Due-date assignment and single machine scheduling with deteriorating jobs

T.C.E. Cheng[1], Liying Kang[1,2], C.T. Ng[1]

[1]Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

[2]Department of Mathematics, Shanghai University, Shanghai 200436, China

Abstract

We study a scheduling problem with deteriorating jobs, i.e., jobs whose processing times are an increasing function of their start times. We consider the case of a single machine and linear job-independent deterioration. The problem is to determine an optimal combination of the due date and schedule so as to minimize the sum of due-date, earliness and tardiness penalties. We give an $O(n\ logn)$ time algorithm to solve this problem.

Keywords: Single machine scheduling; Due-date; Deteriorating jobs.

Introduction

Browne and Yechiali[1] introduced a scheduling problem with deteriorating jobs. In this problem, the job processing time is a non-decreasing, start time dependent linear function. Deterioration in processing time may occur when the machine gradually loses efficiency in the course of processing jobs. At the beginning, the machine is assumed to be at its highest level of efficiency. The efficiency loss is reflected in the fact that a job which is processed later in time has a longer processing time.

A practical example of the problem in which the job processing time is an increasing function of its start time can be found in steel production[2]. It

describes the ingot preheating process in steel mills. After heating in a heat converter, hot liquid metal is poured into steel ladles and next into ingot moulds, where it solidifies. Then, after the ingot stripper process, ingots are appropriately segregated into batches and then transported to the soaking pits, where the process consists of preheating ingot batches up to the required temperature. The ingots are then hot-rolled on the blooming mill. Each ingot batch consists of several ingots and is preheated by gas in a separate soaking pit. The preheating time of the ingots depends on their starting temperature at the beginning of the preheating process. The longer ingots wait for the start of the preheating process, the lower their temperature drops and the longer the preheating process lasts.

Scheduling in the settings described above is known as scheduling *deteriorating* jobs. The processing time of a deteriorating job is given by $f_i(t)$, where $f_i(t)$ is a non-decreasing function of the job start time t. Most of the relevant studies[2–8] are confined to linear deterioration. In its standard form, linear deterioration is given by $f_i(t) = a_i + b_i t$, where a_i is the 'normal processing time', which is the length of time required to complete the job if it is scheduled first ($t = 0$), and b_i is the job dependent deterioration rate, which determines the job's (actual) processing time at $t > 0$.

In this paper we study the case where the job-independent deterioration rates are identical for all jobs, i.e., $b_i = b$. We believe that this is a very realistic setting, particularly in the case of scheduling with deteriorating machines, when all processing times increase by a common factor caused by the machines. Thus, we focus on the case $f_i(t) = a_i + bt$.

In addition, we study the scheduling of deteriorating jobs in the context of the Common Due-Date Problem (CDDP), which deals with job scheduling on a single machine in a just-in-time production environment[9–13]. Applications of the CDDP in real-life situations can readily be found. Baker and Scudder[9] observed that treating due dates as decision variables reflects the practice in

some shops of setting due-dates internally, as targets to guide the progress of shop floor activities. Prescribing a common due-date might represent a situation where several items constitute a single customer's order, or it might reflect an assembly environment in which the components should all be ready at the same time in order to avoid staging delays.

The per unit costs involved in this paper (due date, earliness, and tardiness) are all linear. Panwalkar et al.[13] argued that linear cost functions present a case that is more tractable than that occurring with nonlinear costs. The insight gained from the linear model may be useful when approaching the nonlinear model. Also it is likely that in practice the estimation of costs may introduce more inaccuracies than those occurring with the assumption of linear costs. All of these costs can be regarded as opportunity costs.

Problem formulation

Let n jobs $\mathcal{J} = \{1, 2, \ldots, n\}$ and a single machine that can handle only one job at a time be given. Job processing times deteriorate linearly in relation to their start times. In the remainder of this paper, we denote the normal processing time of job i by a_i, and its actual processing time if processed at time t by $p_i(t) = a_i + bt, i = 1, 2, \ldots, n$. For any given schedule σ, let

$$
\begin{aligned}
s_i(\sigma) &= \text{start time of job } i, \\
d &= \text{common due date}, \\
p_i(\sigma) &= a_i + bs_i(\sigma), \text{ actual processing time of job } i, \\
C_i(\sigma) &= \text{completion time of job } i, \\
E_i(\sigma) &= \max\{0, d - C_i(\sigma)\}, \text{ earliness of job } i, \\
T_i(\sigma) &= \max\{0, C_i(\sigma) - d\}, \text{ tardiness of job } i, \\
F(d, \sigma) &= \sum(\alpha E_i(\sigma) + \beta T_i(\sigma) + \gamma d), \text{ total penalty function, where } \alpha, \beta,
\end{aligned}
$$

and γ are the unit earliness, tardiness and due-date penalty, respectively.

