Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 6093029, 5 pages
https://doi.org/10.1155/2017/6093029

Research Article

Batch Scheduling with Proportional-Linear Deterioration and Outsourcing

Cuixia Miao,[1] Fanxiao Meng,[1] Juan Zou,[1,2] and Binglin Jia[3]

[1] *School of Mathematical Sciences, Qufu Normal University, Qufu, Shandong 273165, China*
[2] *School of Mathematics and Statistics, Zhengzhou University, Zhengzhou 450001, China*
[3] *Caoxian No. 1 Middle School, Heze, Shandong 274400, China*

Correspondence should be addressed to Cuixia Miao; miaocuixia@126.com

Received 10 March 2017; Accepted 23 April 2017; Published 22 June 2017

Academic Editor: Marco Pizzarelli

We consider the bounded parallel-batch scheduling with proportional-linear deterioration and outsourcing, in which the actual processing time is $p_j = \alpha_j(A+Dt)$ or $p_j = \alpha_j t$. A job is either accepted and processed in batches on a single machine by manufactures themselves or outsourced to the third party with a certain penalty having to be paid. The objective is to minimize the maximum completion time of the accepted jobs and the total penalty of the outsourced jobs. For the $p_j = \alpha_j(A + Dt)$ model, when all the jobs are released at time zero, we show that the problem is NP-hard and present a pseudo-polynomial time algorithm, respectively. For the $p_j = \alpha_j t$ model, when the jobs have distinct m $(<n)$ release dates, we provide a dynamic programming algorithm, where n is the number of jobs.

1. Introduction

The parallel-batch scheduling is motivated by burn-in operations in semiconductor manufacturing; see Lee et al. [1] for more details of the background. By Brucker et al. [2], there are two distinct models: the *bounded model*, in which the bound b for each batch size is effective, that is, $b < n$, and the *unbounded model*, in which there is effectively no limit on the size of batch, that is, $b \geq n$. The extensive survey of different models and results was provided both by Potts and Kovalyov [3] and Zhang and Cao [4].

Scheduling with deterioration was first considered by J. N. D. Gupta and S. K. Gupta [5], and Browne and Yechiali [6]. From then on, this scheduling model has been extensively studied. The monograph by Gawiejnowicz [7] presents this scheduling from different perspectives and covers results and examples. Ji and Cheng [8], Liu et al. [9], and Miao [10] gave some new results for this scheduling.

In classical scheduling literatures, all jobs must be processed. In the practical applications, however, this may not be true. Due to the limited resources, the scheduler can have the option to outsource or reject some jobs. However, outsourced jobs will incur penalties. The scheduling with outsourcing was first considered by Bartal et al. [11]. They studied both the offline and the online versions of scheduling with outsourcing on identical parallel machines, the objective is to minimize the maximum completion time of the accepted jobs and the total penalty of the outsourced jobs.

Cao and Yang [12] presented a PTAS for the combined model of the parallel-batch and rejection where jobs arrive dynamically. The objective is to minimize the maximum completion time of the accepted jobs and the total penalty of the outsourced ones. Lu et al. ([13, 14]) considered the unbounded and bounded parallel-batch scheduling problems with outsourcing on a single machine. Cheng and Sun [15] considered the scheduling with linear deteriorating jobs and rejection on a single machine; they gave the proofs of the NP-hardness and presented some pseudo-polynomial time algorithms and FPTASs for some objectives. Miao [16] considered the bounded parallel-batch scheduling with rejection in the V chapter of her thesis.

In this paper, we consider the bounded parallel-batch scheduling with proportional-linear deterioration and outsourcing on a single machine. The objective is to minimize the maximum completion time of the accepted jobs and the total penalty of the outsourced jobs. We analyze the NP-hardness and present pseudo-polynomial time dynamic programming algorithms for two deterioration models

2. Problem Description and Preliminaries

There are n independent nonpreemptive deteriorating jobs $J = \{J_1, \dots, J_n\}$ to be processed on a single batch machine. The actual processing time of job J_j $(j = 1, \dots, n)$ is $p_j = \alpha_j t$ or $p_j = \alpha_j(A + Dt)$, where $A, D > 0$, α_j (≥ 0), and t denote the deteriorating rate and starting time, respectively. J_j has release date r_j and outsourced penalty e_j. Without loss of generality, we assume that the jobs' parameters are integral, unless stated otherwise. Each job J_j is either accepted to be processed on the machine in batches or outsourced with a penalty e_j having to be paid. The machine can process up to b jobs simultaneously as a batch, and the processing time of the batch is equal to the longest time of the job in the batch; in the deterioration model, the deteriorating rate of the batch is equal to the largest deteriorating rate of the job in the batch. Following Gawiejnowicz [7], we denote our problems as $1 \mid p-batch,\ p_j = \alpha_j(A + Dt),\ r_j = 0,\ outsouring,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$ and $1 \mid p-batch,\ p_j = \alpha_j t, r_j,\ outsourcing,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$.

