An FPTAS for scheduling with piecewise linear decreasing processing times to minimize makespan

Min Ji*
College of Computer Science & Information Engineering,
Zhejiang Gongshang University, Hangzhou 310035, P.R. China

T. C. E. Cheng†
Department of Logistics, The Hong Kong Polytechnic University,
Kowloon, Hong Kong

Abstract

We study the problems of scheduling a set of nonpreemptive jobs on a single machine and identical parallel machines, where the processing time of a job is a piecewise linear nonincreasing function of its start time. The objective is to minimize makespan. We first give a fully polynomial-time approximation scheme (FPTAS) for the case with a single machine. We then generalize the result to the case with m identical machines.

Keywords. Machine scheduling; Start time dependent processing times; Makespan

1 Introduction

Machine scheduling problems with jobs having start-time dependent processing times have received increasing attention from the scheduling community over the last decade. For a survey of the research in this area, we refer the reader to Alidaee and Womer [1], and Cheng, Ding and Lin [7].

Recently Cheng et al. [6] considered the following scheduling problem with time-dependent processing times: There are n independent nonpreemptive jobs $\mathcal{J} = \{J_1, J_2, \cdots, J_n\}$, which are simultaneously available, to be scheduled for processing on m parallel machines. Each job can be completely processed by any machine. Each machine can handle at most one job at a time. A schedule is characterized by the sequences of jobs arranged in order of processing on the machines. The processing time of job J_j scheduled on machine k depends on its start time s_j in the following way:

$$p_j^k = \begin{cases} a_j^k, & \text{if } s_j < d, \\ a_j^k - b_j^k(s_j - d), & \text{if } d \le s_j \le D, \\ a_j^k - b_j^k(D - d), & \text{if } s_j > D. \end{cases}$$

On machine k, each job J_j thus has a *normal processing time* a_j^k, a *common initial decreasing date* d, after which the processing time starts to decrease linearly with a *decreasing rate* b_j^k and a *common final decreasing date* D ($D \ge d$), after which it decreases no further. It is assumed that $0 < b_j^k < 1$ and $a_j^k > b_j^k(\min\{D, \sum_{i=1}^n a_i^k - a_j^k\} - d)$ hold for each job J_j and machine k. The first condition

*Email: jimkeen@163.com
†Corresponding author. Email: LGTcheng@polyu.edu.hk

ensures that the decrease of each jobs processing time is less than one unit for every unit delay in its starting moment. The second condition ensures that all the job processing times are positive in a feasible schedule (see also Ho, Leung and Wei [10] for detailed explanations). Given a schedule, the completion time C_j of job J_j, $j = 1, 2, \cdots, n$, is easily determined. The objectives in Cheng et al. [6] are to minimize the makespan $C_{\max} = \max_{j=1,2,\cdots,n} C_j$ and the total completion time $\sum_{j=1}^{n} C_j$. These criteria are related to throughput time and total work-in-process inventories of a production system, respectively. Without loss of generality, it is assumed that all a_j^k, d, and D are integral and all b_j^k are rational so that $b_j^k = v_j^k/L$, where v_j^k are integers and L is a natural number, $j = 1, 2, \cdots, n$, $k = 1, 2, \cdots, m$. For the case with a single machine or identical parallel machines, the index k is dropped in the corresponding notation.

For the makespan problem on a single machine, Cheng et al. [6] proved it was NP-hard in the ordinary sense if $d = 0$ (see Theorem 1 in [6]). And from the parameters in their proof, we can deduce that $b_j \leq a_j/(2D)$ for all j. So Theorem 1 in [6] is in fact valid if $b_j \leq a_j/(2D)$ for all j. This model can provide a better approximation for some real-life situations where job processing times decrease with their start times. For example, in environments where learning effect takes place, the productivity of an operator is at a low level initially, and it gradually increases to a stable level after some time because of physical and safety limitations. And a job's stable processing time is usually not much less than its normal processing time. The assumption $b_j \leq a_j/(2D)$ indicates that the ratio is not less than a half, i.e., $(a_j - b_j D)/a_j \geq 1/2$. In this paper we present a fully polynomial-time approximation scheme (FPTAS) for both the single-machine and identical parallel-machine cases, when $d = 0$ and $b_j \leq a_j/(2D)$ for all j. Using the three-field notation of Graham et al. [8], we denote such scheduling models as $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ and $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$, respectively.

