Parallel-machine scheduling with simple linear deterioration to minimize total completion time

Min Ji[1]

College of Computer Science & Information Engineering,
Zhejiang Gongshang University, Hangzhou 310018, P.R. China

T. C. E. Cheng[2]

Department of Logistics,
The Hong Kong Polytechnic University,
Kowloon, Hong Kong

[1]Email: jimkeen@163.com
[2]Corresponding author. Email: LGTcheng@polyu.edu.hk

Abstract

We consider the parallel-machine scheduling problem in which the processing time of a job is a simple linear increasing function of its starting time. The objective is to minimize the total completion time. We give a fully polynomial-time approximation scheme (FPTAS) for the case with m identical machines, where m is fixed. This study solves an open problem that has been posed in the literature for ten years.

Keywords. Parallel-machine scheduling; Deteriorating jobs; FPTAS

1 Introduction

For most scheduling problems it is assumed that the job processing times are fixed parameters [14]. However, such restrictive assumptions represent an oversimplified view of reality. Job processing times are not necessarily fixed because jobs may deteriorate while waiting to be processed. Examples can be found in financial management, steel production, resource allocation and national defence, etc., where any delay in processing a job may result in deterioration in accomplishing the job. For a list of applications, the reader is referred to [2, 10, 11, 13]. Such problems are generally known as the deteriorating job scheduling problem.

Work on the deteriorating job scheduling problem was initiated by Brown and Yechiali [2], and Gupta and Gupta [6]. They focused on the single-machine makespan problem under the linear deteriorating condition. Since then, scheduling problems with time-dependent processing times have received increasing attention. An extensive survey of different models and problems was provided by Alidaee and Womer [1]. Cheng, Ding and Lin [5] recently presented an updated survey of the results on scheduling problems with time-dependent processing times.

The problem under consideration can be formally described as follows: There are n independent jobs $\mathcal{J} = \{J_1, J_2, \cdots, J_n\}$, which are simultaneously available at time $t_0 > 0$, to be processed non-preemptively on m identical parallel machines. We assume, as in Mosheiov [12, 13], and Chen [3, 4], that the actual processing time of job J_j is $p_j = \alpha_j s_j$, where s_j and α_j are the starting time and the growth (or deterioration) rate of J_j, respectively. The assumption "$t_0 > 0$" is made here to avoid the trivial case of $t_0 = 0$ (when $t_0 = 0$, the completion time of each job will be 0). Let $C_{[i]}$ denote the completion time of the ith job in a schedule. It is easy to see that $C_{[i+1]} = s_{[i+1]} + p_{[i+1]} = s_{[i+1]}(1 + \alpha_{[i+1]}) = C_{[i]}(1 + \alpha_{[i+1]})$. Thus, by induction, $C_{[i]} = t_0 \prod_{j=1}^{i}(1 + \alpha_{[j]})$ for every $i \geq 1$ in a schedule. Since t_0 is a common factor in all the completion times, it will not affect the optimal schedule under any performance criterion. Without loss of generality, we assume that t_0 and all α_j for every j are integral. Our goal is to minimize the total completion time, i.e., $\sum_{j=1}^{n} C_j$. Using the notation of Chen [3], we denote the problem as the PTCT (i.e., parallel-machine total completion time) problem.

The above defined problem may date back to Mosheiov [12], who first considered single-machine scheduling under the simple linear deteriorating assumption. The most commonly used performance measures were considered, such as makespan, total completion time, total weighted completion time, total weighted waiting time, total tardiness, number of tardy jobs, maximum lateness and maximum tardiness. He showed that all these models are polynomially solvable. For the multiple machine case of the simple linear deteriorating problem, Mosheiov [13] showed its NP-hardness even for the two-machine case, and presented an asymptotically optimal heuristic for the parallel-machine makespan problem. Chen [3, 4] proved that the PTCT problem is NP-hard even with a fixed number of machines. He also proposed an approximation algorithm with a parameter dependent worst-case ratio for the two-machine case. For a fixed number of machines, he pointed out that whether there exists a polynomial time heuristic with a constant worst-case ratio remains open. More recently, Wu and Lee [15], and Ji et al. [7] extended the single-machine problem to the situation where the machine has an availability constraint. Wu and Lee [15] studied the resumable case of the makespan problem. They showed that the linear deteriorating model can be solved as a 0-1 integer programming problem. Ji et al. [7] considered the non-resumable case to minimize the makespan and the total completion time. They established their computational complexity, proposed approximation algorithms with tight bounds and presented computational results.