A job sequence σ: $\sigma(1), \ldots, \sigma(n)$, where $\sigma(j)$ denotes the job in position j of σ, is called V**-shaped** with respect to the a_i-values if there exist no three indices $i < j < k$ with $a_{\sigma(i)} < a_{\sigma(j)} > a_{\sigma(k)}$. It is not difficult to see that a sequence is V-shaped if and only if $a_{\sigma(1)} \geq a_{\sigma(2)} \geq \ldots \geq a_{\sigma(k)}$ and $a_{\sigma(k)} \leq a_{\sigma(k+1)} \leq \ldots \leq a_{\sigma(n)}$ for some $1 \leq k \leq n$.

In this paper, we consider the problem of determining an optimal combination of due date d^* and schedule σ^* so that $F(d, \sigma)$ is minimized. Using the three field notation of Graham et al.[14], the problem can be denoted as $1|p_i(s_i) = a_i + bs_i| \sum(\alpha E_i + \beta T_i + \gamma d)$.

In Section 3, we give some preliminary analysis of the problem. In Section 4, we develop a polynomial-time algorithm that finds an optimal solution for the problem.

Preliminary analysis

We first present some elementary results.

Property 1. There exists an optimal schedule in which the machine is not idle between the processing of the jobs.

Proof. Similar to the proof of Lemma 7.1 in 15.

Property 2. For any specified sequence σ, there exists an optimal due date equal to $C_{\sigma(K)}$, where K is the smallest integer greater than or equal to $\frac{n\beta - n\gamma}{\alpha + \beta}$ and exact K jobs will be nontardy.

Proof. We first show that for any specified schedule σ, d^* is equal to the completion time of some job.

Consider a given schedule σ and d with $C_{\sigma(i)} < d < C_{\sigma(i+1)}$, and let F be the corresponding objective value. Define $x = d - C_{\sigma(i)}$ and $y = C_{\sigma(i+1)} - d$. Let F' and F'' be the objective value for $d = C_{\sigma(i)}$ and $d = C_{\sigma(i+1)}$, respectively. Then,

$$F' = F + x(n-i)\beta - xi\alpha - n\gamma x \tag{1}$$

and

$$F'' = F - y(n-i)\beta + yi\alpha + n\gamma y. \tag{2}$$

Thus, we have $F' \leq F$ if $(n-i)\beta \leq i\alpha + n\gamma$, and $F'' < F$ otherwise. This implies that for any specified schedule σ, d^* is equal to the completion time of some job $\sigma(k)$. Assume that $d^* = C_{\sigma(k)}$, and let Z be the optimal solution. Applying (1) and (2) to the situation $x = C_{\sigma(k)} - C_{\sigma(k-1)}$ and $y = C_{\sigma(k+1)} - C_{\sigma(k)}$, respectively, we conclude that $\frac{n\beta - n\gamma}{\alpha+\beta} \leq k \leq \frac{n\beta - n\gamma}{\alpha+\beta} + 1$. This implies that $d^* = C_{\sigma(K)}$. The result follows. ■

So, the total penalty is equal to $F(C_{\sigma(K)}, \sigma)$. Introducing $C_{\sigma(K)} = p_{\sigma(1)} + p_{\sigma(2)} + \ldots + p_{\sigma(K)}$, we get

$$F(C_{\sigma(K)}, \sigma) = \sum_{j=1}^{K} (\alpha(j-1) + n\gamma) p_{\sigma(j)} + \sum_{j=K+1}^{n} \beta(n+1-j) p_{\sigma(j)}. \tag{3}$$

Property 3. For any optimal schedule σ, $a_{\sigma(K+1)} \leq \ldots \leq a_{\sigma(n)}$.

