Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 19(9): 8923–8934.
DOI: 10.3934/mbe.2022414
Received: 15 April 2022
Revised: 04 June 2022
Accepted: 15 June 2022
Published: 20 June 2022

Research article

Polynomial time algorithm for minmax scheduling with common due-window and proportional-linear shortening processing times

Xue Jia, Jing Xue, Shi-Yun Wang and Ji-Bo Wang*

School of Science, Shenyang Aerospace University, Shenyang, China

* **Correspondence:** Email: wangjibo75@163.com.

Abstract: This article deals with common due-window assignment and single-machine scheduling with proportional-linear shortening processing times. Objective cost is a type of minmax, that is, the maximal cost among all processed jobs is minimized. Our goal is to determine an optimal schedule, the optimal starting time, and size of due-window that minimize the worst cost, which consist of four parts: earliness, tardiness, starting time and length of the due-window. Optimal properties of the problem are given, and then an optimal polynomial algorithm is proposed to solve the problem.

Keywords: minmax scheduling; common due-window; deterioration effect

1. Introduction

In practice, the processing times of the jobs are often variable with the change of their starting time, this is the time-dependent processing times [1–3]. In recent years, more and more experts and scholars have studied time-related deterioration. Huang [4] studied a single machine bicriterion problem in which the processing time and group setup time is a linear function of its starting time can be solved optimally. Li and Lu [5] studied single-machine parallel-batch scheduling with deterioration effects. Under total rejection costs which cannot exceed a given constant, they showed that the problem of minimizing the the total weighted completion time (makespan) is NP-hard. Liang et al. [6] showed the weighted sum of makespan and resource cost minimization with deterioration effect and group technology remains polynomially solvable. Huang et al. [7] studied common due-window assignment problem in which the processing time is a proportional linear function. They proved that two different non-regular problems are polynomial solvable.

On the other hand, many scholars have conducted research on minmax scheduling problems on due-date or due-window assignment, i.e., minmax means that the maximal cost is minimized. Mosheiov [8] considered minmax scheduling with a common due-date assignment on parallel identical machines. The goal was to find the job schedule and due-date assignment with minimum cost of the worst

scheduled job, and they proposed an efficient heuristic algorithm. Mosheiov and Sarig [9] studied due-window assignment scheduling problems, where the objection function is a minmax type. The objective function contains earliness, tardiness, the starting time, and size of the due-window. They proved that the problem remains polynomially solvable when the processing time of jobs are constants. Gerstl and Mosheiov [10] investigated single-machine scheduling with the due-date assignment. They demonstrated that the minmax minimization can be solved in polynomial time. Mosheiov [11] studied due-window assignment problems on a type of minmax. The earliness penalties, tardiness penalties, the cost of position, and the size of due-window were considered. The researcher offered evidence that the scheduling problems can be solved in polynomial time on a single-machine and provided an LPT-based heuristic (the LPT rule means processing with the largest processing time) on parallel identical machines which is NP-hard. Mor [12] considered minmax minimization with position-dependent processing time. For the common due-date and the common due-window, the researcher elucidates that these two problems are polynomially solvable by transforming them into assignment problems. Numerical simulation or numerical examples are given for all the problems. Mosheiov et al. [13] studied due-window assignment problems with position-dependent processing time and rejection jobs on a flow shop, and they illustrated that it remains polynomially solvable. More of scheduling with due-window (due-date) assignment can be seen in [14–20].

In the actual processing environment, the processing time of the jobs is often changes with time. In this paper, we study minmax scheduling problems with proportional-linear shortening processing times (denoted by $PLSPT$). The objective function of this study has four components: earliness, tardiness, the starting time, and the size of the due window. The contributions of this article are demonstrated as follows: Firstly, considering the position and size of the due-window are known, the optimal scheduling sequence and the optimal value can be found. Then, the optimal scheduling and the optimal objective function can be found when the size or position of the due-window is known. According to the previous analysis, the optimal ordering is discussed when the size and position of the due-window are unknown. We prove that these problems with $PLSPT$ remain polynomially solvable, i.e., the complexity is $O(n)$, this is identical to that of the classical version (without any $PLSPT$), where n is the number of jobs.

The rest of this study is organized as follows: Section 2 introduces the problem. Section 3 considers the scheduling problem on a single machine, which is discussed in four cases that center on whether or not the location and the size of the due-window are known. Computational experiments are given in Section 4. The last section is conclusion.

2. Problem definition

We investigate a set of n jobs $\breve{Q} = \{J_1, J_2, ...J_n\}$ to be processed on a single-machine that cannot be interrupted. All the jobs are available for processing at time s ($s \geq 0$). The general linear shortening model is as follows: the actual processing time of job J_j is $p_j = a_j - b_j s_j$, where a_j, b_j, s_j represent the normal processing time (the processing time without any linear shortening), shortening rate (the decreasing rate) and starting time of job J_j, respectively. It is assumed that shortening rates b_j satisfy the following condition: $0 \leq b_j < 1$ and $b_j\left(s + \sum_{i=1}^{n} a_i - a_j\right) < a_j$ (see [21,22]). In this article, a special case (i.e., $b_i = \theta a_i$, for some $\theta > 0$) will be studied; that is, $p_j = a_j\left(1 - \theta s_j\right)$, where $\theta\left(s + \sum_{j=1}^{n} a_j - a_{\min}\right) < 1$

($a_{\min} = \min\left\{a_j\right\}, j = 1, 2, ..., n$).