Work on scheduling problems with start time dependent processing times was initiated by Brown and Yechiali [4], and Gupta and Gupta [9]. Since then, scheduling problems with time-dependent processing times have received increasing attention. There are many applications of the model where the job processing time is an increasing function of the job start time. These include the control of queues and communication systems, shops with deteriorating machines, and/or delay of maintenance or cleaning, fire fighting, and hospital emergency wards, scheduling steel rolling mills, etc. [2, 4, 6, 9, 14, 15, 16]. If the job processing time is a decreasing function of the job start time, examples can be found in learning effect, national defense, and aerial threats etc., in which a task consists of destroying an aerial threat and its execution time decreases with time as the threat gets closer. For a list of applications of this scheduling model, the reader is referred to [3, 5, 10, 11, 17, 18].

The presentation of this paper is organized as follows. In Section 2 we propose a fully polynomial-time approximation scheme for the problem $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$, and prove its correctness and establish its time complexity. We then generalize the result to m identical parallel machines in Section 3. Finally, we conclude the paper in Section 4.

2 An FPTAS for the single-machine problem

An algorithm $\mathcal{A}$ is called a $(1+\varepsilon)$-*approximation* algorithm for a minimization problem if it produces a solution that is at most $1+\varepsilon$ times as big as the optimal value, running in time that is polynomial in the input size. A family of approximation algorithms $\{\mathcal{A}_\varepsilon\}$ is a *fully polynomial-time approximation* *scheme* if, for each $\varepsilon > 0$, the algorithm $\mathcal{A}_\varepsilon$ is a $(1+\varepsilon)$-approximation algorithm that is polynomial in the input size and in $1/\varepsilon$. From now on we assume, without loss of generality, that $0 < \varepsilon \leq 1$. If $\varepsilon > 1$, then a 2-approximation algorithm can be taken as a $(1+\varepsilon)$-approximation algorithm.

Ho, Leung and Wei [10] showed that the nonincreasing order of a_j/b_j is optimal for the problem $1|p_j = a_j - b_j s_j|C_{\max}$, which leads to the following property.

Property 1 *There exists an optimal solution for the problems* $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *and* $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *such that on each machine the jobs are sequenced in nonincreasing order of* a_j/b_j *if these jobs' start times are all less than* D.

The following Remark 1 and Property 2 are trivial.

Remark 1 *It is trivial that there exists an optimal solution for the problems* $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *and* $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *such that the sequence of the jobs is immaterial if these jobs' start times are no less than* D.

Property 2 *There is no idle time on each machine in an optimal solution for the problems* $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *and* $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$.

Following Property 1, let the jobs be indexed in nonincreasing order of a_j/b_j so that $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$. We define the 0-1 variables x_j, $j = 1, 2, \cdots, n$, where $x_j = 1$ if the start time of job J_j is less than D, and $x_j = 0$ otherwise. Let X be the set of all the 0-1 vectors $x = (x_1, x_2, \cdots, x_n)$. Let $f_j(x)$ be the total processing time when the jobs $\{J_1, \cdots, J_j\}$ have been processed. Let $g_j(x)$ be $\min\{D$, the total processing time of those jobs whose start times are less than $D\}$ when the jobs $\{J_1, \cdots, J_j\}$ have been processed. We define the following initial and recursive functions on X:

$$\begin{aligned}
f_0(x) &= 0, \\
g_0(x) &= 0, \\
f_j(x) &= f_{j-1}(x) + x_j(a_j - b_j g_{j-1}(x)) + (1 - x_j)(a_j - b_j D), \\
g_j(x) &= \min\{g_{j-1}(x) + x_j(a_j - b_j g_{j-1}(x)), D\}.
\end{aligned}$$

Thus, due to the definition of $f_j(x)$ and Property 2, we conclude that the problem $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ reduces to the following problem:

$$\text{Minimize } f_n(x) \text{ for } x \in X.$$

We first introduce procedure $Partition(A, e, \delta)$ proposed by Kovalyov and Kubiak [12, 13], where $A \subseteq X$, e is a nonnegative integer function on X, and $0 < \delta \leq 1$. This procedure partitions A into disjoint subsets $A_1^e, A_2^e, \cdots, A_{k_e}^e$ such that $|e(x) - e(x')| \leq \delta \min\{e(x), e(x')\}$ for any x, x' from the same subset A_j^e, $j = 1, 2, \cdots, k_e$. The following description provides the details of $Partition(A, e, \delta)$.