Proof. Assume that the schedule σ in which $a_{\sigma(i)} > a_{\sigma(i+1)}$ $(i > K)$ is optimal. The schedule σ' is obtained from σ by interchanging the jobs in the ith and $(i+1)$th positions of σ. Then,

$$\begin{aligned} C_{\sigma(i+1)} &= a_{\sigma(i+1)} + (1+b)a_{\sigma(i)} + (1+b)^2 s_{\sigma(i)}, \\ C_{\sigma'(i+1)} &= a_{\sigma(i)} + (1+b)a_{\sigma(i+1)} + (1+b)^2 s_{\sigma(i)}. \end{aligned}$$

So, $C_{\sigma(i+1)} - C_{\sigma'(i+1)} = b(a_{\sigma(i)} - a_{\sigma(i+1)}) > 0$. Combining this with (3), the difference between the value of $F(d, s)$ for both schedules is as follows:

$$\begin{aligned} & F(C_{\sigma(K)}, \sigma) - F(C_{\sigma'(K)}, \sigma') \\ > \; & \beta(n+1-i)p_{\sigma(i)} + \beta(n-i)p_{\sigma(i+1)} - \beta(n+1-i)p_{\sigma'(i)} - \beta(n-i)p_{\sigma'(i+1)} \\ = \; & \beta(n+1-i)(a_{\sigma(i)} + bs_{\sigma(i)}) + \beta(n-i)(a_{\sigma(i+1)} + ba_{\sigma(i)} + b(1+b)s_{\sigma(i)}) \\ & -\beta(n+1-i)(a_{\sigma(i+1)} + bs_{\sigma(i)}) - \beta(n-i)(a_{\sigma(i)} + ba_{\sigma(i+1)} + b(1+b)s_{\sigma(i)}) \\ = \; & \beta(1+(n-i)b)(a_{\sigma(i)} - a_{\sigma(i+1)}) \\ > \; & 0. \end{aligned}$$

This is a contradiction to the optimality of σ. The lemma follows. ■

For notational convenience, we define the following:

$$\begin{aligned} m_i &= \begin{cases} b\sum_{j=i}^{K}(\alpha(j-1)+n\gamma)(1+b)^{j-i} + b\sum_{j=K+1}^{n}\beta(n+1-j)(1+b)^{j-i} \\ \quad \text{for } 2 \le i \le K, \\ b\sum_{j=i}^{n}\beta(n+1-j)(1+b)^{j-i}, \text{ for } K+1 \le i \le n, \end{cases} \qquad (4) \\ g(i) &= \alpha(i-1)+n\gamma+m_{i+1}, \text{ for } i = 1,2,\ldots,K, \\ f(b) &= (\alpha+n\gamma)b-\alpha+bm_3. \end{aligned}$$

It is easily seen from (4) that

$$m_{i+1} = (\alpha i + n\gamma)b + (1+b)m_{i+2} \quad \text{for } 2 \le i < K. \qquad (5)$$

Property 4. (1). If $f(b) = (\alpha+n\gamma)b-\alpha+bm_3 > 0$, then $(\alpha i+n\gamma)b-\alpha+bm_{i+2} > 0$ for $1 \le i \le K-1$.

(2). If $f(b) = (\alpha+n\gamma)b-\alpha+bm_3 < 0$, then $(\alpha i+n\gamma)b-\alpha+bm_{i+2} < 0$ for $1 \le i \le K-1$.

(3). If $f(b) = (\alpha+n\gamma)b-\alpha+bm_3 = 0$, then $(\alpha i+n\gamma)b-\alpha+bm_{i+2} = 0$ for $1 \le i \le K-1$.

Proof. (1) We proceed by induction on i. If $i = 1$, $(\alpha+n\gamma)b-\alpha+bm_3 > 0$, the result follows. Assume that the result holds for the case $i < k$. For the case $i = k$, by the induction hypothesis,

$$(\alpha(k-1)+n\gamma)b-\alpha+bm_{k+1} > 0. \qquad (6)$$

Combining this with

$$m_{k+1} = (\alpha k+n\gamma)b+(1+b)m_{k+2}, \qquad \text{(by (5))}$$

we have

$$\begin{aligned} & (\alpha k+n\gamma)b-\alpha+bm_{k+2} \\ = \; & (\alpha k+n\gamma)b-\alpha+\frac{bm_{k+1}-(\alpha k+n\gamma)b^2}{1+b} \\ > \; & (\alpha k+n\gamma)b-\alpha+\frac{\alpha-(\alpha(k-1)+n\gamma)b-(\alpha k+n\gamma)b^2}{1+b} \qquad \text{(by (6))} \\ = \; & 0. \end{aligned}$$

The result of (1) follows.