We suppose that all of the jobs have a common due-window $[\widehat{d_1}, \widehat{d_2}]$, where d_1, d_2, $D = \widehat{d_2} - \widehat{d_1}$ represent the starting time, finishing time and size of the due-window. Let C_j be completion time of J_j, and $E_j = \max\left\{\widehat{d_1} - C_j, 0\right\}$ ($T_j = \max\left\{C_j - \widehat{d_2}, 0\right\}$) represent earliness (tardiness) of J_j. In this article, we consider the scheduling of minimizing maximum cost function (including earliness penalties, tardiness penalties, the cost for the starting time, and size of the due-window), i.e., the objective is to minimize the maximum cost of all jobs:

$$Y = \max_{1\le j\le n}\left\{\max\left\{\lambda E_j + \gamma\widehat{d_1} + \delta D, \beta T_j + \gamma\widehat{d_1} + \delta D\right\}\right\} = \max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\} \tag{1}$$

where λ, β, γ, δ represent unit penalty costs of earliness, tardiness, starting time and size of the due-window. As in Gawiejnowicz [3], we denote the scheduling problem as

$$1|CONW, PLS\,PT|\max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\} \tag{2}$$

where $CONW$ is the common due-window.

3. Main results

Lemma 1. *Let $[\xi]$ be the job scheduled at ξth position, if the first job's starting time is s, then*

$$C_{[\xi]} = \left(s - \frac{1}{\theta}\right)\prod_{i=1}^{\xi}(1-\theta a_{[i]}) + \frac{1}{\theta}. \tag{3}$$

Proof. By induction.

$$C_{[1]} = s + a_{[1]} - b_{[1]}s = \left(s - \tfrac{1}{\theta}\right)(1 - \theta a_{[1]}) + \tfrac{1}{\theta}.$$

Suppose Lemma 1 holds for job $J_{[\xi]}$, $\xi \ge 2$, i.e.,

$$C_{[\xi]} = \left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{\xi}(1-\theta a_{[i]}) + \tfrac{1}{\theta}.$$

Consider job $J_{[\xi+1]}$.

$$\begin{aligned} C_{[\xi+1]} &= \left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{\xi}(1-\theta a_{[i]}) + \tfrac{1}{\theta} + a_{[\xi+1]}\left\{1 - \theta\left[\left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{\xi}(1-\theta a_{[i]}) + \tfrac{1}{\theta}\right]\right\} \\ &= \left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{\xi+1}(1-\theta a_{[i]}) + \tfrac{1}{\theta}. \end{aligned}$$

Hence, Lemma 1 holds.

From Lemma 1, if $s = 0$, $C_{[\xi]} = \frac{1}{\theta} - \frac{1}{\theta}\prod_{i=1}^{\xi}(1-\theta a_{[i]})$.

3.1. The $\widehat{d_1}$ and D are known

When $\widehat{d_1}$ and D (thus $\widehat{d_2}$) are known, obviously, the maximum earliness $\max E_j$ (tardiness $\max T_j$) can be determined by the first (final) processed job, hence

$$\begin{aligned} Y &= \max\left\{\lambda\left[\widehat{d_1} - C_{[1]}\right], \beta\left(C_{[n]} - \widehat{d_2}\right)\right\} + \gamma\widehat{d_1} + \delta D \\ &= \max\left\{\lambda(\widehat{d_1} - s - a_1 + \theta a_{[1]}s), \beta\left[\left(s - \frac{1}{\theta}\right)\prod_{i=1}^{n}(1-\theta a_{[i]}) + \frac{1}{\theta} - \widehat{d_2}\right]\right\} \\ &\quad + \gamma\widehat{d_1} + \delta D. \end{aligned} \tag{4}$$

Lemma 2. *In the case that $\widehat{d_1}$ and D (thus $\widehat{d_2}$) are known, the optimal schedule is to process the job with the largest normal processing time first, and the the order of the remaining jobs is arbitrary.*

Proof. Let $a_{\max} = \max\left\{a_j\right\}$ $(1 \le j \le n)$. It is obvious that the actual and normal processing time of any job are non-negative, so we obtain $\theta s - 1 < 0$. From (4), we have

$$\begin{aligned} &\max\left\{\lambda[\widehat{d_1} - s + a_{\max}(\theta s - 1)], \beta\left[\left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{n}(1-\theta a_i) + \tfrac{1}{\theta} - \widehat{d_2}\right]\right\} + \gamma\widehat{d_1} + \delta D \\ &\le \max\left\{\lambda\left(\widehat{d_1} - s - a_{[1]} + \theta a_{[1]}s\right), \beta\left[\left(s - \tfrac{1}{\theta}\right)\prod_{i=1}^{n}(1-\theta a_{[i]}) + \tfrac{1}{\theta} - \widehat{d_2}\right]\right\} + \gamma\widehat{d_1} + \delta D. \end{aligned}$$

Let $\lambda[\widehat{d_1} - s + a_{\max}(\theta s - 1)] = \beta\left[\left(s - \frac{1}{\theta}\right)\prod_{i=1}^{n}(1-\theta a_i) + \frac{1}{\theta} - \widehat{d_2}\right]$, we have

$$s = \frac{\lambda\widehat{d_1} + \beta\widehat{d_2} - \frac{1}{\theta}(\lambda+\beta)}{\beta\prod_{i=1}^{n}(1-\theta a_i) + \lambda(1-\theta a_{\max})} + \frac{1}{\theta} \tag{5}$$

and

$$Y = \beta\left[\left(s - \frac{1}{\theta}\right)\prod_{i=1}^{n}(1-\theta a_i) + \frac{1}{\theta} - \widehat{d_2}\right] + \gamma\widehat{d_1} + \delta D. \tag{6}$$

Algorithm 1

Step 1. Find the job with the largest normal processing time (i.e., $a_{\max} = \max\left\{a_j | j = 1, 2, \ldots, n\right\}$) and process it to the first position (the remaining jobs are scheduled in any order).