In this paper we present a fully polynomial-time approximation scheme (FPTAS) for the PTCT problem with a fixed number of machines, which greatly improves on the bound in Chen [3], noting that the parameter dependent worst-case ratio given in Chen [3] may be infinity. Our FPTAS closely follows the FPTAS developed in [8, 9]. We have thus solved the open problem that has been posed in Chen [3] for ten years. The presentation of this paper is organized as follows. In Section 2 we propose an FPTAS for the m-machine case of the PTCT problem, where m is fixed, and prove its correctness and establish its time complexity. Finally, we conclude the paper in Section 3.

2 An FPTAS

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* (FPTAS) 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.

Mosheiov [12] showed that the smallest growth rate (SGR) order is optimal for the single-machine case of the problem, which leads to the following property.

Property 1 *On each machine of an optimal solution for the PTCT problem, all the jobs are sequenced in the SGR order, i.e., in nondecreasing order of* α_j.

Let the jobs be indexed according to the SGR order so that $\alpha_1 \leq \alpha_2 \leq \cdots \leq \alpha_n$. We introduce variables x_j, $j = 1, 2, \cdots, n$, where $x_j = k$ if job J_j is processed on machine k, $k \in \{1, 2, \cdots, m\}$. Let X be the set of all the vectors $x = (x_1, x_2, \cdots, x_n)$ with $x_j = k$, $j = 1, 2, \cdots, n$, $k = 1, 2, \cdots, m$. We define the following initial and recursive functions on X:

$$\begin{aligned}
f_0^i(x) &= t_0, \ i = 1, 2, \cdots, m, \\
h_0(x) &= 0, \\
f_j^k(x) &= f_{j-1}^k(x) + \alpha_j f_{j-1}^k(x), \ \text{for } x_j = k, \\
f_j^i(x) &= f_{j-1}^i(x), \ \text{for } x_j = k, i \neq k, \\
h_j(x) &= h_{j-1}(x) + \sum_{i=1}^{m} f_j^i(x).
\end{aligned}$$

Thus, the PTCT problem with m machines reduces to the following problem:

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

We first introduce procedure $Partition(A, e, \delta)$ proposed by Kovalyov and Kubiak [8, 9], 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 the order $x^{(1)}, x^{(2)}, \cdots, x^{(|A|)}$ such that $0 \le e(x^{(1)}) \le e(x^{(2)}) \le \cdots \le e(x^{(|A|)})$.

Step 2. Assign vectors $x^{(1)}, x^{(2)}, \cdots, x^{(i_1)}$ to set A_1^e until i_1 is found such that $e(x^{(i_1)}) \le (1+\delta)e(x^{(1)})$ 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 $e(x^{(i_2)}) \le (1+\delta)e(x^{(i_1+1)})$ 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 [8, 9] as follows.

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

Property 3 $k_e \le \log e(x^{(|A|)})/\delta + 2$ *for* $0 < \delta \le 1$ *and* $1 \le e(x^{(|A|)})$.

A formal description of the FPTAS $\mathcal{A}_\varepsilon^m$ for the PTCT problem with m machines is given below.

Algorithm $\mathcal{A}_\varepsilon$

Step 1. (Initialization) Number the jobs in the SGR order so that $\alpha_1 \le \alpha_2 \le \cdots \le \alpha_n$ (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, $k = 1, 2, \cdots, m$, in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, assuming $x_j = k$.

$$\begin{aligned} f_j^k(x) &= f_{j-1}^k(x) + \alpha_j f_{j-1}^k(x), \\ f_j^i(x) &= f_{j-1}^i(x), \text{ for } i \ne k, \\ h_j(x) &= h_{j-1}(x) + \sum_{k=1}^{m} f_j^k(x). \end{aligned} \tag{1}$$

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

Call $Partition(Y'_j, h_j, \delta)$ to partition set Y'_j into disjoint subsets $Y_1^h, Y_2^h, \cdots, Y_{k_h}^h$.

Divide set Y'_j into disjoint subsets $Y_{a_1 \cdots a_m b} = Y_{a_1}^{f^1} \cap \cdots \cap Y_{a_m}^{f^m} \cap Y_b^h$, $a_1 = 1, 2, \cdots, k_{f^1}$; $\cdots$; $a_m = 1, 2, \cdots, k_{f^m}$; $b = 1, 2, \cdots, k_h$. For each nonempty subset $Y_{a_1 \cdots a_m b}$, choose a vector $x^{(a_1 \cdots a_m b)}$ such that

$$h_j(x^{(a_1 \cdots a_m b)}) = \min\{h_j(x) \mid x \in Y_{a_1 \cdots a_m b}\}.$$

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

Repeat Step 2.

