Parallel-machine scheduling with time dependent processing times

Wen-Hung Kuo, Dar-Li Yang*

Department of Information Management, National Formosa University, Yun-Lin, 632, Taiwan, ROC

Received 7 June 2007; received in revised form 4 December 2007; accepted 8 December 2007

Communicated by D.-Z. Du

Abstract

In the literature, most of the parallel-machine scheduling problems, in which the processing time of a job is a linear function of its starting time, are proved to be NP-hard. In this paper, we study a parallel-machine scheduling problem in which the processing time of a job is a linear function of its starting time. The objectives are to minimize the total completion of all jobs and the total load on all machines respectively. We consider two linear functions of job starting time and show that the problems are polynomially solvable.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Parallel machine; Scheduling; Makespan; Total completion time; Total load

1. Introduction

Scheduling problems with time dependent processing times have received increasing attention in recent years. There are two different models considered in the literature. One model assumes that the processing time of a job is an increasing function of its starting time. A practical example of this model is described as follows. Assume that some products need to be processed by a cutting tool. The time required to process one product depends on the tool quality (e.g. sharpness of the cutting tool), that is, whether the tool is new or it has just undergone some maintenance. After some time, because of the wear of the cutting tool, the time required to process one product increases.

By contrast, the other model assumes that the processing time of a job is a decreasing function of its starting time. An application of this model is the so-called “learning effect”, which can be described by the following example. Assume that the worker has to assemble a large number of similar products. The time required by the worker to assemble one product depends on his knowledge, skills, organization of his workplace and others. During the assembling process, the worker learns how to produce. After some time, he is better skilled, his working place is better organized and his knowledge has increased. As a result of his learning, the time required to assemble one product decreases.

Usually, in the problems mentioned above, the processing time of a job can be described by a basic processing time and an increasing (or a decreasing) rate. Most of the relevant studies are confined to linear models. In the
standard form of linear models, the actual processing time of job $j$ is given by $p_j = a_j + b_j t_j$ where $a_j$, $t_j$ and $b_j$ denote the basic processing time, the starting time and the increasing rate of job $j$, respectively. In the literature, studies on such scheduling problems mainly deal with single-machine scheduling problems (see [1,4]). There are only a few studies dealing with parallel-machine scheduling problems. For example, Chen [2,3] considered a parallel-machine scheduling problem in which the processing time of a job is a simple linear function of its starting time (i.e. $p_j = b_j t_j$). The objective is to minimize the total completion of all jobs. They showed that the problem is NP-hard. Further, Mosheiov [9] showed that the parallel-machine scheduling problem of minimizing the makespan is also NP-hard. Therefore, for the standard linear model (i.e. $p_j = a_j + b_j t_j$), the parallel-machine problem of minimizing the makespan or the total completion time is easily proved to be NP-hard.

In this paper, we consider parallel-machine scheduling problems in which the actual processing time of job $J_j$ is $p_j = a_j + b t_j$ or $p_j = a_j - b t_j$. That is, the increasing (or decreasing) rates are identical for all jobs ($b_j = b$). The objectives are to minimize the total completion of all jobs and the total load on all machines, respectively. Moreover, Cheng et al. [5] mentioned that the model of $p_j = a_j + b t_j$ is a very realistic setting, particularly in the case of scheduling with deteriorating machines, when all processing times are increased by a common factor caused by the machine. We show that the problems are polynomially solvable.

### 2. Problem formulation

There are $n$ independent jobs to be processed on $m$ identical parallel machines ($M_i$, $i = 1, 2, \ldots, m$). Each of them is available at time $t_0 \geq 0$. Two linear models of the actual processing time of job $j$ ($J_j$, $j = 1, 2, \ldots, n$) are considered. The first model assumes that the actual processing time of job $j$ is $p_j = a_j + b t_j$ where $b > 0$ is an increasing rate. The other one assumes that the actual processing time of job $j$ is $p_j = a_j - b t_j$ where $0 < b < 1$ is a decreasing rate and $b \left( \sum_{i=1}^{n} a_i - a_j \right) < a_j$, $j = 1, 2, \ldots, n$. The condition of $0 < b < 1$ ensures that the decrease of each job processing time is less than one unit for every unit delay in its starting moment. The other condition of $b \left( \sum_{i=1}^{n} a_i - a_j \right) < a_j$ ensures that all job processing times are positive in a feasible schedule [8]. The objectives of the problems are to minimize the total completion time ($\sum C_j$) of all jobs and to minimize the total load ($\sum_{i=1}^{m} C_{max}^i$) on all machines, respectively. Following the three-field notation introduced by Graham et al. [6], the total completion time minimization problems and the total load minimization problems are denoted as $Pm/p_j = a_j \pm b t_j/\sum C_j$ and $Pm/p_j = a_j \pm b t_j/\sum C_{max}^i$, respectively.