Procedure $Partition(A, e, \delta)$

Step 1. Arrange vectors $x \in A$ in order $x^{(1)}, x^{(2)}, \cdots, x^{(|A|)}$ such that $0 \leq e(x^{(1)}) \leq e(x^{(2)}) \leq \cdots \leq e(x^{(|A|)})$.

Step 2. Assign vectors $x^{(1)}, x^{(2)}, \cdots, x^{(i_1)}$ to set A_1^e until i_1 is found such that $e(x^{(i_1)}) \leq (1 + \delta)e(x^{(1)})$ and $e(x^{(i_1+1)}) > (1 + \delta)e(x^{(1)})$. If such i_1 does not exist, then take $A_{k_e}^e = A_1^e = A$, and stop.

Assign vectors $x^{(i_1+1)}, x^{(i_1+2)}, \cdots, x^{(i_2)}$ to set A_2^e until i_2 is found such that $e(x^{(i_2)}) \leq (1+\delta)e(x^{(i_1+1)})$ and $e(x^{(i_2+1)}) > (1+\delta)e(x^{(i_1+1)})$. If such i_2 does not exist, then take $A_{k_e}^e = A_2^e = A - A_1^e$, and stop.

Continue the above construction until $x^{(|A|)}$ is included in $A_{k_e}^e$ for some k_e. ■

Procedure *Partition* requires $O(|A| \log |A|)$ operations to arrange the vectors of A in nondecreasing order of $e(x)$, and $O(|A|)$ operations to provide a partition. The main properties of *Partition* that will be used in the development of our FPTAS $\mathcal{A}_\varepsilon$ were presented in Kovalyov and Kubiak [12, 13] as follows.

Property 3 $|e(x) - e(x')| \leq \delta \min\{e(x), e(x')\}$ *for any* $x, x' \in A_j^e$, $j = 1, 2, \cdots, k_e$.

Property 4 $k_e \leq \log e(x^{(|A|)})/\delta + 2$ *for* $0 < \delta \leq 1$ *and* $1 \leq e(x^{(|A|)})$.

A formal description of the FPTAS $\mathcal{A}_\varepsilon$ for the problem $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ is given below.

Algorithm $\mathcal{A}_\varepsilon$

Step 1. (Initialization) Number the jobs in nonincreasing order of a_j/b_j so that $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$ (Property 1). Set $Y_0 = \{(0, 0, \cdots, 0)\}$ and $j = 1$.

Step 2. (Generation of $Y_1, Y_2, \cdots, Y_n$) For set Y_{j-1}, generate Y_j' by adding 0 and 1 in position j of each vector from Y_{j-1}, i.e., $Y_j' = Y_{j-1} \cup \{x + (0, 0, \cdots, x_j = 1, 0, \cdots, 0) \mid x \in Y_{j-1}\}$. Calculate the following for any $x \in Y_j'$.

$$\begin{aligned} f_j(x) &= f_{j-1}(x) + x_j(a_j - b_j g_{j-1}(x)) + (1 - x_j)(a_j - b_j D), \\ g_j(x) &= \min\{g_{j-1}(x) + x_j(a_j - b_j g_{j-1}(x)), D\}. \end{aligned}$$

If $j = n$, then set $Y_n = Y_n'$, and go to Step 3.
If $j < n$, then set $\delta = \varepsilon/(2(n+1))$, and perform the following computations.
Call $Partition(Y_j', f_j, \delta)$ to partition set Y_j' into disjoint subsets $Y_1^f, Y_2^f, \cdots, Y_{k_f}^f$.
Call $Partition(Y_j', g_j, \delta)$ to partition set Y_j' into disjoint subsets $Y_1^g, Y_2^g, \cdots, Y_{k_g}^g$.
Divide set Y_j' into disjoint subsets $Y_{ab} = Y_a^f \cap Y_b^g$, $a = 1, 2, \cdots, k_f$, $b = 1, 2, \cdots, k_g$. For each nonempty subset Y_{ab}, choose a vector $x^{(ab)}$ such that

$$f_j(x^{(ab)}) = \min\{f_j(x) \mid x \in Y_{ab}\}.$$

Set $Y_j := \{x^{(ab)} \mid a = 1, 2, \cdots, k_f, b = 1, 2, \cdots, k_g, \text{ and } Y_a^f \cap Y_b^g \neq \emptyset\}$, and $j = j + 1$.
Repeat Step 2.