Step 2. From (5), calculate $s = \frac{\lambda\widehat{d_1} + \beta\widehat{d_2} - \frac{1}{\theta}(\lambda+\beta)}{\beta\prod_{i=1}^{n}(1-\theta a_i) + \lambda(1-\theta a_{\max})} + \frac{1}{\theta}$, set the optimal starting time of the job scheduled at the first position $s^* = \max\{s, 0\}$ and its maximum Y^* can be calculated from (6).

Theorem 1. *If $\widehat{d_1}$ and D are given constants, then Algorithm 1 solves*

$$1|CONW, PLS\,PT|\max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\}$$

in $O(n)$ time.

Proof. Step 1 needs $O(n)$ time; *Step 2* runs in constant time; hence, the total computational complexity is $O(n)$.

Numerical Example 1.

Consider a 8-job problem, where $\theta = 0.03$, $\lambda = 3$, $\beta = 5$, $\gamma = 2$, $\delta = 4$, $a_1 = 5, a_2 = 16, a_3 = 18, a_4 = 8, a_5 = 10, a_6 = 23, a_7 = 12, a_8 = 21$, $\widehat{d_1} = 16$ and $D = 8$.

Solution According to Algorithm 1, the job with the largest normal processing time is J_6 (i.e., $a_{\max} = a_6 = 23$) and process it to the first position (the remaining jobs are scheduled in any order). From (5) and (6), we have $s = -70.61$, the optimal starting time is $s^* = \max\{s, 0\} = 0$ and the maximum $Y^* = 110.03$.

3.2. The D is known

If D is known, we first determined s of the due-window, and then determine the optimal schedule and optimal value of the jobs. Without loss of generality, all the jobs begin processing at time 0.

According to Lemma 2, the job with the largest normal processing time has priority in processing, so $C_{[1]} = a_{\max}$. For any given schedule, the maximum completion time of n jobs is constant, so $C_{[n]}$ is a constant, i.e., $C_{[n]} = -\frac{1}{\theta}\prod_{i=1}^{n}(1-\theta a_i) + \frac{1}{\theta}$.

This minmax minimization can be formulated as the following linear program (*LP*).

$$Min\ Y$$

$$s.t.\begin{cases} Y \geq \lambda E_{[1]} + \gamma\widehat{d_1} + \delta D = (\lambda+\gamma)\,\widehat{d_1} - \lambda C_{[1]} + \delta D, \\ Y \geq \beta T_{[n]} + \gamma\widehat{d_1} + \delta D = \beta C_{[n]} + (\delta-\beta)\,D + (\gamma-\beta)\,\widehat{d_1}, \\ \widehat{d_1} \geq 0. \end{cases} \tag{7}$$

To facilitate a discussion on the optimal location of the due-window, suppose the D is known. Because $\lambda C_{[1]}$, δD, $\beta C_{[n]}$ and $(\delta-\beta)\,D$ are constants, we should discuss the relationship between γ and β. In most cases, it is more realistic to discuss the case of $D \leq C_{[n]}$.

Case 1: If $\gamma > \beta$, we have $\gamma - \beta > 0$. In this case, $\widehat{d_1}$ takes its minimum value, i.e., $\widehat{d_1^*} = 0$.

Case 2: If $\gamma \leq \beta$, we have $\gamma - \beta \leq 0$. Moreover, if $C_{[n]} - C_{[1]} < D \leq C_{[n]}$, we have $0 \leq \widehat{d_1} < C_{[1]}$. In this case, $\widehat{d_1}$ takes its maximum value, i.e. $\widehat{d_1^*} = C_{[1]}$ and this situation may result in tardiness costs due to tardiness jobs, in which the maximum cost is $Y^*(D) = \beta C_{[n]} + (\delta-\beta)\,D + (\gamma-\beta)\,C_{[1]}$.

If $D \leq C_{[n]} - C_{[1]}$, we have $\widehat{d_1} \geq C_{[1]}$, so in this case, there may be jobs that are completed before or after the due-window. Let

$$\lambda E_{[1]} = \beta T_{[n]},$$

i.e.,

$$\lambda\left(\widehat{d_1} - C_{[1]}\right) = \beta\left(C_{[n]} - D - \widehat{d_1}\right),$$

we have

$$\widehat{d_1^*} = \frac{\lambda C_{[1]} + \beta C_{[n]} - \beta D}{\lambda+\beta},$$

and its maximum cost $Y^*(D) = \frac{\lambda(\gamma-\beta)}{\lambda+\beta}C_{[1]} + \frac{\beta(\lambda+\gamma)}{\lambda+\beta}C_{[n]} + \left(\delta - \frac{\beta(\lambda+\gamma)}{\lambda+\beta}\right)D$.