Lemma 1 (see [7]). *Problem $1 \mid p_j = \alpha_j(A + Dt) \mid C_{\max}$ is solvable in $O(n)$ time if $A, D > 0$, $\alpha_j > 0$ for $1 \leq j \leq n$, and the completion time of the jth job and the maximum completion time are $C_j = (A/D)\prod_{i=1}^{j}(1 + D\alpha_i) - A/D$ and $C_{\max} = (A/D)\prod_{j=1}^{n}(1 + D\alpha_j) - A/D$, respectively.*

Lemma 2 (see [15]). *Problem $1 \mid p_j = \alpha_j t,\ r_j = t_0,\ rej \mid C_{\max}(S) + \sum_{\overline{S}} e_j$ is NP-hard.*

Lemma 3 (see [17]). *For the single machine scheduling problem $1 \mid p_j = \alpha_j t \mid C_{\max}$, if a schedule $\pi = \{J_{[1]}, J_{[2]}, \dots, J_{[n]}\}$, and the starting time of job $J_{[1]}$ is t_0, then the makespan is*

$$C_{\max}(\pi) = t_0 \prod_{j=1}^{n}\left(1 + \alpha_{[j]}\right). \tag{1}$$

We list the following useful algorithm stated in Miao et al. [18].

Algorithm FBLDR (fully batching longest deteriorating rate)

Step 1. Reindex jobs in nonincreasing order of their deteriorating rates such that $\alpha_1 \geq \cdots \geq \alpha_n$.

Step 2. Form batches by placing jobs J_{jb+1} through $J_{(j+1)b}$ together in the same batch, for $j = 0, 1, \dots, \lfloor n/b \rfloor$.

Step 3. Schedule the batches in any arbitrary order.

The schedule contains at most $\lfloor n/b \rfloor + 1$ batches and all batches are full except possibly the last one, where $\lfloor n/b \rfloor$ denotes the largest integer less than n/b.

3. The Case with Identical Release Dates

In this section, we discuss problem $1 \mid p-batch,\ p_j = \alpha_j(A + Dt),\ r_j = 0,\ outsouring,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$.

3.1. NP-Hardness. From Lemma 2 and Zhang and Miao [19], we can get the following theorem.

Theorem 4. *Problem $1 \mid p-batch,\ p_j = \alpha_j(A + Dt),\ r_j = 0,\ outsouring,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$ is NP-hard.*

3.2. Pseudo-Polynomial Time Algorithm

Theorem 5. *For problem $1 \mid p-batch,\ p_j = \alpha_j(A+Dt),\ r_j = 0,\ outsouring,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$, there exists an optimal schedule in which the accepted jobs are assigned to the machine by Algorithm FBLDR.*

Assume that the jobs have been indexed so that $\alpha_1 \geq \cdots \geq \alpha_n$.

Let $F_j(b_j, E)$ be the optimal value of the objective function satisfying the following conditions for problem $1 \mid p-batch,\ p_j = \alpha_j(A + Dt),\ r_j = 0,\ outsouring,\ b < n \mid C_{\max}(S) + \sum_{\overline{S}} e_j$:

(i) The jobs in consideration are $J_1, \dots, J_j$.

(ii) The number of processed jobs in the last batch is b_j. If there is no such batch, we set $b_j = 0$.

(iii) The total penalty of outsourced jobs is E.

We design an algorithm as follows.

Algorithm $\mathscr{DP}1$

Step 1 (initialization)

$F_1(1, 0) = A\alpha_1$ and $F_j(b_j, E) = +\infty$ for $(b_j, E) \neq (1, 0)$.

$F_1(0, e_1) = e_1$ and $F_j(b_j, E) = +\infty$ for $(b_j, E) \neq (0, e_1)$.

Step 2 (iteration)

If $b_j = 1$,

$F_j(b_j, E) = \min\{(1 + D\alpha_j)(F_{j-1}(b, E) - E) - A\alpha_j + E, F_{j-1}(b_j, E - e_j) + e_j\}$.

