Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 3780331, 7 pages
https://doi.org/10.1155/2022/3780331

*Research Article*

# A Coordination Mechanism for Scheduling Game on Parallel-Batch Machines with Deterioration Jobs

**Ganhua Yu**

*School of Management, Qufu Normal University, Rizhao 276826, Shandong, China*

Correspondence should be addressed to Ganhua Yu; 1325715538@qq.com

Received 4 August 2022; Revised 4 December 2022; Accepted 7 December 2022; Published 17 December 2022

Academic Editor: Francisco Chicano

In this study, we consider a parallel-batch machines scheduling game problem with deterioration jobs. The processing time of a job is a simple linear function of its starting time. Each of the parallel-batch machines can process up to $B$ jobs simultaneously as a batch. The processing time of a batch is the processing time of the job with the longest deteriorating rate in the batch. All jobs in the same batch start and complete at the same time. Each job as an agent and its individual cost is the completion time of the job. We present a coordination mechanism for the scheduling game problem with social cost of minimizing the makespan in this paper, namely fully batch longest deteriorating rate. For this problem, we precisely quantify the inefficiency of Nash equilibrium by the logarithm price of anarchy. It is defined to be the ratio between the logarithm of social cost of the worst Nash equilibrium and the logarithm of social cost of an optimum schedule. In addition, we discuss the existence of Nash equilibrium and present an upper bound and lower bounds on the logarithm price of anarchy of the coordination mechanism. We show that the mechanism has a logarithm price of anarchy at most $2 - (1/(3\,max\,\{m, B\})) - (2/3B)$.

## 1. Introduction

For tradition scheduling problems, it is assumed that the processing times of jobs are fixed. However, the processing times may deteriorate due to wear and tear of the machines in the real world. Examples can be found in steel production, fire fighting, and resource allocation, where any delay in processing a job may increase its cost. In this paper, we consider the proportional deterioration model. In this model, the processing time of a job is a simple linear function of its starting time. Consider there are $n$ jobs $J = \{J_1, J_2, \ldots, J_n\}$, where each job $j \in J$ has a deteriorating rate $a_j > 0$. The jobs are indexed in nonincreasing order of their deteriorating rates, i.e., $a_1 \geq a_2 \geq \cdots \geq a_n$. There are $m$ machines $M = \{M_1, M_2, \ldots, M_m\}$, and machines are available at time $t_0$. If $J_j$ starts to be processed on machine $M_i$ at time $t \geq t_0$, its actual processing time is $p_j = a_j t$, where $i = 1, 2, \ldots, m$. Without loss of generality, we assume $t_0 = 1$ throughout this paper. Further studies on deterioration model can be found in other papers, e.g., Gawiejnowicz and Kurc [1], Miao [2], Qian and Han [3], Chai et al. [4].

Research on parallel-batch machine scheduling may date back to Brucker et al. [5]. Each of the parallel-batch machine can process up to $B$ jobs simultaneously as a batch. The processing time of a batch is equal to the time of processing the job with the longest deteriorating rate in the batch. All jobs in the same batch start and complete at the same time. For each $i = 1, 2, \ldots, m$, each machine $M_i$ can process $b_i\,(1 \leq b_i \leq n)$ batches, i.e., $\{B_{i1}, B_{i2}, \ldots, B_{ib_{il}}\}$, where $B_{ik}\,(1 \leq k \leq b_i)$ is the k-th batch processed on machine $M_i$. Obviously, we have $J = \cup_{i=1}^{m} \cup_{k=1}^{b_i} B_{ik}$. More results on parallel-batch machine scheduling can be found in Li and Li [6], Muter [7], Xu et al. [8], and Fowler and Monch [9]. According to the three-field notation $\alpha|\beta|\gamma$ of Graham et al. [10], we denoted our problem as $P_m|B < n, p_j = a_j t|C_{\max}$, where $p_j$, $a_j$, and $C_{\max}$ denote the processing time, the deteriorating rate, and the maximum completion time over all jobs, respectively.