Algorithm 2

Step 1. Find the job with the largest normal processing time (i.e., $a_{\max} = \max\left\{a_j | j = 1, 2, \ldots, n\right\}$) and process it to the first position (the remaining jobs are scheduled in any order).

Step 2. If $\gamma > \beta$, setting $\widehat{d_1^*} = 0$ and $Y^*(D) = \beta C_{[n]} + (\delta - \beta) D$.

Otherwise, if $C_{[n]} - C_{[1]} < D \leq C_{[n]}$, set $\widehat{d_1^*} = C_{[1]}$ and $Y^*(D) = \beta C_{[n]} + (\delta - \beta) D + (\gamma - \beta) C_{[1]}$.

Otherwise, set $\widehat{d_1^*} = \frac{\lambda C_{[1]} + \beta C_{[n]} - \beta D}{\lambda + \beta}$ and $Y^*(D) = \frac{\lambda(\gamma-\beta)}{\lambda+\beta} C_{[1]} + \frac{\beta(\lambda+\gamma)}{\lambda+\beta} C_{[n]} + \left(\delta - \frac{\beta(\lambda+\gamma)}{\lambda+\beta}\right) D$.

Theorem 2. *If D is a given constant, Algorithm 2 solves*

$$1|CONW, PLS\,PT| \max_{1 \leq j \leq n} \left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma \widehat{d_1} + \delta D\right\}$$

in O(n) time.

Numerical Example 2.

Consider a 8-job problem, where $s = 0$, $D = 8$, $\theta = 0.03$, $\lambda = 3$, $\beta = 5$, $\gamma = 2$, $\delta = 4$, $a_1 = 5, a_2 = 16, a_3 = 18, a_4 = 8, a_5 = 10, a_6 = 23, a_7 = 12, a_8 = 21$.

Solution According to Algorithm 2, the job with the largest normal processing time is J_6 and process it to the first position (the remaining jobs are scheduled in any order). Since $\gamma < \beta$, we have $C_{[1]} = 23$, $C_{[n]} = 33.21$. In addition, $D = 8 < 23 = C_{[1]}$, we have $\widehat{d_1^*} = \frac{\lambda C_{[1]} + \beta C_{[n]} - \beta D}{\lambda + \beta} = 24.38$ and maximum $Y^*(D) = \frac{\lambda(\gamma-\beta)}{\lambda+\beta} C_{[1]} + \frac{\beta(\lambda+\gamma)}{\lambda+\beta} C_{[n]} + \left(\delta - \frac{\beta(\lambda+\gamma)}{\lambda+\beta}\right) D = 84.89$.

3.3. The $\widehat{d_1}$ is known

In the case where $\widehat{d_1}$ is known, we first determine the size of the due-window, and then determine the optimal schedule of jobs, assuming that all jobs are arrived at time 0. Without loss of generality, all jobs begin processing at time 0.

According to Lemma 2, the jobs with the largest normal processing time have priority in processing, so $C_{[1]} = a_{\max}$ and $C_{[n]} = \frac{1}{\theta} - \frac{1}{\theta} \prod_{i=1}^{n} (1 - \theta a_i)$ is a constant.

This minmax minimization can be formulated as:

$$Min\ Y$$

$$s.t. \begin{cases} Y \geq \lambda E_{[1]} + \gamma \widehat{d_1} + \delta D = (\lambda + \gamma) \widehat{d_1} - \lambda C_{[1]} + \delta D \\ Y \geq \beta T_{[n]} + \gamma \widehat{d_1} + \delta D = \beta C_{[n]} + (\delta - \beta) D + (\gamma - \beta) \widehat{d_1} \\ D \geq 0. \end{cases} \tag{8}$$

Suppose that d_1 is known, we should determine the optimal D. Because $\lambda C_{[1]}$, $(\lambda + \gamma) \widehat{d_1}$, $\beta C_{[n]}$ and $(\gamma - \beta) \widehat{d_1}$ are constants, we should discuss the relationship between δ and β.

Case 1: If $\delta \geq \beta$, we have $\delta - \beta \geq 0$. Because $\delta D > 0$ and $(\delta - \beta) D \geq 0$, it is optimal to make $D^* = 0$. In this case, the maximum consumption function is $Y^*(d_1) = \beta C_{[n]} + (\gamma - \beta) \widehat{d_1}$.

Case 2: If $\delta < \beta$, we have $\delta - \beta < 0$. Because $\delta D > 0$ and $(\delta - \beta) D < 0$, it is optimal to make the earliness penalty of the job with largest normal processing time equal to tardiness penalty of the last job, i.e.,

$$\lambda E_{[1]} = \beta T_{[n]},$$

and we have

$$D^* = \frac{\lambda C_{[1]} + \beta C_{[n]} - (\beta+\lambda)\widehat{d_1}}{\beta}.$$

In this case, the maximum consumption function is $Y^*(\widehat{d_1}) = \frac{\lambda(\delta-\beta)}{\beta} C_{[1]} + \delta C_{[n]} + \left(\lambda + \gamma - \delta - \frac{\delta\lambda}{\beta}\right)\widehat{d_1}$.

Algorithm 3

Step 1. Find the job with the largest normal processing time (i.e., $a_{\max} = \max\left\{a_j | j = 1, 2, \ldots, n\right\}$) and process it to the first position (the remaining jobs are scheduled in any order).