If $b_j > 1$,

$F_j(b_j, E) = \min\{F_{j-1}(b_j - 1, E), F_{j-1}(b_j, E - e_j) + e_j\}$.

Step 3 (solution)

Define the optimal value:

$F^* = \min\{F_n(b_n, E) : 0 \leq b_n \leq b, 0 \leq E \leq \sum_{j=1}^{n} e_j\}$.

In Step 2, when job J_j is accepted and $b_j = 1$, J_j has to start a new batch, combining with Lemma 1, we have $F_j(b_j, E) = (1+D\alpha_j)(F_{j-1}(b, E)-E)-A\alpha_j+E$, when $b_j > 1$, J_j should be assigned to the last batch which has existed, and the makespan does not change; therefore, $F_j(b_j, E) = F_{j-1}(b_j - 1, E)$. When job J_j is outsourced, $F_j(b_j, E) = F_{j-1}(b_j, E - e_j) + e_j$.

Theorem 6. *Algorithm $\mathscr{DP}1$ solves problem $1 \mid p-batch, p_j = \alpha_j(A + Dt), r_j = 0, outsouring, b < n \mid C_{\max}(S) + \sum_{\bar{S}} e_j$ in $O(nb(\sum_{j=1}^{n} e_j))$ time.*

4. The Case with m Distinct Release Dates

In this section, we present a dynamic programming algorithm for the case where the jobs have a constant number of release dates.

From Theorem 4, we have that $1 \mid p-batch, p_j = \alpha_j t, r_j, outsourcing, b < n \mid C_{\max}(S) + \sum_{\bar{S}} e_j$ is NP-hard. Assume that jobs have been indexed so that $\alpha_1 \geq \cdots \geq \alpha_n$ and there are m distinct release dates denoted by $t_0 = R_1 < R_2 < \cdots < R_m$, where m is a constant. We divide $[R_1, +\infty)$ into m time intervals $[R_1, R_2), [R_2, R_3), \ldots, [R_m, R_{m+1})$, where $R_{m+1} = +\infty$.

Given a schedule π, let $J = \bigcup_{i=1}^{m} H_i(\pi)$ and $H_i(\pi) \cap H_j(\pi) = \phi$ for $1 \leq i \neq j \leq m$ such that jobs in $H_i(\pi)$ are started at or after R_i but strictly before R_{i+1}.

We can locally rearrange the schedule of each subset $H_i(\pi)$, without increasing its makespan, so that it follows Algorithm *FBLDR* as the following lemma.

Lemma 7. *For any schedule π for J to minimize $C_{\max}$, there exists a schedule π' for J with $C'_{\max}$, such that $C_{\max}(\pi') \leq C_{\max}(\pi)$, and $H_i(\pi) = H_i(\pi')$, and the schedule for $H_i(\pi')$ according to π' follows Algorithm FBLDR for $i = 1, 2, \ldots, m$.*

Reindex jobs in nonincreasing order of their deteriorating rates so that $\alpha_1 \geq \cdots \geq \alpha_n$. By the above lemma, we partition all jobs into a sequence of m disjoint subsets $H_1, \ldots, H_m$ and can assume that each subset H_i is scheduled in time interval $[R_i, R_{i+1})$ according to algorithm FBLPT. If $H_i \neq \phi$, then there is a starting time $t \geq R_i$ at which the first batch of H_i is started. When the last batch of H_{i-1} is delay, that is, being finished after R_i (even though it is started before time R_i), then $t > R_i$.

Let $F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E)$ be the optimal value of the objective function satisfying the following conditions:

(i) The jobs in consideration are $J_1, \ldots, J_j$.

(ii) $d_i (1 \leq i \leq m)$ is the delay time of the first batch starting in $[R_i, R_{i+1})$ which is caused by the last batch starting in $[R_{i-1}, R_i)$. Without loss of generality, we set $d_1 = 0$.

(iii) $P_i (1 \leq i \leq m)$ is the total length of batches starting in $[R_i, R_{i+1})$. If there is no batch starting in $[R_i, R_{i+1})$, we set $P_i = 0$.

(iv) $b_i (1 \leq i \leq m)$ is the number of jobs in the last batch starting in $[R_i, R_{i+1})$. If there is no batch starting in time $[R_i, R_{i+1})$, then we set $b_i = 0$.

(v) The total penalty of the outsourced jobs is E.

