Research Article

Parallel-Machine Scheduling Problems with Past-Sequence-Dependent Delivery Times and Aging Maintenance

Wei-min Ma,1 Li Sun,1 Xue-qin Zeng,1 and Lei Ning2

1 School of Economics and Management, Tongji University, Shanghai 201804, China
2 Shanghai Chancool Food Technology Co., Ltd., Shanghai 200233, China

Correspondence should be addressed to Li Sun; sunlitij3@126.com

Received 20 August 2014; Accepted 11 October 2014; Published 28 October 2014

Academic Editor: Yunqiang Yin

Copyright © 2014 Wei-min Ma et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We consider parallel-machine scheduling problems with past-sequence-dependent (psd) delivery times and aging maintenance. The delivery time is proportional to the waiting time in the system. Each machine has an aging maintenance activity. We develop polynomial algorithms to three versions of the problem to minimize the total absolute deviation of job completion times, the total load, and the total completion time.

1. Introduction

A common assumption in classical scheduling theory is that the machines are available at all times. However, there are many situations where machines need to be maintained and become unavailable during certain periods. Cheng et al. [1] study single-machine scheduling with deteriorating job processing times. Lee and Leon [2] investigated single machine scheduling with a rate-modifying activity. Kubzin and Strusevich [3] presented a setting where the maintenance activity was deteriorating; that is, delaying the maintenance activity increased the time required to perform it. Lee and Lin [4] considered single machine scheduling with maintenance and repair rate-modifying activity. Lee and Chen [5] discussed parallel-machine scheduling where each machine must be maintained once during the planning horizon. Cheng et al. [1, 6] considered the unrelated parallel-machine scheduling with aging maintenance activities. Cheng et al. [7] studied a single machine problem of common due-window assignment and scheduling of aging jobs and a maintenance activity simultaneously.

Browne and Yechiali [8] claimed that the time and effort required to put out a fire increase if there was a delay in the start of the fire-fighting effort. Scheduling in this setting is known as scheduling with aging jobs, which was first independently introduced by J. N. D. Gupta and S. K. Gupta [9] and Browne and Yechiali [8]. Since then, models of scheduling with deteriorating or aging jobs have been extensively studied from a variety of perspectives. More recent papers which have considered deterioration include Zhao and Tang [10], Cheng et al. [1], Wang et al. [11], Lai and Lee [12], Cheng et al. [13], Huang and Wang [14], Yin et al. [15], Lee et al. [16], Zhao and Tang [17], Wang and Liu [18], Zhao and Tang [19], Wu and Lee [20], Wang [21], Miao et al. [22], Wu and Lee [23], Yin et al. [24], Oron [25], Sicilia et al. [26], Wang et al. [27], Yin et al. [28], J.-B. Wang and J.-J. Wang [29], Xu et al. [30], Yin et al. [31], Bai et al. [32], and Ji et al. [33].

In modern industrial production, the manufacturing environment has an increasing influence on the processing times of jobs. A growing evidence displays that waiting time or manufacturing environment may have a disadvantageous effect on the total processing time of a job before delivery to the customer. For example, an electronic component may be exposed to certain electromagnetic or radioactive fields while waiting in the machines preprocessing area and regulatory authorities require the component to be treated for an amount of time proportional to the jobs’ exposure time.
to these fields. This treatment can be performed after the component has been processed by the machine but before it is delivered to the customer so it can be delivered with a guarantee. Such an extra time for eliminating adverse effects between the main processing and the delivery of a job is viewed as a past-sequence-dependent (psd) delivery time [34]. As usual, we suppose the treatment of the adverse effect for a job does not occupy any machine and has no relation to the schedule of the jobs main processing. In addition, we also assume that the psd delivery time of a job is proportional to the job’s waiting time.

Koulamas and Kyparisis [34] assumed that the psd delivery time of a job is proportional to the job’s waiting time, that is, the start time of processing. They proved some problems could be polynomially solvable. Liu et al. [35] studied a single machine batch scheduling with past-sequence-dependent delivery times, which was introduced in Koulamas and Kyparisis [34]. Liu [36] considered identical parallel-machine scheduling problem with past-sequence-dependent delivery times and learning effect. Shen and Wu [37] introduced the single machine past-sequence-dependent delivery times scheduling with general learning effects. Yin et al. [22] studied a single machine batch scheduling with an unavailability interval. Yin et al. [38] introduced a single-machine scheduling with past-sequence-dependent delivery times and a linear deterioration.