Step 2. If $\delta > \beta$, setting $D^* = 0$ and $Y^*(\widehat{d_1}) = \beta C_{[n]} + (\gamma - \beta)\widehat{d_1}$.

Otherwise, set $D^* = \frac{\lambda C_{[1]} + \beta C_{[n]} - (\beta+\lambda)\widehat{d_1}}{\beta}$ and $Y^*(\widehat{d_1}) = \frac{\lambda(\delta-\beta)}{\beta} C_{[1]} + \delta C_{[n]} + \left(\lambda + \gamma - \delta - \frac{\delta\lambda}{\beta}\right)\widehat{d_1}$.

Theorem 3. *If d_1 is a given constant, Algorithm 3 solves*

$$1|CONW, PLS\,PT| \max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\}$$

in $O(n)$ time.

Numerical Example 3.

Consider a 8-job problem, where $s = 0$, $\widehat{d_1} = 16$, $\theta = 0.03$, $\lambda = 3$, $\beta = 5$, $\gamma = 2$, $\delta = 4$, $a_1 = 5, a_2 = 16, a_3 = 18, a_4 = 8, a_5 = 10, a_6 = 23, a_7 = 12, a_8 = 21$.

Solution According to Algorithm 3, the job with the largest normal processing time is J_6 and process it to the first position (the remaining jobs are scheduled in any order). Since $\delta < \beta$, we have $C_{[1]} = 23$, $C_{[n]} = 33.21$, $D^* = \frac{\lambda C_{[1]} + \beta C_{[n]} - (\beta+\lambda)\widehat{d_1}}{\beta} = 24.61$ and maximum $Y^*(\widehat{d_1}) = \frac{\lambda(\delta-\beta)}{\beta} C_{[1]} + \delta C_{[n]} + \left(\lambda + \gamma - \delta - \frac{\delta\lambda}{\beta}\right)\widehat{d_1} = 96.62$.

3.4. The $\widehat{d_1}$ and D are unknown

In this case, d_1 and D are unknown, and we should find the optimal the starting time, size of the due-window and the job schedule such that $\max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\}$ is minimized. Assume that all jobs are arrived at time 0. Without loss of generality, all jobs begin processing at time 0, similarly, we have the following *LP*.

$$\begin{aligned} &Min\ Y \\ &s.t. \begin{cases} Z \ge \lambda E_{[1]} + \gamma\widehat{d_1} + \delta D = (\lambda+\gamma)\widehat{d_1} - \lambda C_{[1]} + \delta D \\ Z \ge \beta T_{[n]} + \gamma\widehat{d_1} + \delta D = \beta C_{[n]} + (\delta-\beta) D + (\gamma-\beta)\widehat{d_1} \\ \widehat{d_1}, D \ge 0. \end{cases} \end{aligned} \tag{9}$$

Similar to Subsection 3.2, we only consider the case $D \le C_{[n]}$ and $\widehat{d_2} \le C_{[n]}$.

Case 1: If $\gamma > \beta$, we have $\gamma - \beta > 0$. Assuming that D is known, we first find the optimal the position of the due-window (i.e., $\widehat{d_1}$). Since $-\lambda C_{[1]} + \delta D$ and $\beta C_{[n]} + (\delta - \beta) D$ are constants, $(\lambda + \gamma)\widehat{d_1}$ and $(\gamma - \beta)\widehat{d_1}$ are both greater than 0; hence, we have $\widehat{d_1^*} = 0$. At this time, there is no early job but only tardy jobs, yielding $Y^*(D) = \beta T_{[n]} + \gamma\widehat{d_1} + \delta D = \beta C_{[n]} + (\delta - \beta) D$.

If $\delta \ge \beta$, we have $\delta - \beta \ge 0$. Because $\beta C_{[n]}$ is a constant and independent of D, it is necessary to take the minimum value of D, i.e., $\widehat{d_1^*} = 0$. In this case, yielding $\widehat{d_1^*} = \widehat{d_2^*} = D^* = 0$, and $Y^* = \beta C_{[n]}$.

If $\delta < \beta$, we have $\delta - \beta < 0$. Similarly, it is necessary to take the maximum value of the size of the due-window, i.e., $D^* = C_{[n]}$. In this case, yielding $\widehat{d_1^*} = 0$, $\widehat{d_2^*} = C_{[n]}$, $D^* = C_{[n]}$, and $Y^* = \delta C_{[n]}$.

Case 2: If $\gamma \leq \beta$, we have $\gamma - \beta \leq 0$. Moreover, if $C_{[n]} - C_{[1]} < D \leq C_{[n]}$, we have $0 \leq \widehat{d_1} < C_{[1]}$. In this case, there is no early job but only tardy jobs. Assuming that $\widehat{d_2}$ does not change, find the optimal position of the due-window and let its optimal cost function be $\beta T_{[n]} + \gamma\widehat{d_1} + \delta D = \beta C_{[n]} + (\delta - \beta)\widehat{d_2} + (\gamma - \delta)\widehat{d_1}$. Because $\beta C_{[n]} + (\delta - \beta)\widehat{d_2}$ is a constant. Then, according to the size relationship between γ and δ, the optimal value of $\widehat{d_1}$ can be obtained.

