Hindawi Publishing Corporation
Abstract and Applied Analysis
Volume 2014, Article ID 495187, 10 pages
http://dx.doi.org/10.1155/2014/495187

Research Article

Parallel-Batch Scheduling with Two Models of Deterioration to Minimize the Makespan

Cuixia Miao[1,2]

[1] *School of Mathematical Sciences, Qufu Normal University, Qufu, Shandong 273165, China*
[2] *School of Physics and Engineering, Qufu Normal University, Qufu, Shandong 273165, China*

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

Received 8 April 2014; Accepted 21 June 2014; Published 23 July 2014

Academic Editor: Fabio M. Camilli

We consider the bounded parallel-batch scheduling with two models of deterioration, in which the processing time of the first model is $p_j = a_j + \alpha t$ and of the second model is $p_j = a + \alpha_j t$. The objective is to minimize the makespan. We present $O(n \log n)$ time algorithms for the single-machine problems, respectively. And we propose fully polynomial time approximation schemes to solve the identical-parallel-machine problem and uniform-parallel-machine problem, respectively.

1. Introduction

Consider the following problem of parallel-batch scheduling with linear processing times. There are n independent deteriorating jobs $J = \{J_1, \ldots, J_n\}$ to be processed on a single or m (identical or uniform) parallel batching machines. The actual processing time of the first model is $p_j = a_j + \alpha t$ and of the second model is $p_j = a + \alpha_j t$, where a_j and a are the basic processing times, α and α_j are the deteriorating rates, and t is the starting time of J_j in a given schedule. All jobs are available from time zero onwards. The batching machines can process up to b jobs simultaneously as a batch, and the processing time of the batch B is equal to the longest time of any job in the batch; that is, $P(B) = \max\{p_j \mid J_j \in B\} = \max_{J_j \in B}\{a_j\} + \alpha t$ or $P(B) = \max\{p_j \mid J_j \in B\} = a + \max_{J_j \in B}\{\alpha_j\} t$. Let M_i $(i = 1, \ldots, m)$ and s_i denote the ith machine and its proceeding speed, respectively. The speeds are identical in the identical parallel machines environment, while they are different from each other in the uniform parallel machines environment. Our objective is to minimize the makespan. Following Gawiejnowicz [1], we denote our problems as $1(P_m, Q_m) \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$, and $1(P_m, Q_m) \mid \text{p-batch}, p_j = a + \alpha_j t, b < n \mid C_{\max}$, where "p-batch" means parallel-batch.

The model described above falls into two categories: parallel-batch scheduling and scheduling with deteriorating jobs. The parallel-batch scheduling is motivated by burn-in operations in semiconductor manufacturing. Brucker et al. [2] said that a parallel-batch machine is a machine that can process up to b jobs simultaneously as a batch, and the processing time of the batch is equal to the longest time of any job in the batch. All jobs contained in the same batch start and complete at the same time. Once processing of a batch is initiated, it cannot be interrupted and other jobs cannot be introduced into the batch until processing is completed. For the parallel-batch scheduling, there are two 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, $b \geq n$. This processing system has been extensively studied in the last two decades. The extensive survey of different models and results was provided both by Potts and Kovalyov [3] and Zhang and Cao [4]. Afterwards, Yuan et al. [5] gave some new results for the parallel-batch scheduling.

Traditional scheduling problems all assume that the processing times of jobs are constant. However, the processing times may change in the real world. Examples can be found in steel production and firefighting, where any delay in processing a task may increase its completion time. Scheduling with deteriorating job was first considered by J. N. D. Gupta and S. K. Gupta [6] and Browne and Yechiali [7]. From then on, this scheduling model has been extensively

studied. Cheng et al. [8] gave a survey and the monograph by Gawiejnowicz [1] presented this scheduling from different perspectives and covers results and examples. Ji and Cheng [9, 10] and Liu et al. [11] gave some new results for this scheduling.

The parallel-batch scheduling with deteriorating jobs was initiated by Qi et al. [12]; Li et al. [13] and Miao et al. [14] also considered this scheduling; they gave some results for the minimizing makespan of the parallel-batch scheduling with the simple linear deterioration (i.e., $p_j = \alpha_j t$). In this paper, we consider the parallel-batch scheduling with $p_j = a_j + \alpha t$ and $p_j = a + \alpha_j t$. We not only consider the single-machine problem but also the parallel-machine problems.

The remainder of the paper is organized as follows. In Section 2, we give some preliminaries to be used in this paper. In Section 3, we present an $O(n \log n)$ time algorithm for the single-machine problem $1 \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$, and propose two fully polynomial time approximation schemes for problems $P_m \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$ and $Q_m \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$, respectively. In Section 4, we present an $O(n \log n)$ time algorithm for the single-machine problem $1 \mid \text{p-batch}, p_j = a + \alpha_j t, b < n \mid C_{\max}$, and propose two fully polynomial time approximation schemes for problems $P_m \mid \text{p-batch}, p_j = a + \alpha_j t, b < n \mid C_{\max}$ and $Q_m \mid \text{p-batch}, p_j = a + \alpha_j t, b < n \mid C_{\max}$, respectively. We conclude the paper and suggest some interesting topics for future research in the last section.

2. Preliminaries

In this section, we give some preliminaries to be used in the following sections.

J. N. D. Gupta and S. K. Gupta [6] considered the general model $p_j = a_j + \alpha_j t$, where a_j and α_j denote the basic processing time and deteriorating rate of job J_j, respectively. Lemma 1 presented in [6] is useful for our problems.

Lemma 1 (see [6]). *The problem* $1 \mid p_j = a_j + \alpha_j t \mid C_{\max}$ *is solvable in* $O(n \log n)$ *time by scheduling jobs in the nonincreasing order of* α_j / a_j *ratios.*

It leads to Lemmas 2 and 3.

Lemma 2. *The problem* $1 \mid p_j = a_j + \alpha t \mid C_{\max}$ *is solvable in* $O(n \log n)$ *time by scheduling jobs in the increasing order of their basic processing times, and the makespan is* $C_{\max}(\pi) = \sum_{i=1}^{n} a_{[i]}(1+\alpha)^{n-i}$ *for the given schedule* $\pi = \{J_{[1]}, J_{[2]}, \ldots, J_{[n]}\}$ *with* $a_{[1]} \le a_{[2]} \le \cdots \le a_{[n]}$.

Lemma 3. *The problem* $1 \mid p_j = a + \alpha_j t \mid C_{\max}$ *is solvable in* $O(n \log n)$ *time by scheduling jobs in the nonincreasing order of their deteriorating rates, and the makespan is* $C_{\max}(\pi) = a[1 + \sum_{i=2}^{n} \prod_{l=i}^{n}(1 + \alpha_{[l]})]$ *for the given schedule* $\pi = \{J_{[1]}, J_{[2]}, \ldots, J_{[n]}\}$ *with* $\alpha_{[1]} \ge \alpha_{[2]} \ge \cdots \ge \alpha_{[n]}$.