(2) and (3). Similar to the proof of (1). ■

Property 5. (1). If $f(b) < 0$, then $g(i)$ is an increasing function of i ($i = 1, 2, \ldots, K$).

(2). If $f(b) > 0$, then $g(i)$ is a decreasing function of i ($i = 1, 2, \ldots, K$).

(3). If $f(b) = 0$, then $g(1) = g(2) = \ldots = g(K)$.

Proof. (1). Since $m_{i+1} = (\alpha i + n\gamma)b + (1 + b)m_{i+2}$ for $1 \leq i \leq K - 1$, we have

$$\begin{aligned}
& g(i+1) - g(i) \\
= \; & (\alpha i + n\gamma + m_{i+2}) - (\alpha(i-1) + n\gamma + m_{i+1}) \\
= \; & \alpha + m_{i+2} - m_{i+1} \\
= \; & \alpha + m_{i+2} - ((\alpha i + n\gamma)b + (1+b)m_{i+2}) \\
= \; & -((\alpha i + n\gamma)b - \alpha + bm_{i+2}) \\
> \; & 0. \qquad \text{(By Property(4))}
\end{aligned}$$

This completes the proof of (1).

(2) and (3). Similar to the proof of (1). ■

Property 6. (1). If $f(b) \geq 0$, then there exists an optimal schedule σ such that $a_{\sigma(1)} \leq a_{\sigma(2)} \leq \ldots \leq a_{\sigma(K)}$.

(2). If $f(b) < 0$, then for any optimal schedule σ, $a_{\sigma(1)} \geq a_{\sigma(2)} \geq \ldots \geq a_{\sigma(K)}$.

Proof. (1). Assume that the schedule σ_1 in which $a_{\sigma_1(i)} > a_{\sigma_1(i+1)}$ ($1 \leq i \leq K-1$) is optimal. Let σ_1' be the schedule derived from σ_1 by swapping i and $i+1$. Then,

$$\begin{aligned}
& F(C_{\sigma_1(K)}, \sigma_1) - F(C_{\sigma_1'(K)}, \sigma_1') \\
= \; & (\alpha(i-1) + n\gamma)p_{\sigma_1(i)} + (\alpha i + n\gamma)p_{\sigma_1(i+1)} - \\
& (\alpha(i-1) + n\gamma)p_{\sigma_1'(i)} - (\alpha i + n\gamma)p_{\sigma_1'(i+1)} + bm_{i+2}(a_{\sigma_1(i)} - a_{\sigma_1(i+1)})
\end{aligned}$$

$$
\begin{aligned}
&= (\alpha(i-1)+n\gamma)(a_{\sigma_1(i)}+bs_{\sigma_1(i)})+(\alpha i+n\gamma)(a_{\sigma_1(i+1)}+ba_{\sigma_1(i)}+b(1+b)s_{\sigma_1(i)}) \\
&\quad -(\alpha(i-1)+n\gamma)(a_{\sigma_1(i+1)}+bs_{\sigma_1(i)})-(\alpha i+n\gamma)(a_{\sigma_1(i)}+ba_{\sigma_1(i+1)}+b(1+b)s_{\sigma_1(i)}) \\
&\quad +bm_{i+2}(a_{\sigma_1(i)}-a_{\sigma_1(i+1)}) \\
&= ((\alpha i+n\gamma)b-\alpha+bm_{i+2})(a_{\sigma_1(i)}-a_{\sigma_1(i+1)}) \\
&\geq 0. \qquad \text{(By Property 4)}
\end{aligned}
$$

So, σ_1' is an optimal schedule. Proceeding as above, we can obtain an optimal schedule σ such that $a_{\sigma(1)} \leq a_{\sigma(2)} \leq \ldots \leq a_{\sigma(K)}$.

(2). Assume that the schedule σ in which $a_{\sigma(i)} < a_{\sigma(i+1)}$ $(1 \leq i \leq K-1)$ is optimal. Let σ' be the schedule derived from σ by swapping i and $i+1$. Then,

$$
\begin{aligned}
&F(C_{\sigma(K)},\sigma)-F(C_{\sigma'(K)},\sigma') \\
&= ((\alpha i+n\gamma)b-\alpha+bm_{i+2})(a_{\sigma(i)}-a_{\sigma(i+1)}) \\
&> 0. \qquad \text{(By Property 4)}
\end{aligned}
$$

This is a contradiction. This completes the proof. ■

The following theorem is easily established from Property 3 and Property 6.