### 3. Main results

In this paper, four scheduling problems are analyzed and the main results are summarized in Table 1.

For the problems $Pm/p_j = a_j + b t_j/\sum C_j$, $Pm/p_j = a_j - b t_j/\sum C_j$ and $Pm/p_j = a_j + b t_j/\sum C_{max}^i$, the optimal solution is that jobs are sorted in non-decreasing order of $a_j$ and then one by one the jobs in the sequence are assigned to each machine in turn.

As to the problem $Pm/p_j = a_j - b t_j/\sum C_{max}^i$, the optimal solution is that all jobs are assigned to one machine and arranged in non-increasing order of $a_j$. That is, the problem $Pm/p_j = a_j - b t_j/\sum C_{max}^i$ is equivalent to the problem $1/p_j = a_j - b t_j/\sum C_{max}^i$.

### 4. Main techniques

In this section, two useful techniques are provided in advance to analyze the parallel-machine scheduling problems in this paper. First, a technique to obtain the minimum sum of the products of two sequences of numbers is introduced in the following lemma.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Complexity</th>
<th>Optimal solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Pm/p_j = a_j + b t_j/\sum C_j$</td>
<td>$O(n \log(n))$</td>
<td>The jobs are sorted in non-decreasing order of $a_j$ and then one by one the jobs in the sequence are assigned to each machine in turn</td>
</tr>
<tr>
<td>$Pm/p_j = a_j - b t_j/\sum C_j$</td>
<td>$O(n \log(n))$</td>
<td>All the jobs are assigned to one machine and arranged in non-increasing order of $a_j$</td>
</tr>
<tr>
<td>$Pm/p_j = a_j + b t_j/\sum C_{max}^i$</td>
<td>$O(n \log(n))$</td>
<td></td>
</tr>
<tr>
<td>$Pm/p_j = a_j - b t_j/\sum C_{max}^i$</td>
<td>$O(n \log(n))$</td>
<td></td>
</tr>
</tbody>
</table>
Lemma 1. Let there be two sequences of numbers $x_i$ and $y_i$. The sum $\sum_i x_i y_i$ of products of the corresponding elements is the least if the sequences are monotonic in the opposite sense.

Proof. See [7]. □

Next, the group balance principle is given as follows. The group balance principle is to make the number of jobs be as equal as possible in each group. That is, if there are $n$ jobs to be assigned to $m$ groups, the number of jobs in each group is either $l - 1$ or $l$ where $l = \lceil \frac{n}{m} \rceil$ denotes the smallest integer larger than or equal to $\frac{n}{m}$.

5. Preliminary results

To solve the parallel-machine scheduling problems, the corresponding single-machine scheduling problems need to be analyzed first. Thus, the makespan minimization problems $1/p_j = a_j \pm bt_j/C_{\text{max}}$ and the total completion time minimization problems $1/p_j = a_j \pm bt_j/\sum C_j$ on a single machine are analyzed, respectively.

Lemma 2. For the problem $1/p_j = a_j + bt_j/\sum C_j$, if a job sequence $\pi_0 = (J_1, J_2, \ldots, J_n)$ and the starting time of the first job is $t_{0}$, then the total completion time of $\pi_0$ is

$$\sum C_j = w_0 t_0 + w_1 a_1 + w_2 a_2 + \cdots + w_{n-1} a_{n-1} + w_n a_n$$

where $w_0 = \sum_{k=1}^{n} (1 + b)^k$ and $w_j = \sum_{k=0}^{n-j} (1 + b)^k$, $j = 1, 2, \ldots, n$.

Lemma 3. For the problem $1/p_j = a_j - bt_j/C_{\text{max}}$, if a job sequence $\pi_0 = (J_1, J_2, \ldots, J_n)$ and the starting time of the first job is $t_{0}$, then the makespan of $\pi_0$ is

$$C_{\text{max}} = w_0 t_0 + w_1 a_1 + w_2 a_2 + \cdots + w_{n-1} a_{n-1} + w_n a_n$$

where $w_j = (1 + b)^{n-j}$, $j = 0, 1, 2, \ldots, n$.