To the best of our knowledge, the aging maintenance scheduling problems with psd delivery time have not been investigated. In this paper, we introduce parallel machines scheduling model with aging maintenance and psd delivery time. We will present that these scheduling problems are polynomially solvable. The remaining part of this paper is organized as follows. In Section 2, a precise formulation of the problem is given. We consider problems of minimizing the total absolute deviation of job completion times, the total load on all machines, and the total completion time in Section 3. The last section contains some conclusions of our model.

2. Problem Formulation

There are \( n \) independent and nonpreemptive jobs simultaneously available for processing and \( m \) identical parallel machines. Each machine can handle one job at a time and preemption is not allowed. If \( m \geq n \), the problem is trivial. Therefore, we assume that \( m < n \) throughout the paper.

Let \( a_{ij} \) (\( p_{ij} \)) be the normal (actual) processing time of job \( J_{ij} \); let \( a_{ij[k]} \) (\( p_{ij[k]} \)) be the normal (actual) processing time of the job \( J_{ij[k]} \) if it is scheduled in the \( k \)th position on machine \( M_i \) in a sequence. Due to the effects of aging, maintenance may be performed on the machine to improve its production efficiency. We suppose that at most one maintenance operation is allowed on one machine throughout the scheduling horizon. However, the starting time of the maintenance operation in the sequence is not known in advance. It can be scheduled immediately after completing the processing of any job. We further assume the following: (1) after maintenance, the machine will revert to its initial condition and the aging effects will start anew and (2) the machine maintenance duration is a linear function of its starting time and is represented by \( f(t) = \mu + \sigma t \), where \( \mu > 0 \) is the basic maintenance time, \( \sigma > 0 \) is a maintenance factor, and \( t \) is the starting time of the maintenance operation.

Suppose the position of maintenance operation is \( k_i \) on machine \( M_i \). Hence, the machine maintenance duration is \( f(t) = \mu = \mu + \sigma, \) which is introduced in Liu et al. [39]. As in Liu et al. [39], we consider that the actual processing time of job \( J_{ij} \) which is scheduled on machine \( M_i \) in position \( j \) \( (j = 1, 2, \ldots, n_i) \) is given by

\[
p_{ij} = \begin{cases} a_{ij} + bt = a_{ij} + b \sum_{l=1}^{j-1} p_{ij[l]}, & \text{if } j \leq k_i, \\ a_{ij} + bt = a_{ij} + b \sum_{l=k_i}^{j-1} p_{ij[l]}, & \text{if } j > k_i. \end{cases}
\]

As in Koulamas and Kyparisis [34], the processing of job \( J_{ij} \) must be followed by the psd delivery time \( q_{i[r]} \), which can be computed as

\[
q_{i[1]} = 0, \quad q_{i[r]} = \gamma W_{i[r]} = \gamma \sum_{l=1}^{r-1} p_{ij[l]}, \quad r = 2, 3, \ldots, n_i.
\]

\( \gamma \geq 0 \) is a normalizing constant and \( W_{i[r]} \) denotes the waiting time of job \( J_{ij} \). In addition, it is supposed that the postprocessing operation of any job \( J_{ij} \) modeled by its delivery time \( q_{i[r]} \) is performed off-line; consequently, it is not affected by the availability of the machine and it can commence immediately upon completion of the main operation resulting in

\[
C_{ij} = W_{i[j]} + p_{ij[j]} + q_{ij[j]} = (1 + \gamma) \sum_{l=1}^{j-1} p_{ij[l]} + p_{ij[j]}, \quad \text{if } j \leq k_i,
\]

\[
C_{ij} = (1 + \gamma) \left( \mu + (1 + \sigma) \sum_{l=1}^{k_i} p_{ij[l]} + p_{ij[k_i]} \right) + p_{ij[j]}, \quad \text{if } j > k_i,
\]

where \( C_{ij} \) denotes the completion time of job \( J_{ij} \).

For convenience, we denote the psd delivery times given in (2) by \( q_{psd} \). Let \( TADC_{ij} \) denote the total absolute deviation of jobs’ completion times on machine \( M_i \); that is,

\[
\sum_{j=1}^{m_i} TADC_{ij} = \sum_{j=1}^{m_i} \sum_{k=1}^{n_i} |C_{ij[k]} - C_{ij[j]}|. \]

Let \( L_i \) indicate the load of machine \( M_i \); that is, \( L_i = \max(C_{ij}), \) \( r = 1, 2, \ldots, n_i \).