Theorem 7. (1). If $f(b) \geq 0$, then there exists an optimal schedule σ such that $a_{\sigma(1)} \leq a_{\sigma(2)} \leq \ldots \leq a_{\sigma(K)}$ and $a_{\sigma(K+1)} \leq a_{\sigma(K+2)} \leq \ldots \leq a_{\sigma(n)}$.

(2). If $f(b) < 0$, then for any optimal schedule σ, $a_{\sigma(1)} \geq a_{\sigma(2)} \geq \ldots \geq a_{\sigma(K)}$ and $a_{\sigma(K+1)} \leq \ldots \leq a_{\sigma(n)}$. That is, any optimal schedule for the problem must be V-shaped with respect to the normal processing times.

A polynomial-time algorithm

First, we sort and re-label the n jobs so that they are in non-increasing order of their normal processing times, namely $a_1 \geq a_2 \geq \ldots \geq a_n$. In the following, we present an $O(n\ logn)$ algorithm for $1|p_i(s_i) = a_i + bs_i| \sum(\alpha E_i + \beta T_i + \gamma d)$.

Algorithm 1.

Step 1: Initialization.

$i = 1, j = 1, k = n, S_1 = S_2 = \emptyset, m_{n+1} = 0, K = \lceil \frac{n\beta - n\gamma}{\alpha + \beta} \rceil$.

Step 2. If $K = 1$, go to Step 5.

Step 3: Compute the values of $m_{j+1}, m_{j+2}, m_{K+1}$.

Step 4: Compute $f(b) = (\alpha + n\gamma)b - \alpha + bm_{j+2}$. If $f(b) \geq 0$, go to Step 8; otherwise, go to Step 5.

Step 5: Compute $L_{j,k} = \alpha(j-1) + n\gamma + m_{j+1} - \beta(n+1-k) - m_{k+1}$. If $L_{j,k} \geq 0$, set $k := k-1, m_{k+1} := b\beta(n-k) + (1+b)m_{k+1}, S_2 := S_2 \cup \{i\}$; otherwise, set $j := j+1, m_{j+1} := \frac{m_{j+1} - (\alpha(j-1) + n\gamma)b}{1+b}$, $S_1 := S_1 \cup \{i\}$.

Step 6: $i := i + 1$

Step 7: If $j = K + 1$, set $S_2 := \mathcal{J} - S_1$. Let σ be the schedule obtained by arranging the jobs in non-increasing order of the job normal processing times in S_1, followed by arranging the jobs in non-decreasing order of the job normal processing times in S_2, and $d^* = C_{\sigma(K)}$. STOP.

If $k = K$, set $S_1 := \mathcal{J} - S_2$. Let σ be the schedule obtained by arranging the jobs in non-increasing order of the job normal processing times in S_1, followed by arranging the jobs in non-decreasing order of the job normal processing times in S_2, and $d^* = C_{\sigma(K)}$. STOP. Otherwise, go to Step 5.

Step 8: $j := K$

Step 9: Compute $L_{j,k} = \alpha(j-1) + n\gamma + m_{j+1} - \beta(n+1-k) - m_{k+1}$. If $L_{j,k} \geq 0$, set $k := k-1, m_{k+1} := b\beta(n-k) + (1+b)m_{k+1}, S_2 := S_2 \cup \{i\}$; otherwise, set $j := j-1, m_{j+1} := (j\alpha + n\gamma)b + (1+b)m_{j+1}, S_1 := S_1 \cup \{i\}$.

Step 10: $i := i + 1$

Step 11: If $j = 0$, set $S_2 := \mathcal{J} - S_1$. Let σ be the schedule obtained by arranging the jobs in non-decreasing order of the job normal processing times in S_1, followed by arranging the jobs in non-decreasing order of the job normal processing times in S_2, and $d^* = C_{\sigma(K)}$. STOP.

If $k = K$, set $S_1 := \mathcal{J} - S_2$. Let σ be the schedule obtained by arranging the jobs in non-decreasing order of the job normal processing times in S_1, followed by the jobs in non-decreasing order of the job normal processing times in S_2, and $d^* = C_{\sigma(K)}$. STOP. Otherwise, go to Step 9.

To determine the computation complexity of Algorithm 1, we note that Steps 3, 5 and 9 can be completed in $O(n)$ time, while Steps 7, 11 can be completed in $O(n\ logn)$ time. Hence, the overall time complexity of the algorithm is $O(n\ logn)$.