Step 3. (Solution) Select vector $x^0 \in Y_n$ such that $f_n(x^0) = \min\{f_n(x) \mid x \in Y_n\}$. ■

Let $x^* = (x_1^*, x_2^*, \cdots, x_n^*)$ be an optimal solution for the problem $1/p_j = a_j - b_j \min\{s_j, D\}/C_{\max}$ and $L = \log(\max\{n, 1/\varepsilon, a_{\max}\})$, where $a_{\max} = \max_{j=1,2,\cdots,n}\{a_j\}$. We show the main result of this section in the following.

Theorem 1 *When* $b_j \leq \frac{a_j}{2D}$ *for all* j, *Algorithm* $\mathcal{A}_\varepsilon$ *finds* $x^0 \in X$ *for the problem* $1|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ *such that* $f_n(x^0) \leq (1+\varepsilon) f_n(x^*)$ *in* $O(n^3 L^3/\varepsilon^2)$.

Proof. Suppose that $(x_1^*, \cdots, x_j^*, 0, \cdots, 0) \in Y_{ab} \subseteq Y_j'$ for some j and a, b. By the definition of $\mathcal{A}_\varepsilon$, such j always exists, for instance $j = 1$. Algorithm $\mathcal{A}_\varepsilon$ may not choose $(x_1^*, \cdots, x_j^*, 0, \cdots, 0)$ for further construction; however, for a vector $x^{(ab)}$ chosen instead of it, we have

$$|f_j(x^*) - f_j(x^{(ab)})| \le \delta f_j(x^*),$$

and

$$|g_j(x^*) - g_j(x^{(ab)})| \le \delta g_j(x^*),$$

due to Property 3. We consider vector $(x_1^*, \cdots, x_j^*, x_{j+1}^*, 0, \cdots, 0)$ and $\tilde{x}^{(ab)} = (x_1^{(ab)}, \cdots, x_j^{(ab)}, x_{j+1}^*, 0, \cdots, 0)$. We have

$$\begin{aligned}
& |f_{j+1}(x^*) - f_{j+1}(\tilde{x}^{(ab)})| \\
= & \Big| \Big[f_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*)) + (1 - x_{j+1}^*)(a_{j+1} - b_{j+1} D) \Big] \\
& - \Big[f_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)})) + (1 - x_{j+1}^*)(a_{j+1} - b_{j+1} D) \Big] \Big| \\
\le & |f_j(x^*) - f_j(x^{(ab)})| + x_{j+1}^* b_{j+1} |g_j(x^*) - g_j(x^{(ab)})| \\
\le & \delta f_j(x^*) + \delta x_{j+1}^* b_{j+1} g_j(x^*).
\end{aligned}$$

Since $g_j(x^*) \le D$ and $b_k \le a_k/(2D)$ for all k, we obtain $b_{j+1} g_j(x^*) \le a_{j+1} - b_{j+1} g_j(x^*)$. Combining this with the above formula, and setting $\delta_1 = \delta$, we have

$$\begin{aligned}
& |f_{j+1}(x^*) - f_{j+1}(\tilde{x}^{(ab)})| \\
\le & \delta(f_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*))) \le \delta_1 f_{j+1}(x^*).
\end{aligned} \tag{1}$$

Consequently,

$$f_{j+1}(\tilde{x}^{(ab)}) \le (1 + \delta_1) f_{j+1}(x^*).$$

We now show that

$$|g_{j+1}(x^*) - g_{j+1}(\tilde{x}^{(ab)})| \le \delta_1 g_{j+1}(x^*). \tag{2}$$