We, respectively, consider the minimization of the following objective functions: the sum of the total absolute deviation of jobs’ completion times on each machine \( \sum_{i=1}^{m} TADC_{ij} \), the total loads on all machines \( \sum_{i=1}^{m} L_i \), and the total completion time \( \sum_{i=1}^{m} \sum_{j=1}^{n_i} C_{ij} \). Using the three-field notation introduced
by Graham et al. for scheduling problems, we denote our problems as
\[
P_m \mid q_{psd}, P_{i[j]} = a_{i[j]} + bt, \text{ma} \parallel f, \quad f \in \left\{ \sum \sum C_{ij}, \sum L_p \sum \text{TADC}_i \right\}.
\]  
(4)

Before presenting the main results, we first present several Lemmas and some notations that will be used in the proofs in sequel. If the number of jobs \(n_i\) and the position of the job preceding the maintenance operation \(k_i\) on machine \(M_i\) are known in advance, the actual processing times and the completion times of jobs on machine \(M_i\) are as follows:
\[
P_{i[1]} = a_{i[1]}, \quad C_{i[1]} = P_{i[1]} = a_{i[1]}
\]
\[
P_{i[2]} = a_{i[2]} + bp_{i[1]} = a_{i[2]} + ba_{i[j]}
\]
\[
P_{i[k]} = a_{i[k]} + ba_{i[k-1]} + b(1 + b)a_{i[k-2]}
\]
\[
\vdots
\]
\[
P_{i[n]} = a_{i[n]} + ba_{i[n-1]} + \cdots + b(1 + b)^{k-2}a_{i[1]} + b(1 + b)^{k-1}a_{i[k]}
\]
\[
C_{i[k+1]} = (1 + \gamma) \mu + (1 + \sigma)\sum_{j=1}^{k_i} (1 + b)^{k_j-1}a_{i[j]}
\]
\[
+ a_{i[k+1]}
\]
\[
P_{i[n]} = a_{i[n]} + ba_{i[n-1]} + \cdots + b(1 + b)^{n-2}a_{i[k+1]}
\]
\[
C_{i[n]} = (1 + \gamma) \mu + (1 + \sigma)\sum_{j=1}^{k_i} (1 + b)^{k_j-1}a_{i[j]}
\]
\[
+ \sum_{j=k+1}^{n-1} (1 + \gamma + b)(1 + b)^{n-j-1}a_{i[j]} + a_{i[n]}.
\]

Let \(P(n, m, k) = (n_1, n_2, \ldots, n_m; k_1, k_2, \ldots, k_m)\) denote the allocation vector. We derive a lemma to bound \(P(n, m, k)\).

**Lemma 1.** The number of \(P(n, m, k)\) vector is bounded from above by \((n + 1)^{2m-1}/m!\).

**Proof.** For machine \(M_1, n_1\) is chosen from \([0, 1, \ldots, n]\) and there are \(n + 1\) options for \(n_1\). When \(n_1\) is determined, \(k_1\) has at most \(n + 1\) options. After \(n_1\) is determined, \(n_2\) is chosen from \([0, 1, \ldots, n - n_1]\) \(\subseteq [0, 1, \ldots, n]\) at most \(n + 1\) times. Subsequently, \(k_2\) has at most \(n + 1\) options, too. For machines \(M_2, \ldots, M_{m-1}\), we similarly select \(n_3, \ldots, n_{m-1}\) and \(k_3, \ldots, k_{m-1}\). Note that, for the last machine \(M_m, n_m\) is determined after choosing \(n_1, \ldots, n_{m-1}\), because \(\sum n_i = n\), and \(k_m\) has at most \(n + 1\) options. Considering that \(m\) machines are identical, there are \((n + 1)^{2m-1}/m!\) possibilities for \(P(n, m, k)\). This completes the proof. \(\square\)

**Lemma 2** (see [40]). Let there be two sequences of numbers \(x_i, y_i\). The sum \(\sum_i x_i y_i\) of products of the corresponding elements is the least (largest) if the sequences are monotonic in the opposite (same) sense.

### 3. Parallel-Machine Scheduling Problems

#### 3.1. The Total Completion Time

In this section, we consider the problem of minimizing the total completion time; that is, \(P_m \mid q_{psd}, P_{i[j]} = a_{i[j]} + bt, \text{ma}\). For machine \(M_1\), from the above analysis, we derive the total completion time on this machine:
\[
C_{i[i]} = (n_i - k_i)(1 + \gamma) \mu
\]
\[
+ \sum_{j=1}^{k_i} \left( 1 + (1 + \gamma + b)(1 + b)^{k_j-1} - \frac{1}{b} \right) + (n_i - k_i)(1 + \sigma)(1 + b)^{k_j-1})a_{i[j]} + \sum_{j=k_i+1}^{n_i} \left( 1 + (1 + \gamma + b)(1 + b)^{n_j-1} - \frac{1}{b} \right) a_{i[j]}.
\]
\(6\)