Property 8. Let σ be an optimal schedule and $L_{i,j} = \alpha(i-1)+n\gamma+m_{i+1}-\beta(n+1-j)-m_{j+1}$ $(1 \le i \le K, K+1 \le j \le n)$. We have the following properties:

(1). If $L_{i,j} > 0$, then $a_{\sigma(i)} \le a_{\sigma(j)}$; if $L_{i,j} < 0$, then $a_{\sigma(i)} \ge a_{\sigma(j)}$.

(2). If $f(b) < 0$ and $L_{i,j} = 0$, then $a_{\sigma(k)} \ge \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $k < i$ or $k > j$ and $a_{\sigma(k)} \le \min\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $i < k < j$.

(3). If $f(b) = 0$ and $L_{i,j} = 0$, then $a_{\sigma(j-1)} \le a_{\sigma(k)} \le a_{\sigma(j+1)}$ for $1 \le k \le K$.

(4). If $f(b) > 0$ and $L_{i,j} = 0$, then $a_{\sigma(k)} \ge \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $K \ge k > i$ or $n \ge k > j$ and $a_{\sigma(k)} \le \min\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $1 \le k < i$ or $K+1 \le k < j$.

Proof. (1). If $L_{i,j} > 0$, suppose to the contrary that $a_{\sigma(i)} > a_{\sigma(j)}$. Let σ' be the schedule derived from σ by swapping i and j. Then, $F(C_{\sigma(K)}, \sigma) - F(C_{\sigma'(K)}, \sigma') = L_{i,j}(a_{\sigma(i)} - a_{\sigma(j)}) > 0$. This is a contradiction. So, $a_{\sigma(i)} \le a_{\sigma(j)}$. Similar to the above, we have $a_{\sigma(i)} \ge a_{\sigma(j)}$ if $L_{i,j} < 0$.

(2). Since $L_{i,j} = 0$, then $L_{i,k} > 0$ if $k > j$. By (1), $a_{\sigma(k)} \ge \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $k > j$. By Property 5, $g(i-1) < g(i)$, combining this with $L_{i,j} = 0$, we have $L_{k,j} < 0$ if $k < i$. By (1), $a_{\sigma(k)} \ge \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $k < i$. Similarly, we have $a_{\sigma(k)} \le \min\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $i < k < j$.

(3). If $f(b) = 0$ and $L_{i,j} = 0$, by Property 5, $L_{k,j+1} > 0$ and $L_{k,j-1} < 0$ for $1 \le k \le K$. Then, using (1), we have $a_{\sigma(j-1)} \le a_{\sigma(k)} \le a_{\sigma(j+1)}$ for $1 \le k \le K$.

(4). Similar to the proof of (2). ■

For notational convenience, we define the following properties:

($\mathbf{P_1}$). If $L_{i,j} > 0$, then $a_{\sigma(i)} \leq a_{\sigma(j)}$. Moreover, if $L_{i,j} > 0$ and $a_{\sigma(i)} = a_{\sigma(j)}$, then $\sigma(j) < \sigma(i)$. If $L_{i,j} < 0$, then $a_{\sigma(i)} \geq a_{\sigma(j)}$. Moreover, if $L_{i,j} < 0$ and $a_{\sigma(i)} = a_{\sigma(j)}$, then $\sigma(i) < \sigma(j)$.

($\mathbf{P_2}$). If $L_{i,j} = 0$, then $a_{\sigma(j)} \geq a_{\sigma(i)}$. Moreover, if $L_{i,j} = 0$ and $a_{\sigma(j)} = a_{\sigma(i)}$, then $\sigma(j) < \sigma(i)$.

($\mathbf{P_3}$). If $L_{i,j} = 0$, then $a_{\sigma(k)} \geq \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $k < i$ or $k > j$ and $a_{\sigma(k)} \leq \min\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $i < k < j$.

($\mathbf{P_4}$). $a_{\sigma(1)} \geq a_{\sigma(2)} \geq \ldots \geq a_{\sigma(K)}$ and $a_{\sigma(K+1)} \leq \ldots \leq a_{\sigma(n)}$. If $a_{\sigma(i)} = a_{\sigma(j)}$, then $\sigma(i) < \sigma(j)$ if $1 \leq i < j < K$, and $\sigma(i) > \sigma(j)$ if $K < i < j \leq n$.