It is easy to verify (2) if $D \le \min\{g_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*)), g_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)}))\}$. If $D \ge \max\{g_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*)), g_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)}))\}$, then

$$\begin{aligned}
& |g_{j+1}(x^*) - g_{j+1}(\tilde{x}^{(ab)})| \\
= & |[g_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*))] - [g_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)}))]| \\
\le & x_{j+1}^*(1 - b_{j+1}) |g_j(x^*) - g_j(x^{(ab)})| \le \delta x_{j+1}^*(1 - b_{j+1}) g_j(x^*) \le \delta_1 g_{j+1}(x^*).
\end{aligned} \tag{3}$$

So, (2) is satisfied. If $(g_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*)) - D) * (g_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)})) - D) \le 0$, then we have $|g_{j+1}(x^*) - g_{j+1}(\tilde{x}^{(ab)})| \le |[g_j(x^*) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^*))] - [g_j(x^{(ab)}) + x_{j+1}^*(a_{j+1} - b_{j+1} g_j(x^{(ab)}))]|$. Thus, similar to (3), (2) is satisfied, too. Therefore (2) is always true, and (2) implies

$$g_{j+1}(\tilde{x}^{(ab)}) \le (1 + \delta_1) g_{j+1}(x^*).$$

Assume that $\tilde{x}^{(ab)} \in Y_{de} \subseteq Y_{j+1}'$ and that Algorithm $\mathcal{A}_\varepsilon$ chooses $x^{(de)} \in Y_{de}$ instead of $\tilde{x}^{(ab)}$ in the $(j+1)$st iteration. We have

$$|f_{j+1}(\tilde{x}^{(ab)}) - f_{j+1}(x^{(de)})| \le \delta f_{j+1}(\tilde{x}^{(ab)}) \le \delta(1 + \delta_1) f_{j+1}(x^*), \tag{4}$$

and

$$|g_{j+1}(\tilde{x}^{(ab)}) - g_{j+1}(x^{(de)})| \leq \delta g_{j+1}(\tilde{x}^{(ab)}) \leq \delta(1+\delta_1)g_{j+1}(x^*).$$

From (1) and (4), we obtain

$$\begin{aligned} & |f_{j+1}(x^*) - f_{j+1}(x^{(de)})| \\ \leq\ & |f_{j+1}(x^*) - f_{j+1}(\tilde{x}^{(ab)})| + |f_{j+1}(\tilde{x}^{(ab)}) - f_{j+1}(x^{(de)})| \\ \leq\ & (\delta_1 + \delta(1+\delta_1))f_{j+1}(x^*) = (\delta + \delta_1(1+\delta))f_{j+1}(x^*). \end{aligned} \tag{5}$$

Similarly, we have

$$|g_{j+1}(x^*) - g_{j+1}(x^{(de)})| \leq (\delta + \delta_1(1+\delta))g_{j+1}(x^*).$$

Set $\delta_l = \delta + \delta_{l-1}(1+\delta)$, $l = 2, 3, \cdots, n-j+1$. From (5), we obtain

$$|f_{j+1}(x^*) - f_{j+1}(x^{(de)})| \leq \delta_2 f_{j+1}(x^*).$$

Repeating the above argument for $j+2, \cdots, n$, we show that there exists $x' \in Y_n$ such that

$$|f_n(x^*) - f_n(x')| \leq \delta_{n-j+1} f_n(x^*).$$

Since

$$\begin{aligned} \delta_{n-j+1} &\leq \delta \sum_{j=0}^{n} (1+\delta)^j \\ &= (1+\delta)^{n+1} - 1 \\ &= \sum_{j=1}^{n+1} \frac{(n+1)n\cdots(n-j+2)}{j!(n+1)^j} (\frac{\varepsilon}{2})^j \\ &\leq \sum_{j=1}^{n+1} \frac{1}{j!} (\frac{\varepsilon}{2})^j \leq \sum_{j=1}^{n+1} (\frac{\varepsilon}{2})^j \leq \varepsilon \sum_{j=1}^{n+1} (\frac{1}{2})^j \leq \varepsilon. \end{aligned}$$