Finally, we define the makespan of a schedule π for jobs $J_1, \ldots, J_j$ as $R_m + d_m + P_m$. Hence, the makespan is always at least R_m.

Now, we distinguish two cases.

Case 1 (job J_j is outsourced). In this case, we have

$$F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = F_{j-1}(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E - e_j) + e_j. \quad (2)$$

Case 2 (job J_j is accepted). In this case, without loss of generality, we assume that $r_j = R_i$; then it can be scheduled in time interval $[R_k, R_{k+1})$ $(i \leq k \leq m)$. We distinguish two subcases in the following.

Case 2.1 ($b_k = 1$). In this subcase, the last batch starting in $[R_k, R_{k+1})$ is either full or does not exist at all before inserting J_j. Without loss of generality, we assume that the batch is full.

For $i \leq k \leq m$, let x be the increasing time by inserting job J_j in $[R_k, R_{k+1})$. Then we have $(R_k + d_k + P_k - x)\alpha_j = x$; then $x = (R_k + d_k + P_k)\alpha_j/(1 + \alpha_j)$. Thus, the total length of batches starting in $[R_k, R_{k+1})$ before inserting J_j is $P_k - x = P_k - (R_k + d_k + P_k)\alpha_j/(1 + \alpha_j)$.

Therefore, for $i \leq k < m$, if $(R_k + d_k + P_k)\alpha_j/(1 + \alpha_j) \in Z^+$ and $P_k \geq (R_k + d_k + P_k)\alpha_j/(1 + \alpha_j)$, we have

$$F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = F_{j-1}\Bigg(d_1, \ldots, d_m; P_1, \ldots, P_{k-1}, \left(P_k - \frac{(R_k + d_k + P_k)\alpha_j}{1 + \alpha_j}\right), P_{k+1}, \ldots, P_m; b_1, \ldots, b_{k-1}, b, b_{k+1}, \ldots, b_m; E\Bigg). \quad (3)$$

If $(R_k + d_k + P_k)\alpha_j/(1 + \alpha_j) \overline{\in} Z^+$ or $P_k < (R_k + d_k + P_k)\alpha_j/(1 + \alpha_j)$, we have $F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = +\infty$.

For $k = m$, $(R_m + d_m + P_m)\alpha_j/(1 + \alpha_j) \in Z^+$ and $P_m \geq (R_m + d_m + P_m)\alpha_j/(1 + \alpha_j)$, we have

$$F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = F_{j-1}\left(d_1, \ldots, d_m; P_1, \ldots, P_{m-1}, P_m - \frac{(R_m + b_m + P_m)\alpha_j}{1 + \alpha_j}; b_1, \ldots, b_{m-1}, b; E\right) + \frac{(R_m + d_m + P_m)\alpha_j}{1 + \alpha_j}. \quad (4)$$

Otherwise, $F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = +\infty$.

Case 2.2 ($b_k > 1$). In this subcase, J_j can be assigned to the last batch starting in $[R_k, R_{k+1})$. Therefore, we have

$$F_j(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_m; E) = F_{j-1}(d_1, \ldots, d_m; P_1, \ldots, P_m; b_1, \ldots, b_{k-1}, (b_k - 1), b_{k+1}, \ldots, b_m; E). \quad (5)$$

Suppose that $r_1 = R_i$ for some i, if it is accepted, it can only be scheduled at or after R_i. Without loss of generality, suppose that it is scheduled at time interval $[R_k, R_{k+1})$, $i \le k \le m$; then, $C_1 = (R_k + d_k)(1 + \alpha_1)$.

Combining the above discussion, we design Algorithm $\mathscr{DP}2$ as follows.

Algorithm $\mathscr{DP}2$

Step 1 (initialization)

$$\begin{gathered}
F_1(0,\dots,0,d_k,0,\dots,0;0,\dots,0,(R_k+d_k)\alpha_1,0,\dots,0;0,\dots,0,1,0,\dots,0;0) = \max\{R_m,(R_k+b_k)(1+\alpha_1)\},\\
F_1(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E) = +\infty\\
\text{for } (d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E) \ne (0,\dots,0,d_k,0,\dots,0;0,\dots,0,(R_k+d_k)\alpha_1,0,\dots,0;0,\dots,0,1,0,\dots,0;0),\\
F_1(0,\dots,0;0,\dots,0,\dots,0;0,\dots,0;e_1) = R_m + e_1,\\
F_1(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E) = +\infty\\
\text{for } (d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E) \ne (0,\dots,0;0,\dots,0,\dots,0;0,\dots,0;e_1).
\end{gathered} \tag{6}$$