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

Let $x^* = (x_1^*, x_2^*, \cdots, x_n^*)$ be an optimal solution for the PTCT problem with a fixed number of machines. Let $L = \log(\max\{n, 1/\varepsilon, 1 + \alpha_{\max}, t_0\})$, where $\alpha_{\max} = \max_{j=1}^n\{\alpha_j\}$. We show the main result of this section in the following.

Theorem 1 *Algorithm $\mathcal{A}_\varepsilon$ finds $x^0 \in X$ for the PTCT problem with m machines such that $h_n(x^0) \leq (1+\varepsilon)h_n(x^*)$ in $O(n^{2m+3}L^{m+2}/\varepsilon^{m+1})$.*

Proof. Suppose that $(x_1^*, \cdots, x_j^*, 0, \cdots, 0) \in Y_{a_1 \cdots a_m b} \subseteq Y'_j$ for some j and $a_1, \cdots a_m, b$. 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 $x^{(a_1 \cdots a_m b)}$ chosen instead of it, we have

$$|f^i_j(x^*) - f^i_j(x^{(a_1 \cdots a_m b)})| \leq \delta f^i_j(x^*), i = 1, \cdots, m,$$

and

$$|h_j(x^*) - h_j(x^{(a_1 \cdots a_m b)})| \leq \delta h_j(x^*),$$

due to Property 2. Set $\delta_1 = \delta$. We consider vector $(x_1^*, \cdots, x_j^*, x_{j+1}^*, 0, \cdots, 0)$ and $\tilde{x}^{(a_1 \cdots a_m b)} = (x_1^{(a_1 \cdots a_m b)}, \cdots, x_j^{(a_1 \cdots a_m b)}, x_{j+1}^*, 0, \cdots, 0)$. WLOG., we assume $x_{j+1}^* = k$. It follows that

$$\begin{aligned}
& |f_{j+1}^k(x^*) - f_{j+1}^k(\tilde{x}^{(a_1 \cdots a_m b)})| \\
= \; & |(f_j^k(x^*) + \alpha_{j+1} f_j^k(x^*)) - (f_j^k(x^{(a_1 \cdots a_m b)}) + \alpha_{j+1} f_j^k(x^{(a_1 \cdots a_m b)}))| \\
= \; & |(1 + \alpha_{j+1})(f_j^k(x^*) - f_j^k(x^{(a_1 \cdots a_m b)}))| \\
\leq \; & (1 + \alpha_{j+1}) \delta f_j^k(x^*) \\
= \; & \delta_1 f_{j+1}^k(x^*),
\end{aligned}$$

Consequently,

$$f_{j+1}^k(\tilde{x}^{(a_1 \cdots a_m b)}) \leq (1 + \delta_1) f_{j+1}^k(x^*),$$

Similarly, for $i \neq k$, we have

$$f_{j+1}^i(\tilde{x}^{(a_1 \cdots a_m b)}) \leq (1 + \delta_1) f_{j+1}^i(x^*).$$

Therefore, we obtain

$$\begin{aligned}
& |h_{j+1}(x^*) - h_{j+1}(\tilde{x}^{(a_1 \cdots a_m b)})| \\
= \; & |(h_j(x^*) + \sum_{i=1}^m f_{j+1}^i(x^*)) - (h_j(x^{(a_1 \cdots a_m b)}) + \sum_{i=1}^m f_{j+1}^i(\tilde{x}^{(a_1 \cdots a_m b)}))| \\
\leq \; & |h_j(x^*) - h_j(x^{(a_1 \cdots a_m b)})| + \sum_{i=1}^m |f_{j+1}^i(x^*) - f_{j+1}^i(\tilde{x}^{(a_1 \cdots a_m b)})| \\
\leq \; & \delta_1 (h_j(x^*) + \sum_{i=1}^m f_{j+1}^i(x^*)) \\
= \; & \delta_1 h_{j+1}(x^*). \qquad (2)
\end{aligned}$$