Therefore, we have

$$|f_n(x^*) - f_n(x')| \leq \varepsilon f_n(x^*).$$

Then in Step 3, vector x^0 will be chosen such that

$$f_n(x^0) \leq f_n(x') \leq (1+\varepsilon) f_n(x^*).$$

The time complexity of Algorithm $\mathcal{A}_\varepsilon$ can be established by noting that the most time-consuming operation is iteration j in Step 2, i.e., a call of procedure *Partition*, which requires $O(|Y_j'| \log |Y_j'|)$ time to complete. To estimate $|Y_j'|$, recall that $|Y_{j+1}'| \leq 2|Y_j| \leq 2k_f k_g$. By Property 4, we have $k_f \leq 2(n+1)\log(na_{\max})/\varepsilon + 2 \leq 2(n+1)L/\varepsilon + 2$, and the same for k_g. Thus, $|Y_j'| = O(n^2L^2/\varepsilon^2)$, and $|Y_j'| \log |Y_j'| = O(n^2L^3/\varepsilon^2)$. Therefore, the time complexity of Algorithm $\mathcal{A}_\varepsilon$ is $O(n^3L^3/\varepsilon^2)$. ∎

3 An FPTAS for the problem with m identical parallel machines

In this section we generalize the result to the case with m identical parallel machines. We introduce variables x_j, $j = 1, 2, 3, 4, \cdots, 2m-1, 2m$, where $x_j = 2k-1$ if job J_j is processed on machine k, $k \in \{1, 2, \cdots, m\}$, and its start time is less than D, $x_j = 2k$ if job J_j is processed on machine

k, $k \in \{1, 2, \cdots, m\}$, and its start time is no less than D. Let X be the set of all the vectors $x = (x_1, x_2, \cdots, x_n)$ with $x_j = k$, $j = 1, 2, \cdots, n$, $k = 1, 2, \cdots, 2m$. We define the following initial and recursive functions on X:

$$
\begin{aligned}
f_0^i(x) &= 0, \ i = 1, 2, \cdots, m, \\
g_0^i(x) &= 0, \ i = 1, 2, \cdots, m, \\
f_j^k(x) &= f_{j-1}^k(x) + (a_j - b_j g_{j-1}^k(x)), \ \text{if } x_j = 2k - 1, \\
f_j^k(x) &= f_{j-1}^k(x) + (a_j - b_j D), \ \text{if } x_j = 2k, \\
f_j^i(x) &= f_{j-1}^i(x), \ \text{if } x_j = 2k - 1 \text{ or } x_j = 2k, i \neq k, \\
g_j^k(x) &= \min\{g_{j-1}^k(x) + (a_j - b_j g_{j-1}^k(x)), D\}, \ \text{if } x_j = 2k - 1, \\
g_j^i(x) &= g_{j-1}^i(x), \ \text{if } x_j = 2k - 1 \text{ or } x_j = 2k, i \neq k.
\end{aligned}
$$

Thus, the problem $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ reduces to the following problem:

$$\text{Minimize } Q(x) \text{ for } x \in X, \text{ where } Q(x) = \max_{i=1,2,\cdots,m} f_n^i(x).$$

A formal description of the FPTAS $\mathcal{A}_\varepsilon^m$ for the problem $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ is given below.

Algorithm $\mathcal{A}_\varepsilon^m$

Step 1. (Initialization) Number the jobs in nonincreasing order of a_j/b_j so that $a_1/b_1 \geq a_2/b_2 \geq \cdots \geq a_n/b_n$ (Property 1). Set $Y_0 = \{(0, 0, \cdots, 0)\}$ and $j = 1$.

Step 2. (Generation of $Y_1, Y_2, \cdots, Y_n$) For set Y_{j-1}, generate Y_j' by adding k, $k = 1, 2, \cdots, 2m$, in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, assuming $x_j = k$.

$$
\begin{aligned}
f_j^k(x) &= f_{j-1}^k(x) + (a_j - b_j g_{j-1}^k(x)), \ \text{if } x_j = 2k - 1, \\
f_j^k(x) &= f_{j-1}^k(x) + (a_j - b_j D), \ \text{if } x_j = 2k, \\
f_j^i(x) &= f_{j-1}^i(x), \ \text{if } x_j = 2k - 1 \text{ or } x_j = 2k, i \neq k, \\
g_j^k(x) &= \min\{g_{j-1}^k(x) + (a_j - b_j g_{j-1} k(x)), D\}, \ \text{if } x_j = 2k - 1, \\
g_j^i(x) &= g_{j-1}^i(x), \ \text{if } x_j = 2k - 1 \text{ or } x_j = 2k, i \neq k.
\end{aligned}
$$

If $j = n$, then set $Y_n = Y_n'$, and go to Step 3.