Step 2 (iteration)

$$\begin{aligned}
&F_j(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E)\\
&\quad= \min\{F_{j-1}(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E\\
&\quad -e_j)+e_j,\min\{F_k(j)\}\},
\end{aligned} \tag{7}$$

where $F_k(j)$ is defined as follows:

$$\begin{aligned}
F_k(j) &= F_{j-1}(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_{k-1},(b_k-1),\\
&\quad b_{k+1},\dots,b_m;E)
\end{aligned} \tag{8}$$

for $b_k > 1$ and $i \le k \le m$.

$$\begin{aligned}
F_k(j) &= F_{j-1}\Bigg(d_1,\dots,d_m;P_1,\dots,P_{k-1},\\
&\quad \left(P_k - \frac{(R_k+d_k+P_k)\alpha_j}{1+\alpha_j}\right),P_{k+1},\dots,P_m;b_1,\dots,b_{k-1},b,\\
&\quad b_{k+1},\dots,b_m;E\Bigg)
\end{aligned} \tag{9}$$

for $b_k = 1$, $i \le k < m$, $(R_k + d_k + P_k)\alpha_j/(1+\alpha_j) \in Z^+$, and $P_k \ge (R_k + d_k + P_k)\alpha_j/(1+\alpha_j)$.

Otherwise, $F_k(j) = +\infty$,

$$\begin{aligned}
F_m(j) &= F_{j-1}\Bigg(d_1,\dots,d_m;P_1,\dots,P_{m-1},P_m\\
&\quad - \frac{(R_m+b_m+P_m)\alpha_j}{1+\alpha_j};b_1,\dots,b_{m-1},b;E\Bigg)\\
&\quad + \frac{(R_m+d_m+P_m)\alpha_j}{1+\alpha_j},
\end{aligned} \tag{10}$$

for $b_k = 1$, $k = m$, $(R_m + d_m + P_m)\alpha_j/(1+\alpha_j) \in Z^+$ and $P_m \ge (R_m + d_m + P_m)\alpha_j/(1+\alpha_j)$.

Otherwise, $F_m(j) = +\infty$.

Step 3 (optimal value)

Define the optimal value

$$\begin{aligned}
F^* = \min\Bigg\{&F_n(d_1,\dots,d_m;P_1,\dots,P_m;b_1,\dots,b_m;E) : 0\\
&\le d_i \le R_i\alpha_{\max}, 0 \le P_i \le (R_i+d_i)(T-1), 1 \le b_i \le b,\\
&i = 1,\dots,m; 0 \le E \le \sum_{j=1}^{n} e_j\Bigg\},
\end{aligned} \tag{11}$$

where $T = \prod_{j=1}^{n}(1+\alpha_j)$.

Theorem 8. *Algorithm $\mathscr{DP}2$ solves problem $1 \mid p-batch, p_j = \alpha_j t, r_j \in \{R_i : 1 \le i \le m\}$, outsourcing, $b < n \mid C_{\max}(S) + \sum_{J_j\in\bar{S}} e_j$ with the running time of $O(mnR^2(b(1+\alpha_{\max})\alpha_{\max}T)^m)$, where $R = R_1R_2\cdots R_m$.*

Proof. The correctness of the dynamic programming algorithm $\mathscr{DP}2$ is guaranteed by the above discussion. Clearly, we have $1 \le b_i \le b$, $0 \le d_i \le R_i\alpha_{\max}$ for $i = 1,\dots,m$ and $0 \le E \le \sum_{j=1}^{n} e_j$. Then, $0 \le P_i \le (R_i + d_i)(T-1) \le R_i(1+\alpha_{\max})T$. Therefore, there is $O(nR^2(b(1+\alpha_{\max})\alpha_{\max}T)^m)$ set of possible input value. For each set, it takes $O(m)$ time to compute the value of objective. Thus, the running time of algorithm $\mathscr{DP}2$ is $O(mnR^2(b(1+\alpha_{\max})\alpha_{\max}T)^m)$. □

5. Conclusion

In this paper, we presented a pseudo-polynomial time algorithm for $1 \mid p-batch, p_j = \alpha_j(A+Dt), r_j = 0$, *outsouring*, $b < n \mid C_{\max}(S) + \sum_{\bar{S}} e_j$ and a dynamic programming algorithm for the case where jobs have m

distinct release dates of problem $1 \mid p-batch, p_j = \alpha_j t, r_j, outsouring, b < n \mid C_{\max}(S) + \sum_{\bar{S}} e_j$. For future research, it is worth considering other objectives.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the Natural Science Foundation of Shandong Province (nos. ZR2014AM012, ZR2014AM021, and 2014ZRB019TM) and the Research Award Foundation of Qufu Normal University (no. xkj201516).