In contrast to traditional scheduling problem in which controlled by a central decision maker, each job now can individually decide which machine to be processed in scheduling game. Each job only is interested in minimizing its own completion time and does not care about the social optimum. We call a schedule Nash equilibrium (NE) if no job can reduce its cost by moving to a different machine [11]. The inefficient of a NE is measured by the price of anarchy (POA) [11], and it is defined as the ratio between the social cost of the worst Nash equilibrium and the social cost of an optimum schedule. Each machine has a scheduling policy to order their jobs in sequence. All the policies of the machines are named as a coordination mechanism [12, 13]. More studies on coordination mechanism for scheduling game problem can be found in Chen and Tan [14], Ye et al. [15], Chen and Xu [16], and He and Tan [17].

For the scheduling game problem with deterioration jobs, POA may not be an appropriate measure, since the exponents of the social cost of the worst Nash equilibrium and the social cost of an optimum schedule are not the same. In order to measure the worst case of the Nash equilibrium schedule, we define another logarithm price of anarchy (LPOA). It is defined as the ratio between the logarithm of social cost of the worst Nash equilibrium and the logarithm of social cost of an optimum schedule, i.e.,

$$\mathrm{LPOA} = \max_{\alpha\in N(I)} \frac{\ln C_{\max}(\alpha)}{\ln C^{*}_{\max}(I)}, \tag{1}$$

where $\alpha$ is a NE assignment, $I$ is the set of all instance of the scheduling game, and $N(I)$ is the set of NE for $I$.

*1.1. Related Work.* There are many researchers who have concentrated on parallel-batch and deterioration load scheduling problems without considering the game environment. Liu et al. [18] considered the scheduling problem $P_m|p_j = a_j t|C_{\max}$ and proved that the worst-case ratio is $(1 + a_{\max})^{1-(1/m)}$, where $a_{\max} = \max_{j=1,\ldots,n} a_j$. Lageweg et al. [19] showed that the scheduling problem $P_m||C_{\max}$ is NP-hard. Lee et al. [20] considered the scheduling problem $P_m|B<n|C_{\max}$ and designed a $(4/3) - (1/3m)$-approximation algorithm. Miao et al. [21] showed that the scheduling problem $P_m|B<n, p_j = a_j t|C_{\max}$ is NP hard and presented an FPTAS algorithm.

In recent years, Chen et al. [22] considered the scheduling game problem $P_m|p_j = a_j t|C_{\max}$ and proved that the POA has a tight bound of $(1 + a_{\max})^{1-(1/m)}$ under SDR police. Under LDR policy, POA is at most $\max_{i=1,2,\ldots m}(1 + a_{[m+i+1]})^{1-(1/i)}$ and at least $(1 + a_{[m+i+1]})^{1-(1/i)} i = 1, 2, \ldots, m$. Under MS policy, they proved that the POS = 1. Fan et al. [23] proved that the POA of FBLPT coordination mechanism for the scheduling game problem $Q_m|B<n|C_{\max}$ is at most $1 + (s_{\max}/\bar{s})(1 - (1/\max\{m, B\})) + \delta$, where $\delta = (1/m\bar{s}C^{*}_{\max})\sum_{i=1}^{m}(i-1)s_i\varepsilon$. In classical scheduling theory, it is assumed that the processing times of jobs are fixed. But the processing times may deteriorate due to wear and tear of the machines in the real world. Parallel-batch processing and job deterioration coexist in many realistic scheduling game situations. Examples can be found in network economic and resource allocation. In this paper, we consider the proportional deterioration scheduling game model $P_m|B<n, p_j = a_j t|C_{\max}$.

