b_i. \quad (A.10)
\end{aligned}$$

On the other hand, from relations $C_i(\delta_1) < y_2$ and $y_2 < C_j(\delta_2)$, we have $C_i(\delta_1) < C_j(\delta_2)$:

$$\begin{aligned}
&a_i + (1 + b_i)C - b_i y_1 < a_j + (1 + b_j)C - b_j y_1, \\
&a_i + b_i C - b_i y_1 < a_j + b_j C - b_j y_1, \\
&a_i - a_j < (b_j - b_i)C - (b_j - b_i)y_1 = (b_j - b_i)(C - y_1). \quad (A.11)
\end{aligned}$$

Owing to $b_j \leq b_i$ and $y_1 < C$, relation $(b_j - b_i)(C - y_1) \leq 0$ and, consequently, $a_i < a_j$ hold. Accordingly, we have $a_i/b_i \leq a_j/b_j$.

Case 3: $C_i(\delta_1) > y_2$ and $C_j(\delta_2) < y_2$:

$$\begin{aligned}
C_j(\delta_1) - C_i(\delta_2) &= (a_j + b_j(y_2 - y_1) + a_i + (1 + b_i)C \\
&- b_i y_1) - (a_i + b_i a_j + b_i(1 + b_j)C - b_i b_j y_1 \\
&- b_i y_1 + a_j + (1 + b_j)C - b_j y_1) \\
&= a_j + b_j y_2 - b_j y_1 + a_i + C + b_i C - b_i y_1 \\
&- a_i - b_i a_j - b_i C - b_i b_j C + b_i b_j y_1 + b_i y_1 \\
&- a_j - C - b_j C + b_j y_1 \\
&= b_j y_2 - b_i a_j - b_i b_j C \\
&+ C + b_i C - b_i y_1 - a_i - b_i y_2 + b_i y_1 - a_j \\
&+ b_i b_j y_1 - b_j C. \quad (A.12)
\end{aligned}$$

Since $C_i(\delta_1) > y_2$, we can replace $C_i(\delta_1)$ with y_2:

$$b_jy_2 - b_ia_j - b_ib_jC + b_ib_jy_1 - b_jC < b_jC_i(\delta_1) - b_ia_j - b_ib_jC + b_ib_jy_1 - b_jC = b_j(a_i + C + b_iC - b_iy_1) - b_ia_j - b_ib_jC + b_ib_jy_1 - b_jC = b_ja_i - b_ia_j. \quad (A.13)$$

$C_j(\delta_1) - C_i(\delta_2) \leq 0$; therefore, relations $a_ib_j - b_ia_j \leq 0$ and $a_i/b_i \leq a_j/b_j$ are satisfied.

Case 4: $C_i(\delta_1) > y_2$ and $C_j(\delta_2) > y_2$:

$$C_j(\delta_1) - C_i(\delta_2) = (a_j + b_j(y_2 - y_1) + a_i + (1 + b_i)C - b_iy_1) - (a_i + b_i(y_2 - y_1) + a_j + (1 + b_j)C - b_jy_1) = b_j(y_2 - C) + b_i(C - y_2) = (b_j - b_i)(y_2 - C). \quad (A.14)$$

Since we want to have $C_j(\delta_1) - C_j(\delta_2) \leq 0$, we have $(b_j - b_i)(y_1 - C) \leq 0$. From $C \leq y_2$, we obtain $b_j < b_i$. While the shortest normal processing time (a_i) is used in Relation (A.1), the least amount of $C_i(\delta_1)$ in position M is obtained. Therefore, relation $a_i \leq a_j$ is satisfied. Consequently, we have $a_i/b_i \leq a_j/b_j$.

In the above four cases, we proved that non-decreasing ratio a_i/b_i causes to have the least completion time for each position. If the start time of position is before y_1, then ratio a_i/b_i is reduced to a_i. According to Step 1 of algorithm LB, the least amount of a_i from set δ' is selected and scheduled, then it will be omitted from δ'. If the start time is greater than y_1 (e.g., position M), then the shortest a_i remained in set δ' (e.g., a_j) is scheduled for this position so that $a_i \leq a_j$. To ensure non-decreasing ratio of a_i/b_i, relation $b_j < b_i$ must hold; so, M jobs with the least deterioration rates from set $B_{\delta'}$ are chosen and scheduled in positions 1 to M after δ in a non-increasing order. Then, completion time of job in position M is calculated according to the completion times of the previous positions.

We showed how to obtain the least completion time of each position based on the non-decreasing ratio of a_i/b_i so far. Now, we prove that if the least existing due date is assigned to each opposition, the lower bound will be obtained. We have:

$$T_{\max}(\delta'_{LB}) = \max_{1\leq k\leq u}\{0, C_{[k]} - d_{[k]}\} = \max\{0, C_{[1]} - d_{[1]}, C_{[2]} - d_{[2]}, C_{[3]} - d_{[3]}, ..., C_{[u]} - d_{[u]}\}, \quad (A.15)$$

where $C_{[k]}$ is the least completion time of position k, and $d_{[k]}$ is the kth smallest due date from set $D_{\delta'}$.

Suppose that there is a number $1 \leq h \leq u$ in partial sequence δ'_{LB} so that $T_{\max}(\delta'_{LB}) = C_{[h]} - d_{[h]}$. There is partial sequence δ' where $d_{[h]}$ is not assigned to position h; hence, there is a number $1 \leq k \leq u$, so that $d_{[h]} \leq d_{[k]}$ or $d_{[k]} < d_{[h]}$ where $d_{[h]}$ and $d_{[k]}$ are assigned to positions k and h, respectively. Since $T_{\max}(\delta'_{LB}) = C_{[h]} - d_{[h]}$, relations $C_{[k]} - d_{[k]} \leq C_{[h]} - d_{[k]}$ and $T_{\max}(\delta') \geq \max\{C_{[h]} - d_{[k]} \leq C_{[k]} - d_{[h]}\}$ hold.

If $d_{[h]} \leq d_{[k]}$, then relation $h < k$ is satisfied absolutely. Therefore, we have:

$$C_{[h]} < C_{[k]},$$

$$C_{[h]} - d_{[h]} < C_{[k]} - d_{[h]}$$

$$T_{\max}(\delta'_{LB}) = C_{[h]} - d_{[h]} < C_{[k]} - d_{[h]} \leq T_{\max}(\delta'),$$

$$T_{\max}(\delta'_{LB}) \leq T_{\max}(\delta'). \quad (A.16)$$

If $d_{[k]} < d_{[h]}$, then the following relation is satisfied:

$$T_{\max}(\delta'_{LB}) = C_{[h]} - d_{[h]} < C_{[h]} - d_{[k]} \leq T_{\max}(\delta'),$$

$$T_{\max}(\delta'_{LB}) \leq T_{\max}(\delta'). \quad (A.17)$$

Therefore, partial sequence δ'_{LB} dominates partial sequence δ'. Accordingly, LB_3 is a lower bound for the problem.

Biographies

Abbas-Ali Jafari is a PhD candidate at the Department of Industrial Engineering at Yazd University. He received his BS degree from Yazd University and his MS degree from Isfahan University of Technology, both in Iran. One of his main field of interest is the scheduling and production planning.

Mohammad Mehdi Lotfi is an Associate Professor at the Department of Industrial Engineering at Yazd University. He received his BS degree from Sharif University of Technology and his MS and PhD degrees from University of Tehran, all in Iran. One of his main areas of interest is the production planning, scheduling and control in operation systems.