In the uniform parallel machines environment, let n_i $(i = 1, \ldots, m)$ and $J_{[i,j]}$ denote the number of jobs scheduled on machine M_i and the jth job on this machine. Then $\sum_{i=1}^{m} n_i = n$. Without loss of generality, let $J_{[i,1]}, J_{[i,2]}, \ldots, J_{[i,n_i]}$ be the jobs scheduled on machine M_i. Then, there exists g $(g = 1, \ldots, n)$ such that $J_{[i,j]} = J_g$ with $p_{[i,j]} = a_{[i,j]} + \alpha t = a_g + \alpha t$ or $p_{[i,j]} = a + \alpha_{[i,j]} t = a + \alpha_g t$.

From Lemma 2, we get the following lemma for problem $Q_m \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$.

Lemma 4. *The completion time of job* $J_{[i,j]}$ *is* $C_{[i,j]} = \sum_{h=1}^{j} (1 + (\alpha / s_i))^{j-h} (a_{[i,h]} / s_i)$.

And from Lemma 3, we get the following lemma for problem $Q_m \mid \text{p-batch}, p_j = a + \alpha_j t, b < n \mid C_{\max}$.

Lemma 5. *The completion time of job* $J_{[i,j]}$ *is* $C_{[i,j]} = (a / s_i)[1 + \sum_{i=2}^{j} \prod_{h=i}^{j} (1 + (\alpha_{[i,h]} / s_i))]$.

An algorithm 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 approximation algorithm $\{A_\varepsilon\}$ is a fully polynomial-time approximation scheme (FPTAS) if, for each $\varepsilon > 0$, the algorithm A_ε is a $(1+\varepsilon)$-*approximation* algorithm that is polynomial in the input size and in $1/\varepsilon$. Without loss of generality, we assume that $0 < \varepsilon \le 1$.

3. Model $p_j = a_j + \alpha t$

In this section, we discuss the model $p_j = a_j + \alpha t$. First, we present polynomial time algorithm for problem $1 \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$. Then we propose fully polynomial time approximation schemes for problems $P_m \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$ and $Q_m \mid \text{p-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$, respectively.

3.1. Single-Machine Problem $1 \mid$ *p-batch,* $p_j = a_j + \alpha t, b < n \mid C_{\max}$. In this subsection, we present an $O(n \log n)$ time algorithm for the single-machine problem.

Algorithm $\mathcal{A}$. Consider the following steps.

Step 1. Reindex the jobs in increasing order of their basic processing times such that $a_1 \le a_2 \le \cdots \le a_n$.

Step 2. Let $k = \lceil n/b \rceil$ and $j = n - b(k-1)$.

Step 3. Form batches $B_1, \ldots, B_k$ such that $B_1 = \{J_1, \ldots, J_j\}$, $B_2 = \{J_{j+1}, \ldots, J_{j+b}\}, \ldots,$ $B_k = \{J_{n-b+1}, \ldots, J_n\}$.

Step 4. Schedule these batches in the increasing order of their basic processing times from time zero.

Theorem 6. *The problem* $1 \mid$ *p-batch,* $p_j = a_j + \alpha t, b < n \mid C_{\max}$ *is solvable in* $O(n \log n)$ *time by Algorithm* $\mathcal{A}$.

Proof. Suppose that we reindex the jobs in increasing order of their basic processing times. Now, we only need to prove that there exists an optimal schedule satisfying the following properties. (i) The indices of jobs in each batch are consecutive. (ii) All batches are full except possibly the

one which contains job J_1. (iii) Batches are scheduled in the increasing order of their basic processing times.

We consider an optimal schedule π in the following proof.

To show (i), suppose that there are two batches B_f and B_h and three jobs J_i, J_j, and J_l with $a_i \geq a_j \geq a_l$ such that $J_i, J_l \in B_f$ and $J_j \in B_h$ in schedule π. We obtain a new schedule π' by shifting job J_j with job J_l; that is, $B'_f = B_f \setminus \{J_l\} \cup \{J_j\}$ and $B'_h = B_h \setminus \{J_j\} \cup \{J_l\}$. And then $P(B'_f) = P(B_f)$ and $P(B'_h) \leq P(B_h)$ since $a_i \geq a_j \geq a_l$ and the starting time is not changed. Thus, $C(B'_f) \leq C(B_f)$ and $C(B'_h) \leq C(B_h)$, and the completion times of other jobs do not increase. Thus, π' remains optimal. A finite number of repetitions of this procedure yields an optimal schedule of the required form.

To show (ii), suppose that there is a batch B_x in π such that B_x is not full. We know that the indices of jobs in B_x are consecutive from (i). Without loss of generality, let $B_x = \{J_{n_x}, J_{n_x+1}, \ldots, J_{n_x+e}\}$ with $|B_x| = e + 1 < b$. If we move the remaining consecutive $b - (e + 1)$ jobs $\{J_{n_x+e+1-b}, \ldots, J_{n_x-2}, J_{n_x-1}\}$ from other batches to B_x, this procedure cannot increase the objective value since $a_j \leq a_{n_x}$ for all $j = n_x + e + 1 - b, \ldots, n_x - 1$. A finite number of repetitions of this procedure yields an optimal schedule with that all batches are full except possibly the one which contains the first job J_1.

If we view each batch B as single aggregate job with $P(B) = \max_{j\in B}\{a_j\} + \alpha t$ in schedule π, property (iii) holds from Lemma 2.

This completes the proof of Theorem 6. □

3.2. Problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t$, $b < n \mid C_{\max}$. In this subsection, we assume that m (≥ 2), a_j ($j = 1, \ldots, n$) and α are all integral. We drive some properties of the optimal schedule and propose an FPTAS.

We reindex jobs in increasing order of their basic processing times such that $a_1 \leq \cdots \leq a_n$.

Theorem 7. *For problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid C_{\max}$, there exists an optimal schedule satisfying the following properties.*

(i) *The indices of jobs in each batch on every machine are consecutive.*

(ii) *All batches are full except possibly the one which contains job J_1.*

(iii) *Batches are scheduled in the increasing order of their basic processing times on every machine.*

The proof of this theorem is similar to the proof of Theorem 6.

For problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid C_{\max}$, the properties allow us to determine the batch structure of an optimal solution a priori. So we divide jobs into batches $B_1, B_2, \ldots, B_k$ according to Algorithm $\mathcal{A}$, where $k = \lceil n/b \rceil$. It is possible to view the batch B_j ($j = 1, 2, \ldots, k$) as single aggregate job with processing times $P(B_j) = \max_{J_i\in B_j}\{a_i\} + \alpha t = a_{n+b(j-k)} + \alpha t$.

Similar to the establishment of FPTAS in Kovalyov and Kubiak [15], we introduce the variables x_j, $x_j \in \{1, 2, \ldots, m\}$, $j = 1, 2, \ldots, k$, where $x_j = l$ if batch B_j is scheduled on machine M_l ($l \in \{1, 2, \ldots, m\}$). Let X be the set of all vectors $x = (x_1, x_2, \ldots, x_k)$ with $x_j = l$, $j = 1, 2, \ldots, k$, $l = 1, 2, \ldots, m$. Set

$$F_0^i(x) = 0 \quad i = 1, 2, \ldots, m,$$

$$F_j^l(x) = a_{n+b(j-k)} + (1 + \alpha) F_{j-1}^l(x) \quad \text{for } x_j = l, \tag{1}$$

$$F_j^i(x) = F_{j-1}^i(x) \quad \text{for } x_j = l \; i \neq l.$$

Then problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid C_{\max}$ is reduced to the following problem:

$$\text{Minimize } Q(x) = \max\left\{F_k^i(x) \mid i = 1, 2, \ldots, m\right\} \quad \text{for } x \in X. \tag{2}$$

We introduce the procedure Partition (A, F, ρ) proposed by Kovalyov and Kubiak [15], where $A \subseteq X$, F is a nonnegative integer function on X, and $0 < \rho \leq 1$. This procedure partitions A into disjoint subsets $A_1^F, A_2^F, \ldots, A_{k_F}^F$ such that $|F(x) - F(x')| \leq \rho \min\{F(x), F(x')\}$ for any $x, x' \in A_j^F$, $j = 1, 2, \ldots, k_F$. The following description provides the details of Partition (A, F, ρ).