Consequently,

$$h_{j+1}(\tilde{x}^{(a_1 \cdots a_m b)}) \leq (1 + \delta_1) h_{j+1}(x^*).$$

Assume that $\tilde{x}^{(a_1 \cdots a_m b)} \in Y_{c_1 \cdots c_m d} \subseteq Y'_{j+1}$ and Algorithm $\mathcal{A}_\varepsilon$ chooses $x^{(c_1 \cdots c_m d)} \in Y_{c_1 \cdots c_m d}$ instead of $\tilde{x}^{(a_1 \cdots a_m b)}$ in the $(j+1)$st iteration. We have

$$|f_{j+1}^i(\tilde{x}^{(a_1 \cdots a_m b)}) - f_{j+1}^i(x^{(c_1 \cdots c_m d)})| \leq \delta f_{j+1}^i(\tilde{x}^{(a_1 \cdots a_m b)}) \leq \delta(1 + \delta_1) f_{j+1}^i(x^*), i = 1, \cdots, m,$$

and

$$|h_{j+1}(\tilde{x}^{(a_1\cdots a_m b)}) - h_{j+1}(x^{(c_1\cdots c_m d)})| \leq \delta h_{j+1}(\tilde{x}^{(a_1\cdots a_m b)}) \leq \delta(1+\delta_1)h_{j+1}(x^*). \tag{3}$$

From (2) and (3), we obtain

$$\begin{aligned} & |h_{j+1}(x^*) - h_{j+1}(x^{(c_1\cdots c_m d)})| \\ \leq\ & |h_{j+1}(x^*) - h_{j+1}(\tilde{x}^{(a_1\cdots a_m b)})| + |h_{j+1}(\tilde{x}^{(a_1\cdots a_m b)}) - h_{j+1}(x^{(c_1\cdots c_m d)})| \\ \leq\ & (\delta_1 + \delta(1+\delta_1))h_{j+1}(x^*) \\ =\ & (\delta + \delta_1(1+\delta))h_{j+1}(x^*). \end{aligned} \tag{4}$$

Set $\delta_l = \delta + \delta_{l-1}(1+\delta)$, $l = 2, 3, \cdots, n-j+1$. From (4), we obtain

$$|h_{j+1}(x^*) - h_{j+1}(x^{(c_1\cdots c_m d)})| \leq \delta_2 h_{j+1}(x^*).$$

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

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

Since

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

Therefore, we have

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

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

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

The time complexity of Algorithm $\mathcal{A}_\varepsilon$ can be established by noting that the most time-consuming operation of iteration j of Step 2 is a call of procedure *Partition*, which requires $O(|Y'_j|\log|Y'_j|)$ time to complete. To estimate $|Y'_j|$, recall that $|Y'_{j+1}| \leq 2|Y_j| \leq 2k_f^1 k_f^2 \cdots k_f^m k_h$. By Property 3, we have $k_f^i \leq 2(n+1)\log(t_0(1+\alpha_{\max})^n)/\varepsilon + 2 \leq 2(n+1)^2 L/\varepsilon + 2$, $i = 1, 2, \cdots, m$, and the same for k_h. Thus, $|Y'_j| = O(n^{2(m+1)}L^{m+1}/\varepsilon^{m+1})$, and $|Y'_j|\log|Y'_j| = O(n^{2(m+1)}L^{m+2}/\varepsilon^{m+1})$. Therefore, the time complexity of Algorithm $\mathcal{A}_\varepsilon$ is $O(n^{2(m+1)+1}L^{m+2}/\varepsilon^{m+1})$. ■

3 Conclusions

This paper studied the parallel-machine scheduling problem in which the processing time of a job is a simple linear function of its starting time to minimize the total completion time. We gave a fully polynomial-time approximation scheme for the case with m machines, where m is fixed. Future research may focus on scheduling problems with jobs of more general deterioration types. It will also be interesting to investigate problems with other scheduling 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*, 1999, 50: 711-720.

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

[3] Z. L. Chen, Parallel machine scheduling with time dependent processing times, *Discrete Applied Mathematics*, 1996, 70: 81-93.

[4] Z. L. Chen, Erratum to "Parallel machine scheduling with time dependent processing times" [Discrete Appl. Math. 70 (1996) 81-93], *Discrete Applied Mathematics*, 1997, 75: 103.

[5] 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*, 2004, 152: 1-13.

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

[7] M. Ji, Y. He, T. C. E. Cheng, Scheduling linear deteriorating jobs with an availability constraint on a single machine, *Theoretical Computer Science*, 2006, 362: 115-126.

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

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

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

[11] 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*, 1990, 47: 56-64.

[12] G. Mosheiov, Scheduling jobs under simple linear deterioration, *Computers and Operations Research*, 1994, 21: 653-659.

[13] G. Mosheiov, Multi-machine scheduling with linear deterioration. *INFOR*, 1998, 36(4): 205-214.

[14] M. Pinedo, Scheduling: Theory, Algorithms, and Systems, Prentice-Hall, Upper Saddle River, NJ, 2002.

[15] C. C. Wu, W. C. Lee, Scheduling linear deteriorating jobs to minimize makespan with an availability constraint on a single machine, *Information Processing Letters*, 2003, 87: 89-93.