If $j < n$, then set $\delta = \varepsilon/(2(n + 1))$, and perform the following computations.

Call $Partition(Y_j', f_j^i, \delta)$ $(i = 1, 2, \cdots, m)$ to partition set Y_j' into disjoint subsets $Y_1^{f^i}, Y_2^{f^i}, \cdots, Y_{k_{f^i}}^{f^i}$.

Call $Partition(Y_j', g_j^i, \delta)$ $(i = 1, 2, \cdots, m)$ to partition set Y_j' into disjoint subsets $Y_1^{g^i}, Y_2^{g^i}, \cdots, Y_{k_g^i}^{g^i}$.

Divide set Y_j' into disjoint subsets $Y_{a_1 \cdots a_m b_1 \cdots, b_m} = Y_{a_1}^{f^1} \cap \cdots \cap Y_{a_m}^{f^m} \cap Y_{b_1}^{g^1} \cap \cdots \cap Y_{g_m}^{g^m}$, $a_1 = 1, 2, \cdots, k_{f^1}$; $\cdots$; $a_m = 1, 2, \cdots, k_{f^m}$; $b_1 = 1, 2, \cdots, k_{g^1}$; $\cdots$; $b_m = 1, 2, \cdots, k_{g^m}$. For each nonempty subset $Y_{a_1 \cdots a_m b_1 \cdots b_m}$, choose a vector $x^{(a_1 \cdots a_m b_1 \cdots b_m)}$ such that

$$\max_{i=1,2,\cdots,m} f_j^i(x^{(a_1 \cdots a_m b_1 \cdots, b_m)}) = \min\{\max_{i=1,2,\cdots,m} f_j^i(x) \mid x \in Y_{a_1 \cdots a_m b_1 \cdots, b_m}\}.$$

Set $Y_j := \{x^{(a_1\cdots a_m b_1\cdots b_m)} \mid a_1 = 1,2,\cdots,k_{f^1};\cdots;a_m = 1,2,\cdots,k_{f^m}; b_1 = 1,2,\cdots,k_{g^1};\cdots;b_m = 1,2,\cdots,k_{g^m};$ and $Y_{a_1}^{f^1} \cap \cdots \cap Y_{a_m}^{f^m} \cap Y_{b_1}^{g^1} \cap \cdots \cap Y_{b_m}^{g^m} \neq \emptyset\}$, and $j = j+1$.

Repeat Step 2.

Step 3. (Solution) Select vector $x^0 \in Y_n$ such that $Q(x^0) = \min\{Q(x) \mid x \in Y_n\} = \min\{\max_{i=1,2,\cdots,m} f_n^i(x) \mid x \in Y_n\}$. ■

Let $x^* = (x_1^*, x_2^*, \cdots, x_n^*)$ be an optimal solution for the problem $P_m/p_j = a_j - b_j \min\{s_j, D\}/C_{\max}$. Let $L = \log(\max\{n, 1/\varepsilon, a_{\max}\})$, where $a_{\max} = \max_{j=1,2,\cdots,n}\{a_j\}$. We have the following result.

Theorem 2 *When $b_j \leq \frac{a_j}{2D}$ for all j, Algorithm $\mathcal{A}_\varepsilon^m$ finds $x^0 \in X$ for the problem $P_m|p_j = a_j - b_j \min\{s_j, D\}|C_{\max}$ such that $Q(x^0) \leq (1+\varepsilon)Q(x^*)$ in $O(n^{2m+1}L^{2m+1}/\varepsilon^{2m})$.*

Proof. Using an argument similar to that used to establish Theorem 1, we can show that there exists $x' \in Y_n$ such that

$$|f_n^i(x^*) - f_n^i(x')| \leq \varepsilon f_n^i(x*), i = 1,2,\cdots,n.$$

It implies

$$|\max_{i=1,2,\cdots,m} f_n^i(x') - \max_{i=1,2,\cdots,m} f_n^i(x^*)| \leq \varepsilon \max_{i=1,2,\cdots,m} f_n^i(x^*).$$