Lemma 4. For the problem $1/p_j = a_j - bt_j/\sum C_j$, if a job sequence $\pi_0 = (J_1, J_2, \ldots, J_n)$ and the starting time of the first job is $t_{0}$, then the total completion time of $\pi_0$ is

$$\sum C_j = w_0 t_0 + w_1 a_1 + w_2 a_2 + \cdots + w_{n-1} a_{n-1} + w_n a_n$$

where $w_0 = \sum_{k=1}^{n} (1 - b)^k$ and $w_j = \sum_{k=0}^{n-j} (1 - b)^k$, $j = 1, 2, \ldots, n$.

Lemma 5. For the problem $1/p_j = a_j - bt_j/C_{\text{max}}$, if a job sequence $\pi_0 = (J_1, J_2, \ldots, J_n)$ and the starting time of the first job is $t_{0}$, then the makespan of $\pi_0$ is

$$C_{\text{max}} = w_0 t_0 + w_1 a_1 + w_2 a_2 + \cdots + w_{n-1} a_{n-1} + w_n a_n$$

where $w_j = (1 - b)^{n-j}$, $j = 0, 1, 2, \ldots, n$.

6. Minimization of the total completion time

In the following, we show that the problem $Pm/p_j = a_j \pm bt_j/\sum C_j$ can be solved in $O(n \log(n))$ time.

Theorem 1. For the problem $Pm/p_j = a_j + bt_j/\sum C_j$, it suffices to consider the group balance principle in a schedule.

Proof. Assume that there are $n$ jobs to be processed on $m$ parallel machines. Each of them is available at time $t_0 \geq 0$. Let $n_i$ denote the number of jobs assigned to machine $i$ and then $\sum_{i=1}^{m} n_i = n$. Let $a_{ij}$, $t_{ij}$ and $C_{ij}$ denote the basic processing time, the starting time and the completion time of job $j$ in machine $i$, respectively. The actual processing time of job $j$ in machine $i$ ($J_{ij}, i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n_i$) is $p_{ij} = a_{ij} + bt_{ij}$ where $b > 0$ is an increasing rate. Note that $t_{i1} = t_0$, $i = 1, 2, \ldots, m$. Then, from Lemma 2, the total completion time of all jobs in these parallel machines is calculated as follows.

$$\sum_{i=1}^{m} \sum_{j=1}^{n_i} C_{ij} = \sum_{i=1}^{m} \left( w_0 t_0 + \left( \sum_{j=1}^{n_i} w_{ij} a_{ij} \right) \right)$$

where $w_{ij} = \sum_{k=1}^{n_i-j} (1 + b)^k$ for $i = 1, 2, \ldots, m$ and $w_{ij} = \sum_{k=0}^{n_i-j} (1 + b)^k$ for $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n_i$. 
If it does not satisfy the group balance principle in the schedule, there exists at least a pair of machines in which the difference of the numbers of jobs is greater than one. Without loss of generality, assume that the number of jobs in machine $M_i$ is $n_i$ and that of $M_j$ is $n_j$ ($n_i > n_j$). In addition, $n_i - n_j = z$ where $z$ is an integer greater than one ($z > 1$). Let $\Pi = (\pi_1, \pi_2, \ldots, \pi_i, \pi_j, \ldots, \pi_m)$ denote the schedule where the number of jobs in schedule $\pi_i$ (on machine $M_i$) is $n_i$ and that for $\pi_j$ (on machine $M_j$) is $n_j$. Let $\Pi' = (\pi_1, \pi_2, \ldots, \pi_i', \pi_j', \ldots, \pi_m)$ denote the schedule where $\pi_i'$ and $\pi_j'$ denote the schedules of jobs processed on $M_i$ and $M_j$ after removing the first job from schedule $\pi_i$ to the first position of schedule $\pi_j$ in schedule $\Pi$, respectively. That is, the number of jobs in schedule $\pi_i'$ (on machine $M_i$) becomes $(n_i - 1)$ and that for $\pi_j'$ (on machine $M_j$) becomes $(n_j + 1)$. Let

\[ W = \left( W_1, \sum_{k=1}^{n_i} (1 + b)^k, \sum_{k=0}^{n_i-1} (1 + b)^k, \ldots, \sum_{k=0}^{n_i-2} (1 + b)^k, 1, W_2, \right. \]