Procedure Partition (A, F, ρ). Arrange the vectors $x \in A$ in the order $x^{(1)}, x^{(2)}, \ldots, x^{(|A|)}$, where $0 \leq F(x^{(1)}) \leq F(x^{(2)}) \leq \cdots \leq F(x^{(|A|)})$. Assign the vectors $x^{(1)}, x^{(2)}, \ldots, x^{(i_1)}$ to set A_1^F until i_1 is found such that $F(x^{(i_1)}) \leq (1 + \rho)F(x^{(1)})$ and $F(x^{(i_1+1)}) > (1 + \rho)F(x^{(1)})$. If such i_1 does not exist, then take $A_1^F = A$ and stop.

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

Continue the above process until $x^{(|A|)}$ is included in $A_{k_F}^F$ for some k_F.

The main properties of Partition were given by Kovalyov and Kubiak [15] as follows.

Proposition 8. *Consider $|F(x) - F(x')| \leq \rho \min\{F(x), F(x')\}$ for any $x, x' \in A_j^F$, $j = 1, 2, \ldots, k_F$.*

Proposition 9. *Consider $k_F \leq ((\log F(x^{(|A|)}))/\rho) + 2$ for $0 < \rho \leq 1$ and $1 \leq F(x^{(|A|)})$.*

Now, we give a fully polynomial time approximation scheme for problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid C_{\max}$.

Algorithm A_ε^P. Consider the following steps.

Step 1. Reindex the jobs in increasing order of their basic processing times such that $a_1 \leq a_2 \leq \cdots \leq a_n$.

Step 2. Form batches $B_1, \ldots, B_k$ by using Algorithm $\mathcal{A}$, where $k = \lceil n/b \rceil$.

Step 3. Regard batch B_j as an aggregate job with $P(B_j) = a_{n+b(j-k)} + \alpha t$ $(j = 1, \ldots, k)$.

Step 4. Set $Y_0 = \{(\overbrace{0, 0, \ldots, 0}^{k})\}$, $j = 1$, and $F_0^i(x) = 0$ for $i = 1, 2, \ldots, m$.

Step 5. For the set Y_{j-1}, generate the set Y_j' by adding l $(l = 1, 2, \ldots, m)$ in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, without loss of generality, assuming $x_j = l$:

$$\begin{aligned} F_j^l(x) &= a_{n+b(j-k)} + (1+\alpha) F_{j-1}^l(x), \\ F_j^i(x) &= F_{j-1}^i(x) \quad \text{for } i \neq l. \end{aligned} \tag{3}$$

If $j = k$, then set $Y_k = Y_k'$, and go to Step 6.

If $j < k$, then set $\rho = (\varepsilon/2(k+1))$, and perform the following computation.

Call Partition (Y_j', F_j^i, ρ) $(i = 1, \ldots, m)$ to partition the set Y_j' into disjoint subsets $Y_1^{F^i}, Y_2^{F^i}, \ldots, Y_{k_{F^i}}^{F^i}$.

Divide set Y_j' into disjoint subsets $Y_{b_1 \cdots b_m} = Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m}$, $b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$. For each nonempty subset $Y_{b_1 \cdots b_m}$, choose a vector $x^{(b_1 \cdots b_m)}$ such that

$$F_j^l\left(x^{(b_1 \cdots b_m)}\right) = \min\left\{\max_{i=1,\ldots,m} F_j^i(x) \mid x \in Y_{b_1 \cdots b_m}\right\}. \tag{4}$$

Set $Y_j := \{x^{(b_1 \cdots b_m)} \mid b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$, and $Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m} \neq \phi\}$, and $j = j + 1$. Repeat Step 5.

Step 6. Select set $x^0 \in Y_k$ such that $Q(x^0) = \min_{x \in Y_k}\{\max_{i=1,\ldots,m} F_k^i(x)\}$.

Let $T = \log(\max\{k, 1/\varepsilon, 1+\alpha, a_{\max}\})$, where $k = \lceil n/b \rceil$, $a_{\max} = \max_{j=1,\ldots,n}\{a_j\}$.

We get the following theorem.

Theorem 10. *Algorithm A_ε^P finds $x^0 \in X$ for $P_m \mid p\text{-batch}, p_j = a_j + \alpha t, b < n \mid C_{\max}$ such that $Q(x^0) \leq (1+\varepsilon)Q(x^*)$ in $O((\lceil n/b \rceil)^{(2m+1)} T^{m+1}/\varepsilon^m)$, where x^* is an optimal solution.*

Proof. Suppose that $x^*[j] = (x_1^*, \ldots, x_j^*, 0, \ldots, 0) \in Y_{b_1 \cdots b_m} \subseteq Y_j'$ for some j and $b_1, \ldots, b_m$. Algorithm A_ε^P may not choose $x^*[j]$ for further construction; however, for a vector $x^{(b_1 \cdots b_m)}$ chosen instead of it, we have

$$\left|F_j^i(x^*[j]) - F_j^i\left(x^{(b_1 \cdots b_m)}\right)\right| \leq \rho F_j^i(x^*[j]), \quad i = 1, \ldots, m. \tag{5}$$

Set $\rho_1 = \rho$. Consider the vector $x^*[j+1] = (x_1^*, \ldots, x_j^*, x_{j+1}^*, 0, \ldots, 0)$ and $\widetilde{x}^{(b_1 \cdots b_m)} = (x_1^{(b_1 \cdots b_m)}, \ldots, x_j^{(b_1 \cdots b_m)}, x_{j+1}^*, 0, \ldots, 0)$. We assume $x_{j+1}^* = l$. It follows that

$$\begin{aligned} &\left|F_{j+1}^l(x^*[j+1]) - F_{j+1}^l\left(\widetilde{x}^{(b_1 \cdots b_m)}\right)\right| \\ &= \left|\left(a_{n+b(j+1-k)} + (1+\alpha) F_j^l(x^*[j])\right)\right. \\ &\qquad \left. - \left(a_{n+b(j+1-k)} + (1+\alpha) F_j^l\left(x^{(b_1 \cdots b_m)}\right)\right)\right| \\ &= \left|(1+\alpha)\left(F_j^l(x^*[j]) - F_j^l\left(x^{(b_1 \cdots b_m)}\right)\right)\right| \\ &\leq (1+\alpha)\rho F_j^l(x^*[j]) \\ &\leq \rho_1\left(a_{n+b(j+1-k)} + (1+\alpha) F_j^l(x^*[j])\right) \\ &= \rho_1 F_{j+1}^l(x^*[j+1]). \end{aligned} \tag{6}$$

Consequently,

$$F_{j+1}^l\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) \leq (1+\rho_1) F_{j+1}^l(x^*[j+1]). \tag{7}$$