If $\gamma \geq \delta$, we have $\gamma - \delta \geq 0$. It is optimal to take the minimum value of the starting time of $\widehat{d_1}$, i.e., $\widehat{d_1^*} = 0$. Because there is no early job, in order to minimize the objective function, the tardiness penalty is minimized. In other words, we have $\widehat{d_2^*} = C_{[n]}$. In this case, $\widehat{d_1^*} = 0$, $\widehat{d_2^*} = C_{[n]}$, $D^* = C_{[n]}$, and the optimal cost function is $Y^* = \delta C_{[n]}$.

If $\gamma < \delta$, we have $\gamma - \delta < 0$. It is optimal to take the maximum value of the starting time of $\widehat{d_1}$, i.e. $\widehat{d_1^*} = C_{[1]}$. Similarly, we have $\widehat{d_2^*} = C_{[n]}$.

For the case of $D \leq C_{[n]} - C_{[1]}$, there may be both earliness and tardiness penalties, so set the earliness and tardiness penalties equal to each other, we have

$$\lambda\left(\widehat{d_1} - C_{[1]}\right) = \beta\left(C_{[n]} - D - \widehat{d_1}\right),$$

we have

$$\widehat{d_1^*} = \frac{\lambda C_{[1]} + \beta C_{[n]} - \beta D}{\lambda + \beta},$$

the cost function

$$Y^*(D) = \frac{\lambda(\gamma - \beta)}{\lambda + \beta} C_{[1]} + \frac{\beta(\lambda + \gamma)}{\lambda + \beta} C_{[n]} + \left(\delta - \frac{\beta(\lambda + \gamma)}{\lambda + \beta}\right) D.$$

Because $\frac{\lambda(\gamma - \beta)}{\lambda + \beta} C_{[1]} + \frac{\beta(\lambda + \gamma)}{\lambda + \beta} C_{[n]}$ is a constant, we should discuss the optimal value of D. If $\delta \geq \frac{\beta(\lambda + \gamma)}{\lambda + \beta}$, we have $\delta - \frac{\beta(\lambda + \gamma)}{\lambda + \beta} \geq 0$, so we take the minimum value of D, i.e., $D^* = 0$; then, the starting time and end time of the due-window are both equal to $\frac{\lambda C_{[1]} + \beta C_{[n]}}{\lambda + \beta}$ and the cost function is $\frac{\lambda(\gamma - \beta)}{\lambda + \beta} C_{[1]} + \frac{\beta(\lambda + \gamma)}{\lambda + \beta} C_{[n]}$; if $\delta < \frac{\beta(\lambda + \gamma)}{\lambda + \beta}$, we have $\delta - \frac{\beta(\lambda + \gamma)}{\lambda + \beta} < 0$, so we take the maximum value of D, i.e., $D = C_{[n]} - C_{[1]}$; then $\widehat{d_1^*} = C_{[1]}$ and $\widehat{d_2^*} = C_{[n]}$; and the cost function is $(\gamma - \delta) C_{[1]} + \delta C_{[n]}$.

Algorithm 4

Step 1. Find the job with the largest normal processing time (i.e., $a_{\max} = \max\left\{a_j | j = 1, 2, \ldots, n\right\}$) and process it to the first position (the remaining jobs are scheduled in any order).

Step 2. If $\gamma > \beta$;

If $\delta \geq \beta$, setting $\widehat{d_1^*} = \widehat{d_2^*} = D^* = 0$, $Y^* = \beta C_{[n]}$.

Otherwise, set $\widehat{d_1^*} = 0$, $\widehat{d_2^*} = C_{[n]}$, $D^* = C_{[n]}$, and $Y^* = \delta C_{[n]}$.

If $\gamma \leq \beta$;

If $\gamma < \delta$, setting $\widehat{d_1^*} = 0$, $\widehat{d_2^*} = C_{[n]}$, $D^* = C_{[n]}$ and $Y^* = \delta C_{[n]}$;

Otherwise, if $\delta \geq \frac{\beta(\lambda + \gamma)}{\lambda + \beta}$, setting $\widehat{d_1^*} = \widehat{d_2^*} = \frac{\lambda C_{[1]} + \beta C_{[n]}}{\lambda + \beta}$ and $Y^* = \frac{\lambda(\gamma - \beta)}{\lambda + \beta} C_{[1]} + \frac{\beta(\lambda + \gamma)}{\lambda + \beta} C_{[n]}$.

Otherwise, set $\widehat{d_1^*} = C_{[1]}$, $\widehat{d_2^*} = C_{[n]}$ and $Y^* = (\gamma - \delta) C_{[1]} + \delta C_{[n]}$.

Theorem 4. *Algorithm 4 solves*

$$1|CONW, PLSPT| \max_{1 \leq j \leq n} \left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\}$$

in $O(n)$ time.

Numerical Example 4.

Consider a 8-job problem, where $s = 0$, $\theta = 0.03$, $\lambda = 3, \beta = 5, \gamma = 2, \delta = 4$, $a_1 = 5, a_2 = 16, a_3 = 18, a_4 = 8, a_5 = 10, a_6 = 23, a_7 = 12, a_8 = 21$.