\[ \begin{array}{c}
\sum_{k=1}^{n_j} (1 + b)^k, \sum_{k=0}^{n_j-1} (1 + b)^k, \ldots, \sum_{k=0}^{n_j-2} (1 + b)^k, 1, W_3 \end{array} \]

and

\[ W' = \left( W_1, \sum_{k=1}^{n_i-1} (1 + b)^k, \sum_{k=0}^{n_i-2} (1 + b)^k, \ldots, \sum_{k=0}^{n_i-3} (1 + b)^k, 1, W_2, \right. \]

\[ \begin{array}{c}
\sum_{k=1}^{n_j+1} (1 + b)^k, \sum_{k=0}^{n_j} (1 + b)^k, \ldots, \sum_{k=0}^{n_j-2} (1 + b)^k, 1, W_3 \end{array} \]

denote the sets of weights in schedule $\Pi$ and $\Pi'$, respectively. $W_1, W_2$ and $W_3$ are the subsets of weights in schedule $\Pi$ (or $\Pi'$). Thus, if we remove the first job from schedule $\pi_i$ to the first position of schedule $\pi_j$ in schedule $\Pi$, then compare with set $W$, it adds the weight of $\sum_{k=1}^{n_i+1} (1 + b)^k$ to set $W'$ and subtracts the weight of $\sum_{k=1}^{n_i} (1 + b)^k$ from set $W$. Let $T_{ij}(\Pi)$ denote the sum of the total completion times of jobs in $\pi_i$ and $\pi_j$ in schedule $\Pi$ and $T_{ij}(\Pi')$ denote that for $\pi_i'$ and $\pi_j'$ in schedule $\Pi'$. Then

\[ T_{ij}(\Pi) = t_0 \sum_{k=1}^{n_i} (1 + b)^k + a_1 \sum_{k=0}^{n_i-1} (1 + b)^k + a_2 \sum_{k=0}^{n_i-2} (1 + b)^k + \cdots + a_i n_i, \]

\[ + t_0 \sum_{k=1}^{n_j} (1 + b)^k + a_1 \sum_{k=0}^{n_j-1} (1 + b)^k + a_2 \sum_{k=0}^{n_j-2} (1 + b)^k + \cdots + a_j n_j \]

and

\[ T_{ij}(\Pi') = t_0 \sum_{k=1}^{n_i-1} (1 + b)^k + a_2 \sum_{k=0}^{n_i-2} (1 + b)^k + \cdots + a_i n_i, \]

\[ + t_0 \sum_{k=1}^{n_j+1} (1 + b)^k + a_1 \sum_{k=0}^{n_j} (1 + b)^k + a_2 \sum_{k=0}^{n_j-1} (1 + b)^k + \cdots + a_j n_j. \]

Therefore, the change of the total completion time is calculated as follows.

\[ T_{ij}(\Pi') - T_{ij}(\Pi) = \left( t_0 \sum_{k=1}^{n_i-1} (1 + b)^k + t_0 \sum_{k=1}^{n_j+1} (1 + b)^k + a_i \sum_{k=0}^{n_i} (1 + b)^k \right) \]
We use the same notations as in the proof of Theorem 1. The proof is similar to that of Theorem 2 since jobs are assigned one by one to each machine in turn. Let \( \Pi \) be the set of weights in \( \pi \). Theorem 4. If \( r \neq 0 \), without loss of generality, let each of the first \( r \) machines process \( l = \left\lceil \frac{n}{m} \right\rceil \) jobs and each of the other machines process \( (l - 1) \) jobs, then the set of weights for the schedule is expressed as follows:

\[
W = \left( w_{10}, w_{11}, w_{12}, \ldots, w_{1n_1}, w_{20}, w_{21}, w_{22}, \ldots, w_{2n_2}, \ldots, w_{m0}, w_{m1}, w_{m2}, \ldots, w_{mn_m} \right)
\]

\[
= \left( \sum_{k=1}^{l} (1 + b)^k, \sum_{k=0}^{l-1} (1 + b)^k, 1, \ldots, \sum_{k=1}^{l} (1 + b)^k, \sum_{k=0}^{l-1} (1 + b)^k, 1 \right)
\]

Thus, if \( a_{11} \leq a_{21} \leq \cdots \leq a_{n1} \leq a_{22} \leq \cdots \leq a_{m2} \leq \cdots \leq a_{1,l-1} \leq a_{2,l-1} \leq \cdots \leq a_{m,l-1} \leq a_{1l} \leq a_{2l} \leq \cdots \leq a_{rl}, \) from Lemma 1, the total completion time is minimized since \( t_1 = t_0, i = 1, 2, \ldots, m. \) On the other hand, if \( r = 0 \), the above result can also be proved in a similar manner. This completes the proof of Theorem 2.