($\mathbf{P_5}$). $a_{\sigma(1)} \leq a_{\sigma(2)} \leq \ldots \leq a_{\sigma(K)}$ and $a_{\sigma(K+1)} \leq \ldots \leq a_{\sigma(n)}$. If $a_{\sigma(i)} = a_{\sigma(j)}$, then $\sigma(i) > \sigma(j)$ if $1 \leq i < j \leq K$ or $K + 1 \leq i < j \leq n$.

($\mathbf{P_6}$). If $L_{i,j} = 0$, then $a_{\sigma(k)} \geq \max\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $K \geq k > i$ or $n \geq k > j$ and $a_{\sigma(k)} \leq \min\{a_{\sigma(i)}, a_{\sigma(j)}\}$ for $1 \leq k < i$ or $K + 1 \leq k < j$.

($\mathbf{P_7}$). If $L_{i,j} = 0$, then $a_{\sigma(j-1)} \leq a_{\sigma(k)} \leq a_{\sigma(j+1)}$ for $1 \leq k \leq K$.

The following property is easily established from Algorithm 1.

Property 9. (1). If $f(b) < 0$, let σ be the schedule derived by Algorithm 1, then σ satisfies properties $(P_1), (P_2), (P_3)$ and (P_4).

(2). If $f(b) > 0$, let σ be the schedule derived by Algorithm 1, then σ satisfies properties $(P_1), (P_2), (P_5)$ and (P_6).

(3). If $f(b) = 0$, let σ be the schedule derived by Algorithm 1, then σ satisfies properties $(P_1), (P_2), (P_5)$ and (P_7).

Property 10. (1). If $f(b) < 0$, then there exists an optimal schedule that satisfies properties $(P_1), (P_2), (P_3)$ and (P_4).

(2). If $f(b) > 0$, then there exists an optimal schedule that satisfies properties

$(P_1), (P_2), (P_5)$ and (P_6).

(3). If $f(b) = 0$, then there exists an optimal schedule that satisfies properties $(P_1), (P_2), (P_5)$ and (P_7).

Proof. (1). Assume that σ is an optimal schedule with properties $(P_1), (P_3)$, and (P_4), this is easily obtained by Theorem 7 and Property 8. If there exist integers i, j with $1 \leq i \leq K, K+1 \leq j \leq n$, such that $L_{i,j} = 0$ and $a_{\sigma(j)} < a_{\sigma(i)}$ or $a_{\sigma(j)} = a_{\sigma(i)}$ and $\sigma(j) > \sigma(i)$, let σ_1 be the schedule derived from σ by swapping i and j. Then, $F(C_{\sigma(K)}, \sigma) - F(C_{\sigma_1(K)}, \sigma_1) = L_{i,j}(a_{\sigma(i)} - a_{\sigma(j)}) = 0$. So, σ_1 is an optimal schedule. By repeating the above procedure, we eventually obtain an optimal schedule σ' with properties $(P_1), (P_2), (P_3)$ and (P_4).

(2) and (3). Similar to the proof of (1). ■

Property 11. (1). If $f(b) < 0$, let σ, σ' be two schedules with properties $(P_1), (P_2), (P_3)$ and (P_4), then $\sigma(i) = \sigma'(i)$ for $1 \leq i \leq n$.

(2). If $f(b) > 0$, let σ, σ' be two schedules with properties $(P_1), (P_2), (P_5)$ and (P_6), then $\sigma(i) = \sigma'(i)$ for $1 \leq i \leq n$.

(3). If $f(b) = 0$, let σ, σ' be two schedules with properties $(P_1), (P_2), (P_5)$ and (P_7), then $\sigma(i) = \sigma'(i)$ for $1 \leq i \leq n$.

Proof. (1). We proceed by induction on i. If $i = 1$ and there exists an integer j_1 $(K < j_1 \leq n)$ such that $L_{1,j_1} = 0$, then $L_{1,j} > 0$ if $j > j_1$. By properties (P_1) and (P_2), $\sigma(1) = \sigma'(1) = n - j_1 + 2$. If $i = 1$ and there exists no integer j $(K < j \leq n)$ such that $L_{1,j} = 0$, then $\sigma(1) = \sigma'(1) = 1$ if $L_{1,n} < 0$. If $L_{1,n} > 0$, let $j_1' = \min\{j | L_{1,j} > 0\}$, then by properties (P_1) and (P_2), $\sigma(1) = \sigma'(1) = n - j_1' + 2$.