Solution According to Algorithm 4, the job with the largest normal processing time is J_6 and process it to the first position (the remaining jobs are scheduled in any order). Since $\gamma < \beta$ and $\gamma < \delta$, we have $C_{[1]} = 23$, $C_{[n]} = 33.21$, $\widehat{d_1^*} = 0$, $D^* = C_{[n]} = 33.21$ and maximum $Y^* = \delta C_{[n]} = 132.82$

4. Computational experiments

In order to verify the effectiveness of Algorithms 1–4 for problem, problem instances are generated randomly. Visual Studio 2022 was used to code the Algorithm 1–4. For each problem size, 20 instances were generated and solved on a PC with an Intel(R) Core (TM) I7-10750H 2.6 GHz CPU memory of 16.00 GB RAM. The characteristics of the instances are as follows:

1) $n = 100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800, 850, 900, 950, 1000$;
2) $\theta = 0.00001$;
3) a_j is a uniformly distributed over [1, 100] such that $\theta\left(s + \sum_{j=1}^{n} a_j - a_{\min}\right) < 1$ ($a_{\min} = \min\left\{a_j\right\}, j = 1, 2, ..., n$);
4-1) If $\widehat{d_1}$ and D are known, $\widehat{d_1} = \frac{1}{2}[\frac{1}{\theta} - \frac{1}{\theta}\prod_{i=1}^{n}(1 - \theta a_i)]$ and $D = \frac{1}{4}[\frac{1}{\theta} - \frac{1}{\theta}\prod_{i=1}^{n}(1 - \theta a_i)]$ (see Algorithm 1);
4-2) If D is known, $D = \frac{1}{4}[\frac{1}{\theta} - \frac{1}{\theta}\prod_{i=1}^{n}(1 - \theta a_i)]$ (see Algorithm 2);
4-3) If $\widehat{d_1}$ is known, $\widehat{d_1} = \frac{1}{2}[\frac{1}{\theta} - \frac{1}{\theta}\prod_{i=1}^{n}(1 - \theta a_i)]$ (see Algorithm 3);
5) The coefficients λ, β, γ and δ are uniformly distributed over [1,10].

The computational experiments of Algorithms 1–4 are summarized as follows. The maximum and average CPU time (ms) required to find the optimal solutions are given in Table 1. From Table 1, we can observe that Algorithms 1–4 are very efficient and fast, and the CPU time of Algorithms 1–4 increases steady as n increases from 100 to 1000.

5. Conclusions

In this article, we investigated the minmax minimization with $CONW$ assignment and $PLSPT$. The aim was to minimize $\max_{1\le j\le n}\left\{\max\left\{\lambda E_j, \beta T_j\right\} + \gamma\widehat{d_1} + \delta D\right\}$. A polynomial algorithm was proposed for scenarios in which d_1 and D are known or not. Future research may focus on minmax scheduling with resource allocation, investigate the problems with general deterioration effects, or study the minmax scheduling with learning effects (see [23–27].

Table 1. CPU time of Algorithms 1–4 (ms).

	Algorithm 1		Algorithm 2		Algorithm 3		Algorithm 4	
jobs (n)	Max	Mean	Max	Mean	Max	Mean	Max	Mean
100	1	0.65	1	0.45	2	0.25	1	0.15
150	1	0.25	0	0	3	0.15	0	0
200	1	0.25	1	0.05	0	0	1	0.05
250	0	0	1	0.05	0	0	3	0.2
300	1	0.1	1	0.05	3	0.2	0	0
350	1	0.05	1	0.15	1	0.05	1	0.45
400	1	0.1	1	0.35	3	0.45	4	1.55
450	3	0.6	3	0.65	1	0.45	7	2.1
500	6	1.45	4	1.2	4	1.25	4	1.55
550	4	1.6	5	1.55	3	1.15	5	1.6
600	7	1.6	3	1.1	5	1.7	5	1.6
650	3	1.45	4	1	2	1.15	4	1.8
700	5	1.25	6	1.3	3	1.25	5	1.65
750	4	1.55	3	1.1	3	1.2	5	2
800	3	1.3	2	1.05	3	1.1	4	1.4
850	5	1.95	6	1.85	3	1.75	5	1.7
900	3	1.4	5	1.95	4	1.55	4	1.4
950	8	2.25	6	1.45	4	1.45	5	1.9
1000	2	1.11	3	1.2	4	1.1	3	1.1

Acknowledgments

This Work was supported by LiaoNing Revitalization Talents Program (Grant No. XLYC2002017) and Natural Science Foundation of LiaoNing Province, China (Grant No. 2020-MS-233).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. Y. Y. Lu, Research on no-idle permutation flowshop scheduling with time-dependent learning effect and deteriorating jobs, *Appl. Math. Modell.*, **40** (2016), 3447–3450. https://doi.org/10.1016/j.apm.2015.09.081
2. F. Liu, J. Yang, Y. Y. Lu, Solution algorithms for single-machine group scheduling with ready times and deteriorating jobs, *Eng. Optim.*, **51** (2019), 862–874. https://doi.org/10.1080/0305215X.2018.1500562
3. S. Gawiejnowicz, *Models and Algorithms of Time-Dependent Scheduling*, Springer-Berlin, 2020. https://doi.org/10.1007/978-3-662-59362-2