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

$$F_{j+1}^i\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) \leq (1+\rho_1) F_{j+1}^i(x^*[j+1]). \tag{8}$$

Assume that $\widetilde{x}^{(b_1 \cdots a_m)} \in Y_{c_1 \cdots c_m} \subseteq Y_{j+1}'$ and Algorithm A_ε^P chooses $x^{(c_1 \cdots c_m)} \in Y_{c_1 \cdots c_m}$ instead of $\widetilde{x}^{(b_1 \cdots b_m)}$ in the $(j+1)$th iteration; then we have

$$\begin{aligned} &\left|F_{j+1}^i\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) - F_{j+1}^i\left(x^{(c_1 \cdots c_m)}\right)\right| \\ &\leq \rho F_{j+1}^i\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) \\ &\leq \rho(1+\rho_1) F_{j+1}^i(x^*[j+1]), \quad i = 1, \ldots, m. \end{aligned} \tag{9}$$

Then we have

$$\begin{aligned} &\left|F_{j+1}^i(x^*[j+1]) - F_{j+1}^i\left(x^{(c_1 \cdots c_m)}\right)\right| \\ &\leq \left|F_{j+1}^i(x^*[j+1]) - F_{j+1}^i\left(\widetilde{x}^{(b_1 \cdots b_m)}\right)\right| \\ &\quad + \left|F_{j+1}^i\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) - F_{j+1}^i\left(x^{(c_1 \cdots c_m)}\right)\right| \\ &\leq (\rho_1 + \rho(1+\rho_1)) F_{j+1}^i(x^*[j+1]) \\ &= (\rho + \rho_1(1+\rho)) F_{j+1}^i(x^*[j+1]), \quad i = 1, 2, \ldots, m. \end{aligned} \tag{10}$$

Set $\rho_h = \rho + (1+\rho)\rho_{h-1}$, $h = 2, \ldots, k-j$.

Then,

$$\begin{aligned} &\left|F_{j+1}^i(x^*[j+1]) - F_{j+1}^i\left(x^{(b_1 \cdots b_m)}\right)\right| \\ &\qquad \leq \rho_2 F_{j+1}^i(x^*[j+1]), \quad i = 1, \ldots, m. \end{aligned} \tag{11}$$

By repeating the above argument for $j+2, \ldots, k$, we show that $\exists x' \in Y_k$ such that

$$\left|F_k^i(x') - F_k^i(x^*)\right| \leq \rho_{k-j+1} F_k^i(x^*), \quad i = 1, \ldots, m. \tag{12}$$

Since,

$$\begin{aligned}\rho_{k-j+1} &\le \rho\sum_{j=0}^{k}(1+\rho)^{j}\\ &= (1+\rho)^{k+1}-1\\ &= \sum_{j=1}^{k+1}\frac{(k+1)\,k\cdots(k-j+2)}{j!\,(k+1)!}\left(\frac{\varepsilon}{2}\right)^{j}\\ &\le \sum_{j=1}^{k+1}\frac{1}{j!}\frac{\varepsilon}{2}\le\varepsilon.\end{aligned}\tag{13}$$

Then,

$$\left|F_k^i\left(x'\right)-F_k^i\left(x^*\right)\right|\le\varepsilon F_k^i\left(x^*\right),\quad i=1,\ldots,m.\tag{14}$$

Now, we have

$$\left|\max_{i=1,\ldots,m}\left\{F_k^i\left(x'\right)\right\}-\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}\right|\le\varepsilon\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}.\tag{15}$$

From Step 6 in Algorithm A_ε^P, we know that the vector x^0 will be chosen such that

$$\begin{aligned}&\left|\max_{i=1,\ldots,m}\left\{F_k^i\left(x^0\right)\right\}-\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}\right|\\ &\le\left|\max_{i=1,\ldots,m}\left\{F_k^i\left(x'\right)\right\}-\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}\right|\\ &\le\varepsilon\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}.\end{aligned}\tag{16}$$

Then,

$$\max_{i=1,\ldots,m}\left\{F_k^i\left(x^0\right)\right\}\le(1+\varepsilon)\max_{i=1,\ldots,m}\left\{F_k^i\left(x^*\right)\right\}.\tag{17}$$

So,

$$Q\left(x^0\right)\le(1+\varepsilon)\,Q\left(x^*\right).\tag{18}$$

To establish the computation of Algorithm A_ε^P, we know that Step 5 requires the time of $O(|Y_j'|\log|Y_j'|)$ to complete. We have $|Y_{j+1}'|\le 2|Y_j|\le 2k_{F^1}\cdots k_{F^m}$.

By Proposition 9, for $i=1,\ldots,m$,

$$\begin{aligned}k_{F^i}&\le\frac{2\left(\lceil n/b\rceil+1\right)\log\left(\lceil n/b\rceil\,a_{\max}(1+\alpha)^{\lceil n/b\rceil}\right)}{\varepsilon}+2\\ &\le\frac{2(\lceil n/b\rceil+1)^2T}{\varepsilon}+2.\end{aligned}\tag{19}$$

So $|Y_j'|=O((\lceil n/b\rceil)^{2m}T^m/\varepsilon^m)$ and $|Y_j'|\log|Y_j'|=O((\lceil n/b\rceil)^{2m}T^{m+1}/\varepsilon^m)$. Now, we have that the time complexity of Algorithm A_ε^P is $O((\lceil n/b\rceil)^{2m+1}T^{m+1}/\varepsilon^m)$.

This completes the proof of Theorem 10. □

3.3. Problem $Q_m \mid p\text{-batch},\ p_j=a_j+\alpha t,\ b<n \mid C_{\max}$. Motivated by Liu et al. [11], we propose an FPTAS for our uniform-parallel-machine problem $Q_m \mid$ p-batch, $p_j=a_j+\alpha t, b<n \mid C_{\max}$.

We reindex jobs in increasing order of their basic processing times such that $a_1\le\cdots\le a_n$.

We can obtain the following similar theorem to Theorem 7.

Theorem 11. *For problem $Q_m \mid p\text{-batch}, p_j=a_j+\alpha t, b<n \mid C_{\max}$, there exists an optimal schedule satisfying the following properties.*

(i) *The indices of jobs in each batch on every machine are consecutive.*

(ii) *All batches are full except possibly the one which contains job J_1.*

(iii) *Batches are scheduled in the increasing order of their basic processing times on every machine.*

For problem $Q_m \mid$ p-batch, $p_j=a_j+\alpha t, b<n \mid C_{\max}$, these properties allow us to determine the batch structure of an optimal solution a priori. So we divide jobs into batches $B_1,B_2,\ldots,B_k$ according to Algorithm $\mathcal{A}$, where $k=\lceil n/b\rceil$. It is possible to view the batch B_j $(j=1,2,\ldots,k)$ as single aggregate job with processing times $P(B_j)=\max_{i\in B_j}\{a_i\}+\alpha t=a_{n+b(j-k)}+\alpha t$.