Theorem 3. For the problem \( Pm/p_j = a_j - bt_j/\sum C_j \), it suffices to consider the group balance principle in a schedule.

Proof. The proof is similar to that of Theorem 1.

Theorem 4. For the problem \( Pm/p_j = a_j - bt_j/\sum C_j \), there exists an optimal schedule in which jobs are sorted in non-decreasing order of their basic processing times \( (a_j) \) and then one by one the jobs in the sequence are assigned to each machine in turn.

Proof. The proof is similar to that of Theorem 2.

Thus, from Theorems 1–4, the problems \( P/p_j = a_j + bt_j/\sum C_j \) and \( P/p_j = a_j - bt_j/\sum C_j \) can be solved in \( O(n \log(n)) \) time.

7. Minimization of the total load on all machines

Let \( C_{\text{max}}^j \) denote the largest completion time on machine \( i, i = 1, 2, \ldots, m. \) Let \( n_i \) denote the number of jobs assigned to machine \( i \) \( (M_i, i = 1, 2, \ldots, m). \) The total load is the sum of the largest completion times on all the
machines or the total time that all machines work. Therefore, for the model \( p_j = a_j + bt_j \), from Lemma 3, the total load is as follows:

\[
\sum_{i=1}^{m} C_{i \max}^i = \sum_{i=1}^{m} \left( (1 + b)^{n_i} t_0 + (1 + b)^{n_{i-1}} a_1 + (1 + b)^{n_{i-2}} a_2 + \cdots + (1 + b)^{a_{n_i-1}} + a_i \right).
\]

Thus, we have the following theorems.

**Theorem 5.** For the problem \( P_m/p_j = a_j + bt_j/\sum C_{i \max}^i \), it suffices to consider the group balance principle in a schedule.

**Proof.** The proof is similar to that of Theorem 1. \( \square \)

**Theorem 6.** For the problem \( P_m/p_j = a_j + bt_j/\sum C_{i \max}^i \), there exists an optimal schedule in which jobs are sorted in non-decreasing order of their basic processing times (\( a_j \)) and then one by one the jobs in the sequence are assigned to each machine in turn.

**Proof.** The proof is similar to that of Theorem 2. \( \square \)

As for the model \( p_j = a_j - bt_j \), from Lemma 5, we can see that the later the starting time of a job is, the shorter the actual processing time of a job is. Therefore, the total load on all machines is minimized when all jobs are assigned to one machine and arranged in non-increasing order of their basic processing times. Hence, we have the following theorem.

**Theorem 7.** For the problem \( P_m/p_j = a_j - bt_j/\sum C_{i \max}^i \), there exists an optimal schedule in which all jobs are assigned to one machine and arranged in non-increasing order of their basic processing times. That is, the problem \( P_m/p_j = a_j - bt_j/\sum C_{i \max}^i \) is equivalent to the problem \( \lambda/p_j = a_j - bt_j/C_{\max} \).

**Proof.** The proof follows from the above analysis. \( \square \)

Thus, from Theorems 5 and 7, the problems \( P_m/p_j = a_j + bt_j/\sum C_{i \max}^i \) and \( P_m/p_j = a_j - bt_j/\sum C_{i \max}^i \) can also be solved in \( O(n \log(n)) \) time. However, the optimal solutions are different for the two problems.

**8. Conclusions**

In this paper, we focus on parallel-machine scheduling problems with time-dependent processing times. We discuss two linear models \( p_j = a_j + bt_j \) and \( p_j = a_j - bt_j \). The objectives are to minimize the total completion time of all jobs and the total load on all machines, respectively. It is interesting that the optimal job sequences are the same for the objective of the total completion time minimization in these two models. However, the optimal job sequences are different for the objective of the total load minimization in these two models.

**Acknowledgements**

The authors are grateful to the referee for his helpful comments and suggestions on an earlier version of this paper. This research is supported in part by the National Science Council of Taiwan, Republic of China, under grant number NSC-95-2221-E-150-050.

**References**