4. X. Huang, Bicriterion scheduling with group technology and deterioration effect, *J. Appl. Math. Comput.*, **60** (2019), 455–464. https://doi.org/10.1007/s12190-018-01222-1
5. D. W. Li, X. W. Lu, Parallel-batch scheduling with deterioration and rejection on a single machine, *Appl. Math. J. Chin. Univ.*, **35** (2020), 141–156. https://doi.org/10.1007/s11766-020-3624-2
6. X. X. Liang, M. Liu, Y. B. Feng, J. B. Wang, L. S. Wen, Solution algorithms for single-machine resource allocation scheduling with deteriorating jobs and group technology, *Eng. Optim.*, **52** (2020), 1184–1197. https://doi.org/10.1080/0305215X.2019.1638920
7. X. Huang, N. Yin, W. W. Liu, J. B. Wang, Common due window assignment scheduling with proportional linear deterioration effects, *Asia-Pacific J. Oper. Res.*, **37** (2020), 1950031. https://doi.org/10.1142/S0217595919500313
8. G. Mosheiov, A common due-date assignment problem on parallel identical machines, *Comput. Oper. Res.*, **28** (2001), 719–732. https://doi.org/10.1016/S0305-0548(99)00127-6
9. G. Mosheiov, A. Sarig, Minmax scheduling problems with a common due-window, *Comput. Oper. Res.*, **36** (2009), 1886–1892. https://doi.org/10.1016/j.cor.2008.06.001
10. E. Gerstl, G. Mosheiov, Minmax due-date assignment with a time window for acceptable lead-times, *Ann. Oper. Res.*, **211** (2013), 167–177. https://doi.org/10.1007/s10479-013-1458-5
11. G. Mosheiov, A due-window determination in minmax scheduling problems, *INFOR: Inf. Syst. Oper. Res.*, **39** (2001), 107–123. https://doi.org/10.1080/03155986.2001.11732429
12. B. Mor, Minmax scheduling problems with common due-date and completion time penalty, *J. Comb. Optim.*, **38** (2019), 50–71. https://doi.org/10.1007/s10878-018-0365-8
13. G. Mosheiov, A. Sarig, V. Strusevich, Minmax scheduling and due-window assignment with position-dependent processing times and job rejection, *4OR*, **18** (2020), 439–456. https://doi.org/10.1007/s10288-019-00418-w
14. W. Liu, X. Hu, X. Y. Wang, Single machine scheduling with slack due dates assignment, *Eng. Optim.*, **49** (2017), 709–717. https://doi.org/10.1080/0305215X.2016.1197611
15. Y. Q. Yin, D. J. Wang, T. C. E. Cheng, *Due Date-Related Scheduling with Two Agents*, Springer-Berlin, 2020. https://doi.org/10.1007/978-981-15-2105-8
16. G. A. Rolin, M. S. Nagano, Structural properties and algorithms for earliness and tardiness scheduling against common due dates and windows: A review, *Comput. Ind. Eng.*, **149** (2020), 106803. https://doi.org/10.1016/j.cie.2020.106803
17. X. Sun, X. N. Geng, T. Liu, Due-window assignment scheduling in the proportionate flow shop setting, *Ann. Oper. Res.*, **292** (2020), 113–131. https://doi.org/10.1007/s10479-020-03653-1
18. S. Zhao, Resource allocation flowshop scheduling with learning effect and slack due window assignment, *J. Ind. Manage. Optim.*, **17** (2021), 2817–2835. https://doi.org/10.3934/jimo.2020096
19. W. Liu, X. Wang, X. Wang, P. Zhao, Due-window assignment scheduling with past-sequence-dependent setup times, *Math. Biosci. Eng.*, **19** (2022), 3110–3126. https://doi.org/10.3934/mbe.2022144
20. X. Wang, W. Liu, L. Li, P. Zhao, R. Zhang, Due date assignment scheduling with positional-dependent weights and proportional setup times, *Math. Biosci. Eng.*, **19** (2022), 5104–5119. https://doi.org/10.3934/mbe.2022238

21. K. I. J. Ho, J. Y. T. Leung, W. D. Wei, Complexity of scheduling tasks with time-dependent execution times, *Inf. Process. Lett.*, **48** (1993), 315–320. https://doi.org/10.1016/0020-0190(93)90175-9

22. J. B. Wang, Z. Q. Xia, Scheduling jobs under decreasing linear deterioration, *Inf. Process. Lett.*, **94** (2005), 63–69. https://doi.org/10.1016/j.ipl.2004.12.018

23. C. C. Wu, W. C. Lee, M. J. Liou, Single-machine scheduling with two competing agents and learning consideration, *Inf. Sci.*, **251** (2013), 136–149. https://doi.org/10.1016/j.ins.2013.06.054

24. C. C. Wu, Y. Yin, S. R. Cheng, Single-machine and two-machine flowshop scheduling problems with truncated position-based learning functions, *J. Oper. Res. Soc.*, **64** (2013), 147–156. https://doi.org/10.1057/jors.2012.46

25. W. C. Yeh, P. J. Lai, W. C. Lee, M. C. Chuang, Parallel-machine scheduling to minimize makespan with fuzzy processing times and learning effects, *Inf. Sci.*, **269** (2014), 142–158. https://doi.org/10.1016/j.ins.2013.10.023

26. J. B. Wang, D. Y. Lv, J. Xu, P. Ji, F. Li, Bicriterion scheduling with truncated learning effects and convex controllable processing times, *Int. Trans. Oper. Res.*, **28** (2021), 1573–1593. https://doi.org/10.1111/itor.12888

27. D. Y. Lv, J. B. Wang, Study on resource-dependent no-wait flow shop scheduling with different due-window assignment and learning effects, *Asia-Pacific J. Oper. Res.*, **38** (2021), 2150008. https://doi.org/10.1142/S0217595921500081

AIMS Press