We design the FPTAS by using the procedure *Partition* proposed in Kovalyov and Kubiak [15] which requires that the function used must be a nonnegative integer function. Similar to Liu et al. [11], we first modify the objective function to a nonnegative integer function, and this operation does not affect the schedule. For any $i\in\{1,\ldots,m\}$ and $j\in\{1,\ldots,n\}$, define $\Delta_1=\min\{a_j/s_i\}$, $\Delta_2=\min\{1+(\alpha/s_i)\}$. For simplicity, we suppose that Δ_1 and Δ_2 are finite decimals. Arbitrarily find integers $\{l_1,l_2\}\in N^+$ such that $10^{l_1}\Delta_1\in N^+$ and $10^{l_2}\Delta_2\in N^+$. The parameter $10^{l_1+jl_2}C_{[i,j]}$ can be verified as an integer since $C_{[i,j]}=\sum_{h=1}^{j}(1+(\alpha/s_i))^{j-h}(a_{[i,h]}/s_i)$. Define $L=10^{l_1+kl_2}$; the transformed objective function can be expressed as $LC_{\max}$. And we have that $LC_{\max}$ is a nonnegative integer. Meanwhile, the above scale operation only increases the absolute value of the $C_{\max}$ and does not change the schedule. We use the new objective function instead of the original one in the remainder. Now our problem is equivalent to $Q_m \mid$ p-batch, $p_j=a_j+\alpha t, b<n \mid LC_{\max}$.

Similar to the FPTAS in Section 3.2, we introduce the variables $x_j, x_j\in\{1,2,\ldots,m\}$, $j=1,2,\ldots,k$, where $x_j=l$ if batch B_j is scheduled on M_l $(l\in\{1,2,\ldots,m\})$. Let X be the set of all vectors $x=(x_1,x_2,\ldots,x_k)$ with $x_j=l$, $j=1,2,\ldots,k$, $l=1,2,\ldots,m$. Set

$$F_0^i(x)=0\quad i=1,2,\ldots,m,$$

$$F_j^l(x) = \frac{a_{n+b(j-k)}}{s_l}10^{l_1+jl_2} + \left(1+\frac{\alpha}{s_l}\right)F_{j-1}^l(x)\cdot 10^{l_2} \quad \text{for } x_j = l,$$
$$F_j^i(x) = F_{j-1}^i(x)\cdot 10^{l_2} \quad \text{for } x_j = l\ i \neq l, \tag{20}$$

where $F_j^i(x)$ is the magnified workload of machine M_i for the jobs among $J_1, J_2, \ldots, J_j$.

Then problem $Q_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid LC_{\max}$ is reduced to the following problem:

$$\text{Minimize } G(x) = \max\left\{F_k^i(x) \mid i = 1, 2, \ldots, m\right\} \quad \text{for } x \in X. \tag{21}$$

Now, we propose the fully polynomial time approximation scheme.

Algorithm A_ε^Q. Consider the following steps.

Step 1. Reindex the jobs in increasing order of their basic processing times such that $a_1 \le a_2 \le \cdots \le a_n$.

Step 2. Form batches $B_1, \ldots, B_k$ by using Algorithm $\mathscr{A}$, where $k = \lceil n/b \rceil$.

Step 3. Regard batch B_j as an aggregate job with $P(B_j) = a_{n+b(j-k)} + \alpha t$ $(j = 1, \ldots, k)$.

Step 4. Set $Y_0 = \{(\overbrace{0, 0, \ldots, 0}^{k})\}$, $j = 1$ and $F_0^i(x) = 0$ for $i = 1, 2, \ldots m$.

Step 5. For the set Y_{j-1}, generate the set Y_j' by adding l $(l = 1, 2, \ldots, m)$ in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, without loss of generality, assuming $x_j = l$. Set

$$F_0^i(x) = 0 \quad i = 1, 2, \ldots, m,$$
$$F_j^l(x) = \frac{a_{n+b(j-k)}}{s_l}10^{l_1+jl_2} + \left(1+\frac{\alpha}{s_l}\right)F_{j-1}^l(x)\cdot 10^{l_2} \quad \text{for } x_j = l,$$
$$F_j^i(x) = F_{j-1}^i(x)\cdot 10^{l_2} \quad \text{for } x_j = l\ i \neq l. \tag{22}$$

If $j = k$, then set $Y_k = Y_k'$, and go to Step 6.

If $j < k$, then set $\rho = \varepsilon/(2(k+1))$, and perform the following computation.

Call Partition (Y_j', F_j^i, ρ) $(i = 1, \ldots, m)$ to partition the set Y_j' into disjoint subsets $Y_1^{F^i}, Y_2^{F^i}, \ldots, Y_{k_{F^i}}^{F^i}$.

Divide set Y_j' into disjoint subsets $Y_{b_1\cdots b_m} = Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m}$, $b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$. For each nonempty subset $Y_{b_1\cdots b_m}$, choose a vector $x^{(b_1\cdots b_m)}$ such that

$$F_j^l\left(x^{(b_1\cdots b_m)}\right) = \min\left\{\max_{i=1,\ldots,m} F_j^i(x) \mid x \in Y_{b_1\cdots b_m}\right\}. \tag{23}$$

Set $Y_j := \{x^{(b_1\cdots b_m)} \mid b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$, and $Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m} \neq \phi\}$, and $j = j + 1$. Repeat Step 5.

Step 6. Select set $x^0 \in Y_k$ such that $G(x^0) = \min_{x\in Y_k}\{\max_{i=1,\ldots,m} F_k^i(x)\}$.

Let $K = \log(\max\{\lceil n/b \rceil, (1/\varepsilon), A_{\max}, 1 + B_{\max}, 10^{l_1}, 10^{l_2}\})$, where $A_{\max} = \max_{j=1,\ldots,n;i=1,\ldots,m}\{a_j/s_i\}$ and $B_{\max} = \max_{i=1,\ldots,m}\{\alpha/s_i\}$.

We get the following theorem.

Theorem 12. *Algorithm* A_ε^Q *finds* $x^0 \in X$ *for* $Q_m \mid$ *p-batch,* $p_j = a_j + \alpha t, b < n \mid LC_{\max}$ *such that* $G(x^0) \le (1+\varepsilon)G(x^*)$ *in* $O((\lceil n/b \rceil)^{(2m+1)}K^{m+1}/\varepsilon^m)$*, where* x^* *is an optimal solution.*

Analysis. The proof is similar to the proof of Theorem 10 except the following:

$$\begin{aligned}
&\left|F_{j+1}^l(x^*[j+1]) - F_{j+1}^l\left(\tilde{x}^{(b_1\cdots b_m)}\right)\right| \\
&= \left|\frac{a_{n+b(j+1-k)}}{s_l}10^{l_1+jl_2} + \left(1+\frac{\alpha}{s_l}\right)F_j^l(x^*[j])\cdot 10^{l_2}\right. \\
&\qquad \left. - \frac{a_{n+b(j+1-k)}}{s_l}10^{l_1+jl_2} - \left(1+\frac{\alpha}{s_l}\right)F_j^l\left(x^{(b_1\cdots b_m)}\right)\cdot 10^{l_2}\right| \\
&= \left(1+\frac{\alpha}{s_l}\right)\cdot 10^{l_2}\left|F_j^l(x^*[j]) - F_j^l\left(x^{(b_1\cdots b_m)}\right)\right| \\
&\le \left(1+\frac{\alpha}{s_l}\right)\cdot 10^{l_2}\rho F_j^l(x^*[j]) \\
&\le \rho_1\left(\frac{a_{n+b(j+1-k)}}{s_l}10^{l_1+jl_2} + \left(1+\frac{\alpha}{s_l}\right)F_j^l(x^*[j])\,10^{l_2}\right) \\
&= \rho_1 F_{j+1}^l(x^*[j+1]).
\end{aligned} \tag{24}$$

Consequently,

$$F_{j+1}^l\left(\tilde{x}^{(b_1\cdots b_m)}\right) \le (1+\rho_1)F_{j+1}^l(x^*[j+1]). \tag{25}$$

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

$$F_{j+1}^i\left(\tilde{x}^{(b_1\cdots b_m)}\right) \le (1+\rho_1)F_{j+1}^i(x^*[j+1]). \tag{26}$$

To establish the computation of Algorithm A_ε^Q, we know that Step 5 requires the time of $O(|Y_j'|\log|Y_j'|)$ to complete. We have $|Y_{j+1}'| \le 2|Y_j| \le 2k_{F^1}\cdots k_{F^m}$.