Therefore, the total completion time is
\[
\sum_{i=1}^{m} \sum_{j=1}^{n_i} C_{ij}
\]
\[
= \sum_{i=1}^{m} (n_i - k_i)(1 + \gamma) \mu
\]
\[
+ \sum_{i=1}^{m} \sum_{j=1}^{k_i} \left( 1 + (1 + \gamma + b)(1 + b)^{k_j-1} - \frac{1}{b} \right) + (n_i - k_i)(1 + \sigma)(1 + b)^{k_j-1})a_{i[j]} + \sum_{i=1}^{m} \sum_{j=k_i+1}^{n_i} \left( 1 + (1 + \gamma + b)(1 + b)^{n_j-1} - \frac{1}{b} \right) a_{i[j]}.
\]
\(7\)
investigate the problem of minimizing the sum of load on all machines. The problem can be solved in $O(n \log n)$ time.

When $n_i$ and $k_i$ are given, $\sum_{i=1}^{m} (n_i-k_i)(1+\gamma)\mu$ is a constant. And $\sum_{i=1}^{m} \sum_{j=1}^{n_i} w_{ij}a_{ij}$ can be viewed as the scalar product of the $w_{ij}$ vector and $a_{ij}$ vector. Therefore, on machine $M_i$, if the number $n_i$ of jobs and the maintenance position $k_i$ are known, by Lemma 2, all jobs are sorted in nondecreasing order of their normal processing times first. Then the largest job is assigned to the position with the smallest value of $w_{ij}$, the second largest job to the position with the second smallest value of $w_{ij}$, and so on. The time complexity of sorting algorithm on machine $M_i$ is $O(n_i \log n_i)$. Together with Lemma 1, we derive the following theorem.

**Theorem 3.** The problem $P_{m, i} a_{ij} p_{ij} = a_{ij} + bt, ma \sum C_{ij}$ can be solved in $O(n^{2m-2} \log n)$ time.

### 3.2. The Sum of Load on All Machines.
In this section, we investigate the problem of minimizing the sum of load on all machines; that is, $P_{m, i} a_{ij} p_{ij} = a_{ij} + bt, ma \sum L_i$. For machine $M_i$, from the above analysis, we calculate the sum of load on this machine:

$$L_i = (1 + \gamma) \mu$$

$$+ (1 + \gamma)(1 + \sigma) \sum_{j=1}^{k_i} (1 + b)^{k_i-j} a_{ij}$$

$$+ \sum_{j=k_i+1}^{n_i} (1 + \gamma + b) (1 + b)^{n_i-j} a_{ij} + a_{ij}$$

Hence, the sum of load on all machines is

$$\sum_{i=1}^{m} L_i = m (1 + \gamma) \mu$$

$$+ \sum_{i=1}^{m} \sum_{j=1}^{k_i} (1 + \gamma)(1 + \sigma) (1 + b)^{k_i-j} a_{ij}$$

$$+ \sum_{i=1}^{m} \sum_{j=k_i+1}^{n_i} (1 + \gamma + b) (1 + b)^{n_i-j} a_{ij}$$

$$+ \sum_{i=1}^{m} a_{i[n_i]} = m (1 + \gamma) \mu + \sum_{i=1}^{m} \sum_{j=1}^{n_i} w_{ij} a_{ij}$$

Since $m(1 + \gamma)\mu$ is a constant, if $n_i$ and $k_i$ are given, $\sum_{i=1}^{m} \sum_{j=1}^{n_i} w_{ij}a_{ij}$ can be viewed as the scalar product of the $w_{ij}$ vector and $a_{ij}$ vector. Similar to the above analysis, on machine $M_i$, the time complexity of sorting algorithm is $O(n_i \log n_i)$. Together with Lemma 1, we conclude the following theorem.

**Theorem 4.** The problem $P_{m, i} a_{ij} p_{ij} = a_{ij} + bt, ma \sum L_i$ can be solved in $O(n^{2m-1} \log n)$ time.