References

[1] C.-Y. Lee, R. Uzsoy, and L. A. Martin-Vega, "Efficient algorithms for scheduling semiconductor burn-in operations," *Operations Research*, vol. 40, no. 4, pp. 764–775, 1992.

[2] P. Brucker, A. Gladky, H. Hoogeveen et al., "Scheduling a batching machine," *Journal of Scheduling*, vol. 1, no. 1, pp. 31–54, 1998.

[3] C. N. Potts and M. Y. Kovalyov, "Scheduling with batching: a review," *European Journal of Operational Research*, vol. 120, no. 2, pp. 228–249, 2000.

[4] Y. Z. Zhang and Z. G. Cao, "Parallel batch scheduling: a survey," *Advances in Mathematics (China). Shuxue Jinzhan*, vol. 37, no. 4, pp. 392–408, 2008.

[5] J. N. D. Gupta and S. K. Gupta, "Single facility scheduling with nonlinear processing times," *Computers and Industrial Engineering*, vol. 14, no. 4, pp. 387–393, 1988.

[6] S. Browne and U. Yechiali, "Scheduling deteriorating jobs on a single processor," *Operations Research*, vol. 38, no. 3, pp. 495–498, 1990.

[7] S. Gawiejnowicz, *Time-Dependent Scheduling*, Monographs in Theoretical Computer Science. An EATCS Series, Springer, Berlin, Germany, 2008.

[8] M. Ji and T. C. Cheng, "Parallel-machine scheduling of simple linear deteriorating jobs," *Theoretical Computer Science*, vol. 410, no. 38-40, pp. 3761–3768, 2009.

[9] M. Liu, F. Zheng, C. Chu, and J. Zhang, "An FPTAS for uniform machine scheduling to minimize makespan with linear deterioration," *Journal of Combinatorial Optimization*, vol. 23, no. 4, pp. 483–492, 2012.

[10] C. Miao, "Parallel-batch scheduling with two models of deterioration to minimize the makespan," *Abstract and Applied Analysis*, vol. 2014, Article ID 495187, 10 pages, 2014.

[11] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. r. Sgall, and L. Stougie, "Multiprocessor scheduling with rejection," *SIAM Journal on Discrete Mathematics*, vol. 13, no. 1, pp. 64–78, 2000.

[12] Z. Cao and X. Yang, "A PTAS for parallel batch scheduling with rejection and dynamic job arrivals," *Theoretical Computer Science*, vol. 410, no. 27-29, pp. 2732–2745, 2009.

[13] L. Lu, L. Zhang, and J. Yuan, "The unbounded parallel batch machine scheduling with release dates and rejection to minimize makespan," *Theoretical Computer Science*, vol. 396, no. 1-3, pp. 283–289, 2008.

[14] L. Lu, T. C. Cheng, J. Yuan, and L. Zhang, "Bounded single-machine parallel-batch scheduling with release dates and rejection," *Computers & Operations Research*, vol. 36, no. 10, pp. 2748–2751, 2009.

[15] Y. Cheng and S. Sun, "Scheduling linear deteriorating jobs with rejection on a single machine," *European Journal of Operational Research*, vol. 194, no. 1, pp. 18–27, 2009.

[16] C. X. Miao, *Some problems of the batch scheduling and scheduling with availability constraints*, Qufu Normal University, Jining, China, 2011.

[17] G. Mosheiov, "Scheduling jobs under simple linear deterioration," *Computers & Operations Research*, vol. 21, no. 6, pp. 653–659, 1994.

[18] C. Miao, Y. Zhang, and Z. Cao, "Bounded parallel-batch scheduling on single and multi machines for deteriorating jobs," *Information Processing Letters*, vol. 111, no. 16, pp. 798–803, 2011.

[19] Y. Z. Zhang and C. X. Miao, "Copy method and its applications to batching scheduling problems," *Journal of Qufu Normal University*, vol. 30, no. 2, pp. 41–43, 2004 (Chinese).