By Proposition 9, for $i = 1, \ldots, m$,

$$k_{F^i} \leq \frac{2(\lceil n/b \rceil + 1)\log\left(10^{l_1+\lceil n/b\rceil l_2}\lceil n/b\rceil A_{\max}(1+B_{\max})^{\lceil n/b\rceil}\right)}{\varepsilon} + 2 \leq \frac{2(\lceil n/b\rceil+1)^2 K}{\varepsilon} + 2. \tag{27}$$

So $|Y_j'| = O((\lceil n/b\rceil)^{2m}K^m/\varepsilon^m)$ and $|Y_j'|\log|Y_j'| = O((\lceil n/b\rceil)^{2m}K^{m+1}/\varepsilon^m)$. Now, we have that the time complexity of Algorithm A_ε^Q is $O((\lceil n/b\rceil)^{2m+1}K^{m+1}/\varepsilon^m)$.

4. Model $p_j = a + \alpha_j t$

In this section, we discuss the model $p_j = a + \alpha_j t$. First, we present polynomial time algorithm for problem $1 \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$. Then we propose fully polynomial time approximation schemes for problems $P_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$ and $Q_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$, respectively.

4.1. Single-Machine Problem $1 \mid$ *p-batch,* $p_j = a + \alpha_j t$, $b < n \mid C_{\max}$. In this subsection, we present an $O(n \log n)$ time algorithm for the single-machine problem.

Algorithm FBLDR (fully batching longest deteriorating rate). Consider the following steps

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 $j = 0, 1, \ldots, \lfloor n/b \rfloor$

Step 3. Schedule the batches in the increasing order of their indices.

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 or equal to n/b.

Theorem 13. *Algorithm FBLDR solves problem* $1 \mid$ *p-batch,* $p_j = a + \alpha_j t, b < n \mid C_{\max}$ *optimally, and the optimal objective value is* $C_{\max}^* = a[1 + \sum_{i=2}^{n}\prod_{l=i}^{n}(1+\alpha_{(l-1)b+1})]$.

We omit the proof as it is simple.

4.2. Problem $P_m \mid$ *p-batch,* $p_j = a + \alpha_j t$, $b < n \mid C_{\max}$. In this subsection, we assume that m (≥ 2), α_j $(j = 1, \ldots, n)$ and a are all integral. We drive some properties of the optimal schedule and propose an FPTAS.

We reindex jobs in nonincreasing order of their deteriorating rates such that $\alpha_1 \geq \cdots \geq \alpha_n$.

Theorem 14. *For problem* $P_m \mid$ *p-batch,* $p_j = a + \alpha_j t, b < n \mid C_{\max}$, *there exists an optimal schedule satisfying the following properties.*

(i) *The indices of jobs in each batch on every machine are consecutive.*

(ii) *All batches are full except possibly the one which contains job* J_n.

(iii) *Batches are scheduled in the increasing order of their indices on every machine.*

For problem $P_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$, the properties allow us to determine the batch structure of an optimal solution a priori. So we divide jobs into batches $B_1, B_2, \ldots, B_k$ according to Algorithm FBLDR, where $k = \lceil n/b \rceil$. It is possible to view the batch B_j $(j = 1, 2, \ldots, k)$ as single aggregate job with processing times $P(B_j) = a + \max_{J_i \in B_j}\{\alpha_i\}t = a + \alpha_{(j-1)b+1}t$.

Similar to the establishment of FPTAS in Kovalyov and Kubiak [15], we introduce the variables $x_j, x_j \in \{1, 2, \ldots, m\}$, $j = 1, 2, \ldots, k$, where $x_j = l$ if batch B_j is scheduled on machine M_l $(l \in \{1, 2, \ldots, m\})$. Let X be the set of all vectors $x = (x_1, x_2, \ldots, x_k)$ with $x_j = l$, $j = 1, 2, \ldots, k$, $l = 1, 2, \ldots, m$. Set

$$\begin{aligned} F_0^i(x) &= 0 \quad i = 1, 2, \ldots, m, \\ F_j^l(x) &= a + \left(1 + \alpha_{(j-1)b+1}\right)F_{j-1}^l(x) \quad \text{for } x_j = l, \\ F_j^i(x) &= F_{j-1}^i(x) \quad \text{for } x_j = l \; i \neq l. \end{aligned} \tag{28}$$

Then problem $P_m \mid$ p-batch, $p_j = a_j + \alpha t, b < n \mid C_{\max}$ is reduced to the following problem:

$$\text{Minimize } Q(x) = \max\left\{F_k^i(x) \mid i = 1, 2, \ldots, m\right\} \quad \text{for } x \in X. \tag{29}$$

Using the *Procedure Partition* (A, F, ρ), we design a fully polynomial time approximation scheme for problem $P_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$ as follows.

Algorithm H_ε^P. Consider the following steps.

Step 1. Reindex the jobs in nonincreasing order of their deteriorating rates such that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$.

Step 2. Form batches $B_1, \ldots, B_k$ by using Algorithm FBLDR, where $k = \lceil n/b \rceil$.

Step 3. Regard batch B_j as an aggregate job with $P(B_j) = a + \alpha_{(j-1)b+1}$ $(j = 1, \ldots, k)$.

Step 4. Set $Y_0 = \{(\overbrace{0, 0, \ldots, 0}^{k})\}$, $j = 1$, and $F_0^i(x) = 0$ for $i = 1, 2, \ldots m$.

Step 5. For the set Y_{j-1}, generate the set Y_j' by adding l $(l = 1, 2, \ldots, m)$ in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, without loss of generality, assuming $x_j = l$:

$$\begin{aligned} F_j^l(x) &= a + \left(1 + \alpha_{(j-1)b+1}\right)F_{j-1}^l(x), \\ F_j^i(x) &= F_{j-1}^i(x) \quad \text{for } i \neq l. \end{aligned} \tag{30}$$

If $j = k$, then set $Y_k = Y'_k$, and go to Step 6.

If $j < k$, then set $\rho = (\varepsilon/(2(k+1)))$, and perform the following computation.

Call Partition (Y'_j, F^i_j, ρ) $(i = 1, \ldots, m)$ to partition the set Y'_j into disjoint subsets $Y^{F^i}_1, Y^{F^i}_2, \ldots, Y^{F^i}_{k_{F^i}}$.