### 3.3. The Sum of TADC on All Machines.
In this section, we present the problem of minimizing the sum of TADC on all machines; that is, $P_{m, i} a_{ij} p_{ij} = a_{ij} + bt, ma \sum L_i$. For machine $M_i$, from the above analysis, we express the sum of TADC on this machine:

$$\text{TADC}_i$$

$$= \sum_{j=k_i+1}^{n_i} (2j - 1 - n_i) (1 + \gamma) \mu$$

$$+ \sum_{j=1}^{k_i} \sum_{h=j+1}^{k_i} (2h - 1 - n_i) (1 + \gamma + b)$$

$$\times (1 + b)^{h-j-1}$$

$$+ \sum_{j=k_i+1}^{n_i} (2h - 1 - n_i) (1 + \gamma)$$

$$\times (1 + \sigma)(1 + b)^{h-j} a_{ij}$$

$$+ \sum_{j=k_i+1}^{n_i} (2j - 1 - n_i) + \sum_{h=k_i+1}^{n_i} (2h - 1 - n_i) (1 + \gamma + b)$$

$$\times (1 + b)^{h-j-1} a_{ij}$$

$$+ (n_i - 1) a_{i[n_i]}$$

(12)
Thus, the sum of TADC on all machines is

$$\sum_{i=1}^{m} TADC_{i} = \sum_{i=1}^{m} \sum_{j=k_{i}+1}^{n_{i}} (2j - 1 - n_{i})(1 + \gamma) \mu$$

$$+ \sum_{i=1}^{m} \sum_{j=1}^{k_{i}} (2j - 1 - n_{i})(1 + \gamma + b)(1 + b)^{h-j-1}$$

$$+ \sum_{h=j+1}^{k_{i}} (2h - 1 - n_{i})(1 + \gamma + b)(1 + b)^{h-j-1}$$

$$+ \sum_{h=k_{i}+1}^{k_{i}+1} (2h - 1 - n_{i})(1 + \gamma)(1 + \sigma)$$

$$\times (1 + b)^{h-j-1} a_{i[j]}$$

$$+ \sum_{i=1}^{m} \sum_{j=1}^{n_{i}} (2j - 1 - n_{i})(1 + \gamma) \mu$$

$$+ \sum_{i=1}^{m} \sum_{j=1}^{n_{i}} w_{ij} a_{i[j]}.$$  \hspace{1cm} (13)

where

$$w_{ij} = \begin{cases} (2j-n_{i}) + \sum_{k_{i}+1}^{n_{i}} (2k-n_{i})(1+\gamma+b)(1+b)^{k-j-1} \\ \sum_{h=k_{i}+1}^{n_{i}} (2h-n_{i})(1+\gamma+\sigma)(1+b)^{h-j}, \\ i = 1, 2, \ldots, m, \ \ j = 1, 2, \ldots, k_{i}, \\ (2j-n_{i}) + \sum_{h=j+1}^{n_{i}} (2h-n_{i})(1+\gamma+b)(1+b)^{h-j}, \\ i = 1, 2, \ldots, m, \ \ j = k_{i} + 1, k_{i} + 2, \ldots, n_{i} - 1, \\ n_{i} - 1, \\ i = 1, 2, \ldots, m, \ j = n_{i}. \end{cases}$$  \hspace{1cm} (14)

Since $\sum_{i=1}^{m} \sum_{j=k_{i}+1}^{n_{i}} (2j - 1 - n_{i})(1 + \gamma) \mu$ is a constant, when $n_{i}$ and $k_{i}$ are given, $\sum_{i=1}^{m} \sum_{j=1}^{n_{i}} w_{ij} a_{i[j]}$ can be viewed as the scalar product of the $w_{ij}$ vector and $a_{i[j]}$ vector. Similar to the above analysis, the time complexity of scheduling algorithm on machine $M_{j}$ is $O(n_{i}\log n_{i})$. Together with Lemma 1, we obtain the following theorem.

**Theorem 5.** The problem $P_{m|\text{psd}, \text{deter|a}|\text{st}} = a_{i[j]} + bt, m\text{al} \sum TADC_{i}$ can be solved in $O(n^{2m-1}\mu \log n)$ time.

### 4. Conclusions

In this paper we study parallel-machine scheduling problems with past-sequence-dependent (psd) delivery times and aging maintenance. The delivery time of a job is proportional to its waiting time. Each machine has an aging maintenance activity. Polynomially solvable problems have been explored for parallel-machine scheduling. The research extends the existed models of parallel machine scheduling with delivery times. For future research, it will be worth extending the problem to multiple maintenances or flow-shop environment.

**Conflict of Interests**

The authors declare that there is no conflict of interests regarding the publication of this paper.

**Acknowledgments**

The work was partly supported by the National Natural Science Foundation of China (71071113), Ph.D. Programs Foundation of the Ministry of Education of China (20100072110011), Shanghai Philosophical and Social Science Foundation of the Ministry of Education of China (2010BZH003), and the Fundamental Research Funds for the Central Universities. The authors would like to thank the anonymous referees for their constructive comments and suggestion.

**References**