Then, in Step 3, vector x^0 will be chosen such that

$$\begin{aligned} & |\max_{i=1,2,\cdots,m} f_n^i(x^0) - \max_{i=1,2,\cdots,m} f_n^i(x^*)| \\ \leq & |\max_{i=1,2,\cdots,m} f_n^i(x') - \max_{i=1,2,\cdots,m} f_n^i(x^*)| \\ \leq & \varepsilon \max_{i=1,2,\cdots,m} f_n^i(x^*). \end{aligned}$$

Therefore we have $Q(x^0) \leq (1+\varepsilon)Q(x^*)$

Similar to Theorem 1, we can show that the computational complexity of Algorithm $\mathcal{A}_\varepsilon^m$ is $O(n^{2m+1}L^{2m+1}/\varepsilon^{2m})$. ■

4 Conclusions

This paper studied the scheduling problem in which the processing time of a job is a piecewise linear nonincreasing function of its start time to minimize makespan. We first gave a fully polynomial-time approximation scheme for the single-machine case, then generalized the result to the case with m machines, where m is fixed. Future research may focus on other objectives.

Acknowledgment

Cheng was supported in part by The Hong Kong Polytechnic University under a grant from the *Area of Strategic Development in China Business Services.*

References

[1] B. Alidaee, N.K. Womer, Scheduling with time dependent processing times: Review and extensions, *Journal of Operational Research Society*, 50 (1999) 711-720.

[2] A. Bachman, A. Janiak, M.Y. Kovalyov, Minimizing the total weighted completion time of deteriorating jobs, *Information Processing Letters*, 81 (2002) 81-84.

[3] A. Bachman, T.C.E. Cheng, A. Janiak, C.T. Ng, Scheduling start time dependent jobs to minimize the total weighted completion time, *Journal of the Operational Research Society*, 53 (2002) 688-693.

[4] S. Brown, U. Yechiali, Scheduling deteriorating jobs on a single process, *Operations Research*, 38 (1990) 495-498.

[5] Z.L. Chen, A note on single-processor scheduling with time dependent execution times, *Operations Research Letters*, 17 (1995) 127-129.

[6] T.C.E. Cheng, Q. Ding, M.Y. Kovalyov, A. Bachman, A. Janiak, Scheduling jobs with piecewise linear decreasing processing times, *Naval Research Logistics*, 50 (2003) 531-554.

[7] T.C.E. Cheng, Q. Ding, B.M.T. Lin, A concise survey of scheduling with time-dependent processing times, *European Journal of Operational Research*, 152 (2004) 1-13.

[8] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, *Annals of Discrete Mathematics*, 5 (1979) 287-326.

[9] J.N.D. Gupta, S.K. Gupta, Single facility scheduling with nonlinear processing times, *Computers and Industrial Engineering*, 14 (1988) 387-393.

[10] K.I.J. Ho, J.Y.T. Leung, W.D. Wei, Complexity of scheduling tasks with time-dependent execution times, *Information Processing Letters*, 48 (1993) 315-320.

[11] M. Ji, Y. He, T.C.E. Cheng, A simple linear time algorithm for scheduling with step-improving processing times, *Computers and Operations Research*, in press.

[12] M.Y. Kovalyov, W. Kubiak, A fully polynomial approximation scheme for minimizing makespan of deteriorating jobs, *Journal of Heuristics*, 3 (1998) 287-297.

[13] M.Y. Kovalyov, W. Kubiak, A fully polynomial approximation scheme for the weighted earliness-tardiness problem, *Operations Research*, 47 (1999) 757-761.

[14] W. Kubiak, S.L. van de Velde, Scheduling deteriorating jobs to minimize makespan, *Naval Research Logistics*, 45 (1998) 511-523.

[15] A.S. Kunnathur, S.K. Gupta, Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem, *European Journal of Operational Research*, 47 (1990) 56-64.

[16] G. Mosheiov, V-shaped policies for scheduling deteriorating jobs, *Operations Research*, 39 (1991) 979-991.

[17] C.T. Ng, T.C.E. Cheng, A. Bachman, A. Janiak, Three scheduling problems with deteriorating jobs to minimize the total completion time, *Information Processing Letters*, 81 (2002) 327-333.

[18] G.J. Woeginger, Scheduling with time-dependent execution times, *Information Processing Letters*, 54 (1995) 155-156.