Divide set Y'_j into disjoint subsets $Y_{b_1 \cdots b_m} = Y^{F^1}_{b_1} \bigcap \cdots \bigcap Y^{F^m}_{b_m}$, $b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$. For each nonempty subset $Y_{b_1 \cdots b_m}$, choose a vector $x^{(b_1 \cdots b_m)}$ such that

$$F^l_j\left(x^{(b_1 \cdots b_m)}\right) = \min\left\{\max_{i=1,\ldots,m} F^i_j(x) \mid x \in Y_{b_1 \cdots b_m}\right\}. \quad (31)$$

Set $Y_j := \{x^{(b_1 \cdots b_m)} \mid b_1 = 1, \ldots, k_{F^1}; \cdots; b_m = 1, \ldots, k_{F^m}$, and $Y^{F^1}_{b_1} \bigcap \cdots \bigcap Y^{F^m}_{b_m} \neq \phi\}$, and $j = j + 1$. Repeat Step 5.

Step 6. Select set $x^0 \in Y_k$ such that $Q(x^0) = \min_{x \in Y_k}\{\max_{i=1,\ldots,m} F^i_k(x)\}$.

Let $T = \log(\max\{k, (1/\varepsilon), 1 + \alpha_{\max}, a\})$, where $\alpha_{\max} = \max_{j=1,\ldots,n}\{\alpha_j\}$.

We get the following theorem.

Theorem 15. *Algorithm H^P_ε finds $x^0 \in X$ for $P_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$ such that $Q(x^0) \leq (1 + \varepsilon)Q(x^*)$ in $O((\lceil n/b \rceil)^{(2m+1)} T^{m+1}/\varepsilon^m)$, where x^* is an optimal solution.*

Proof. Suppose that $x^*[j] = (x^*_1, \ldots, x^*_j, 0, \ldots, 0) \in Y_{b_1 \cdots b_m} \subseteq Y'_j$ for some j and $b_1, \ldots, b_m$. Algorithm H^P_ε may not choose $x^*[j]$ for further construction; however, for a vector $x^{(b_1 \cdots b_m)}$ chosen instead of it, we have

$$\left|F^i_j(x^*[j]) - F^i_j\left(x^{(b_1 \cdots b_m)}\right)\right| \leq \rho F^i_j(x^*[j]), \quad i = 1, \ldots, m. \quad (32)$$

Set $\rho_1 = \rho$. Consider the vector $x^*[j+1] = (x^*_1, \ldots, x^*_j, x^*_{j+1}, 0, \ldots, 0)$ and $\widetilde{x}^{(b_1 \cdots b_m)} = (x^{(b_1 \cdots b_m)}_1, \ldots, x^{(b_1 \cdots b_m)}_j, x^*_{j+1}, 0, \ldots, 0)$. We assume $x^*_{j+1} = l$. It follows that

$$\begin{aligned}
&\left|F^l_{j+1}(x^*[j+1]) - F^l_{j+1}\left(\widetilde{x}^{(b_1 \cdots b_m)}\right)\right| \\
&= \left|\left(a + \left(1 + \alpha_{jb+1}\right) F^l_j(x^*[j])\right)\right. \\
&\qquad \left. - \left(a + \left(1 + \alpha_{jb+1}\right) F^l_j\left(x^{(b_1 \cdots b_m)}\right)\right)\right| \\
&= \left|\left(1 + \alpha_{jb+1}\right)\left(F^l_j(x^*[j]) - F^l_j\left(x^{(b_1 \cdots b_m)}\right)\right)\right| \quad (33) \\
&\leq \left(1 + \alpha_{jb+1}\right) \rho F^l_j(x^*[j]) \\
&\leq \rho_1\left(a + \left(1 + \alpha_{jb+1}\right) F^l_j(x^*[j])\right) \\
&= \rho_1 F^l_{j+1}(x^*[j+1]).
\end{aligned}$$

Consequently,

$$F^l_{j+1}\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) \leq (1 + \rho_1) F^l_{j+1}(x^*[j+1]). \quad (34)$$

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

$$F^i_{j+1}\left(\widetilde{x}^{(b_1 \cdots b_m)}\right) \leq (1 + \rho_1) F^i_{j+1}(x^*[j+1]). \quad (35)$$

The remained proof of this theorem is similar to that of Theorem 10. □

4.3. Problem $Q_m \mid$ p-batch, $p_j = a + \alpha_j t$, $b < n \mid C_{\max}$. We reindex jobs in nonincreasing order of their deteriorating rates such that $\alpha_1 \geq \cdots \geq \alpha_n$.

Theorem 16. *For problem $Q_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$, there exists an optimal schedule satisfying the following properties.*

(i) *The indices of jobs in each batch on every machine are consecutive.*

(ii) *All batches are full except possibly the one which contains job J_n.*

(iii) *Batches are scheduled in the increasing order of their indices on every machine.*

For problem $Q_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid C_{\max}$, these properties allow us to determine the batch structure of an optimal solution a priori. So we divide jobs into batches $B_1, B_2, \ldots, B_k$ according to Algorithm FBLDR, where $k = \lceil n/b \rceil$. It is possible to view the batch B_j $(j = 1, 2, \ldots, k)$ as single aggregate job with processing times $P(B_j) = a + \alpha_{(j-1)b+1} t$.

Followed that in Section 3.3, for any $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$, define $\Delta_1 = \min\{a/s_i\}$, $\Delta_2 = \min\{1 + (\alpha_j/s_i)\}$. For simplicity, we suppose that Δ_1 and Δ_2 are finite decimals. Arbitrarily find integers $\{l_1, l_2\} \in N^+$ such that $10^{l_1}\Delta_1 \in N^+$ and $10^{l_2}\Delta_2 \in N^+$. From Lemma 5, the parameter $10^{l_1 + jl_2} C_{[i,j]}$ can be verified as an integer since $C_{[i,j]} = (a/s_i)[1 + \sum_{i=2}^{j} \prod_{h=i}^{j}(1 + (\alpha_{[i,h]}/s_i))]$. Define $L = 10^{l_1 + kl_2}$, the transformed objective function can be expressed as $LC_{\max}$. And we have that $LC_{\max}$ is a nonnegative integer. Meanwhile, the above scale operation only increases the absolute value of the $C_{\max}$ and does not change the schedule. We use the new objective function instead of the original one in the remainder. Now our problem is equivalent to $Q_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid LC_{\max}$.

Similar to the FPTAS in Section 3.3, now, we introduce the variables $x_j, x_j \in \{1, 2, \ldots, m\}$, $j = 1, 2, \ldots, k$, where $x_j = l$ if batch B_j is scheduled on M_l $(l \in \{1, 2, \ldots, m\})$. Let X be the set of all vectors $x = (x_1, x_2, \ldots, x_k)$ with $x_j = l$, $j = 1, 2, \ldots, k$, $l = 1, 2, \ldots, m$. Set

$$\begin{aligned}
F^i_0(x) &= 0 \quad i = 1, 2, \ldots, m, \\
F^l_j(x) &= \frac{a}{s_l} 10^{l_1 + jl_2} \\
&\quad + \left(1 + \frac{\alpha_{(j-1)b+1}}{s_l}\right) F^l_{j-1}(x) \cdot 10^{l_2} \quad \text{for } x_j = l, \\
F^i_j(x) &= F^i_{j-1}(x) \cdot 10^{l_2} \quad \text{for } x_j = l \; i \neq l,
\end{aligned} \quad (36)$$

where $F_j^i(x)$ is the magnified workload of machine M_i for the jobs among $J_1, J_2, \ldots, J_j$.