Assume that for $i < k$, the result follows. We consider the case $i = k$. If there exists an integer j_k $(K < j_k \leq n)$ such that $L_{k,j_k} = 0$, then by properties (P_3) and (P_4), $\sigma(k) = \max\{\sigma(k-1), \sigma(j_k - 1)\} + 2$. Similarly, $\sigma'(k) = \max\{\sigma'(k-1), \sigma'(j_k - 1)\} + 2$. By the induction hypothesis, $\sigma(i) = \sigma'(i)$

for $1 \leq i \leq k-1$. By properties (P_3) and (P_4), this implies that $\sigma(j_k - 1) = \sigma'(j_k - 1)$. So, $\sigma(k) = \sigma'(k)$. If there exists no integer j $(K < j \leq n)$ such that $L_{k,j} = 0$, let $j_k = n - (\sigma(k-1) - (k-1))$, then $\sigma(k) = \sigma'(k) = \sigma(k-1) + 1$ if $L_{k,j_k} < 0$. If $L_{k,j_k} > 0$, let $j'_k = \min\{j | L_{k,j} > 0\}$, then $\sigma(k) = \sigma'(k) = \sigma(k-1) + (j_k - j'_k + 2)$. The result follows.

(2) and (3). Similar to the proof of (1). ■

The following theorem follows from Property 9, Property 10 and Property 11.

Theorem 12. Algorithm 1 computes in $O(n\ logn)$ time an optimal solution for the problem $1|p_i(s_i) = a_i + bs_i| \sum(\alpha E_i + \beta T_i + \gamma d)$.

Conclusions

This paper studies the problem of setting a common due date and scheduling jobs with linear deterioration of job processing times having a common job-independent deterioration rate on a single machine. Our objective is to minimize the sum of due-date, earliness and tardiness penalties. We show that the optimal solution can be found in $O(n\ logn)$ time.

Future research may focus on scheduling deteriorating jobs in multi-machine settings (parallel machines or shops). Alternatively, one may consider more general non-linear deterioration types. Finally, it will also be interesting to investigate the "mirror" problem in which the job processing times are non-increasing function of their start times.

Acknowledgments

This research was supported in part by The Hong Kong Polytechnic University under grant number G-YW81. The second author was also supported by the National Natural Science Foundation of China under grant number 10101010.

References

1 Brown S and Yechiali U (1990). Scheduling deteriorating jobs on a single processor. Operations Research **38:** 495-498.

2 Ng CT, Cheng TCE, Bachman A and Janiak A (2002). Three scheduling problems with deteriorating jobs to minimize the total completion time. Information Processing Letters **81:** 327-333.

3 Cheng TCE and Ding Q (1998). The complexity of scheduling starting time dependent tasks with release date. Information Processing Letters **65:** 75-79.

4 Mosheiov G (1991). V-shaped policies for scheduling deteriorating jobs. Operations Research **39:** 979-991.

5 Mosheiov G (1994). Scheduling jobs under simple linear deterioration. Computers and Operations Research **21:** 653-659.

6 Cheng TCE, Ding Q and Lin BMT (2004). A concise survey of scheduling with time-dependent processing times. European Journal of Operational Research **152:** 1-13.

7 Alidaee B (1991). Single machine scheduling with nonlinear cost functions. Computers and Operations Research **18:** 317-322.

8 Alidaee B and Womer NK (1999). Scheduling with time dependent processing times: Review and extentions. Journal of the Operational Research Society **50:** 711-720.

9 Baker KR and Scudder GD (1990). Sequencing with earliness and tardiness penalities: a review. Operations Research **38:** 22-36.

10 Gordon VS, Porth JM and Chu CB (2002). A survey of the state-of-art of common due date assignment and scheduling research. European Journal of Operational Research **139:** 1-25.

11 Gordon VS, Porth JM, Chu CB (2002). Due date assignment and scheduling: SLK, TWK and other due date assignment models. Production Planning and Control **13:** 117-132.

12 Cheng TCE, Chen ZL and Shakhlevich NV (2002). Common due date assignment and scheduling with ready times. Computers and Operations Research **29:** 1957-1967.

13 Panwalkar SS, Smith ML and Seidmann (1982). Common due date assignment to minimize total penality for the one machine scheduling problem. Operations Research **30:** 391-399.

14 Graham RL, Lawler EL, Lenstra JK and Rinnoog Kan AHG (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Discrete Mathematics **3:** 287-326.

15 Brucker P (1995). Scheduling Algorithms, Springer: New York.