Then problem $Q_m \mid$ p-batch, $p_j = a + \alpha_j t, b < n \mid LC_{\max}$ is reduced to the following problem:

$$\text{Minimize } G(x) = \max\left\{F_k^i(x) \mid i = 1, 2, \ldots, m\right\} \quad \text{for } x \in X. \tag{37}$$

Now, we propose the fully polynomial time approximation scheme.

Algorithm H_ε^Q. Consider the following steps.

Step 1. Reindex the jobs in non-increasing order of their deteriorating rates such that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$.

Step 2. Form batches $B_1, \ldots, B_k$ by using Algorithm FBLDR, where $k = \lceil n/b \rceil$.

Step 3. Regard batch B_j as an aggregate job with $P(B_j) = a + \alpha_{(j-1)b+1} t$ $(j = 1, \ldots, k)$.

Step 4. Set $Y_0 = \{(\overbrace{0, 0, \ldots, 0}^{k})\}$, $j = 1$, and $F_0^i(x) = 0$ for $i = 1, 2, \ldots m$.

Step 5. For the set Y_{j-1}, generate the set Y_j' by adding l $(l = 1, 2, \ldots, m)$ in position j of each vector from Y_{j-1}. Calculate the following for any $x \in Y_j'$, without loss of generality, assuming $x_j = l$. Set

$$\begin{aligned} F_0^i(x) &= 0 \quad i = 1, 2, \ldots, m, \\ F_j^l(x) &= \frac{a}{s_l} 10^{l_1 + j l_2} \\ &\quad + \left(1 + \frac{\alpha_{(j-1)b+1}}{s_l}\right) F_{j-1}^l(x) \cdot 10^{l_2} \quad \text{for } x_j = l, \\ F_j^i(x) &= F_{j-1}^i(x) \cdot 10^{l_2} \quad \text{for } x_j = l \; i \neq l. \end{aligned} \tag{38}$$

If $j = k$, then set $Y_k = Y_k'$, and go to Step 6.

If $j < k$, then set $\rho = (\varepsilon/(2(k+1)))$, and perform the following computation.

Call Partition (Y_j', F_j^i, ρ) $(i = 1, \ldots, m)$ to partition the set Y_j' into disjoint subsets $Y_1^{F^i}, Y_2^{F^i}, \ldots, Y_{k_{F^i}}^{F^i}$.

Divide set Y_j' into disjoint subsets $Y_{b_1 \cdots b_m} = Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m}$, $b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$. For each nonempty subset $Y_{b_1 \cdots b_m}$, choose a vector $x^{(b_1 \cdots b_m)}$ such that

$$F_j^l\left(x^{(b_1 \cdots b_m)}\right) = \min\left\{\max_{i=1,\ldots,m} F_j^i(x) \mid x \in Y_{b_1 \cdots b_m}\right\}. \tag{39}$$

Set $Y_j := \{x^{(b_1 \cdots b_m)} \mid b_1 = 1, \ldots, k_{F^1}; \ldots; b_m = 1, \ldots, k_{F^m}$, and $Y_{b_1}^{F^1} \bigcap \cdots \bigcap Y_{b_m}^{F^m} \neq \phi\}$, and $j = j + 1$. Repeat Step 5.

Step 6. Select set $x^0 \in Y_k$ such that $G(x^0) = \min_{x \in Y_k}\{\max_{i=1,\ldots,m} F_k^i(x)\}$.

Let $K = \log(\max\{\lceil n/b \rceil, (1/\varepsilon), D_{\max}, 1 + E_{\max}, 10^{l_1}, 10^{l_2}\})$, where $D_{\max} = \max_{i=1,\ldots,m}\{a/s_i\}$ and $E_{\max} = \max_{j=1,\ldots,n; i=1,\ldots,m}\{\alpha_j/s_i\}$.

We get the following theorem.

Theorem 17. *Algorithm* H_ε^Q *finds* $x^0 \in X$ *for* $Q_m \mid$ *p-batch,* $p_j = a + \alpha_j t, b < n \mid LC_{\max}$ *such that* $G(x^0) \leq (1 + \varepsilon) G(x^*)$ *in* $O((\lceil n/b \rceil)^{(2m+1)} K^{m+1}/\varepsilon^m)$, *where* x^* *is an optimal solution.*

5. Conclusion

In this paper, we consider the parallel-batch scheduling with $p_j = a_j + \alpha t$ and $p_j = a + \alpha_j t$; the objective is to minimize the makespan. For these two models of deterioration, we present $O(n \log n)$ algorithms for the single-machine problem and propose fully polynomial time approximation schemes to solve the identical-parallel-machine problem and uniform-parallel-machine problem, respectively.

For future research, it is worth considering other objectives.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The author thanks the anonymous reviewers for their helpful and detailed comments on an earlier version of our paper. This work was supported by the National Natural Science Foundation of China (11201259, 61340045), the Doctoral Fund of the Ministry of Education (20123705120001, 20123705110003), the Natural Science Foundation of Shandong Province (ZR2011AL017, SDYC13036), Domestic Visiting Scholar Program for Outstanding Teachers of Higher Education in Shandong Province, Doctoral Research Fund (20110130), and Postdoctoral Researcher of Qufu Normal University.

References

[1] S. Gawiejnowicz, *Time-Dependent Scheduling*, vol. 18 of *Monographs in Theoretical Computer Science. An EATCS Series*, Springer, Berlin, Germany, 2008.

[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*, vol. 37, pp. 392–408, 2008 (Chinese).

[5] J. J. Yuan, S. S. Li, J. Tian, and R. Y. Fu, "A best on-line algorithm for the single machine parallel-batch scheduling with restricted delivery times," *Journal of Combinatorial Optimization*, vol. 17, no. 2, pp. 206–213, 2009.

[6] 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.

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

[8] T. C. E. Cheng, Q. Ding, and B. M. T. Lin, "A concise survey of scheduling with time-dependent processing times," *European Journal of Operational Research*, vol. 152, no. 1, pp. 1–13, 2004.

[9] M. Ji and T. C. E. Cheng, "Parallel-machine scheduling with simple linear deterioration to minimize total completion time," *European Journal of Operational Research*, vol. 188, no. 2, pp. 342–347, 2008.

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

[11] 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.

[12] X. Qi, S. Zhou, and J. Yuan, "Single machine parallel-batch scheduling with deteriorating jobs," *Theoretical Computer Science*, vol. 410, no. 8–10, pp. 830–836, 2009.

[13] S.-S. Li, C. T. Ng, T. C. E. Cheng, and J.-J. Yuan, "Parallel-batch scheduling of deteriorating jobs with release dates to minimize the makespan," *European Journal of Operational Research*, vol. 210, no. 3, pp. 482–488, 2011.

[14] C. X. Miao, Z. Y. Zhang, and Z. G. Cao, "Parallel-batch scheduling of deteriorating jobs with release date to minimize the makespan," *Information Processing Letters*, vol. 111, no. 16, pp. 798–803, 2011.

[15] M. Y. Kovalyov and W. Kubiak, "A fully polynomial approximation scheme for minimizing makespan of deteriorating jobs," *Journal of Heuristics*, vol. 3, no. 4, pp. 287–297, 1998.