*1.2. Our Contribution.* To the best of our knowledge, the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$ has never been discussed. For the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$, (i) we design a coordination mechanism with social cost of minimizing the makespan in this paper, namely fully batch longest deteriorating rate; (ii) we present a logarithm price of anarchy to measure the inefficient of a Nash equilibrium and discuss the existence of Nash equilibrium; (iii) we show that the coordination mechanism has a logarithm price of anarchy at most $2 - (1/3\max\{m, B\}) - (2/3B)$; (iv) if $m \le B$, the coordination mechanism has a logarithm price of anarchy at least $2 - (1/m)$; if $m > B$, the coordination mechanism has a logarithm price of anarchy at least $2 - (m + B - R/mB)$.

## 2. The FBLDR Coordination Mechanism

We first describe the longest deteriorating rate (LDR) rule and fully batch longest deteriorating rate (FBLDR) rule for the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$.

*2.1. LDR Rule*

Step 1: Sort all jobs in $J$ in nonincreasing order of their deteriorating rates and obtain a list of all jobs

Step 2: Assign each job in the list to the machine that completes the job earliest

*2.2. FBLDR Rule*

Step 1: Sort all jobs in $J$ in nonincreasing order of their deteriorating rates and obtain a list of all jobs

Step 2: If there are more than $B$ jobs in the list, place the first $B$ jobs in a batch and iterate. Otherwise, place the remaining jobs in a batch, called an unfilled batch.

In the following, we describe the FBLDR coordination mechanism for the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$.

*2.3. FBLDR Coordination Mechanism.* Scheduling policy of $M_i$:

Group all the jobs that have selected $M_i$ into batches by FBLDR rule. At time $t_0 = 1$, start processing the resulting batches greedily in nonincreasing order of their deteriorating rates.

## 3. Existence of NE for the Scheduling Game Problem $P_m|B<n, p_j = = a_j t|C_{\max}$

In the section, we prove the existence of NE of the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$. We first consider the LDR-Greedy algorithm for the scheduling problem $P_m|B<n, p_j = a_j t|C_{\max}$ and then show that the NE is the schedule generated by the LDR-Greedy algorithm.

For the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$, we obtain a LDR-Greedy algorithm. The idea of the LDR-Greedy algorithm is similar to LPT-

Greedy algorithm [24]. The following description provides the details of LDR-Greedy algorithm.

Sorting the jobs in nonincreasing order of their deteriorating rates, then assigning the jobs one by one into some batch, and processing the resulting batches greedily. If the job $j$ has been scheduled and the job $j+1$ has not been processed, it is called state $j$. If the batch has not been full, then the job $j+1$ will be allocated in the last batch of a machine which completes job $j+1$ earliest; otherwise, the job $j+1$ will be allocated in a new batch of a machine which completes job $j+1$ earliest. Let set $C_i(j)$ and $n_i(j)$ be the completion time on machine $M_i$ in state $j$ and the number of jobs scheduled on machine $M_i$ in state $j$, respectively. Set

$$G_i(j+1) = \begin{cases} C_i(j)\left(1+a_{j+1}\right), \text{if } n_i(j) \bmod B = 0, \\ C_i(j), \text{otherwise.} \end{cases} \tag{2}$$

*3.1. Algorithm LDR-Greedy*

Step 1. For $i = 1, 2, \ldots, m$, set $C_i(0) = 1$, $n_i(0) = 0$, $j = 1$

Step 2. Compute $G_i(j)$ and $i^*(j) = \arg\min_{1\le i\le m}\{G_i(j)\}$. Then, assign job $j$ to the $\lceil n_{i^*(j)}(j-1) + 1/B\rceil$-th batch of $M_{i^*(j)}$

Step 3. If $i = i^*(j)$, set $C_{i^*(j)}(j) = G_{i^*(j)}(j)$, $n_{i^*(j)}(j) = n_{i^*(j)}(j-1)+1$; otherwise, set

$$\begin{aligned} C_i(j) &= G_i(j-1), \\ n_i(j) &= n_i(j-1). \end{aligned} \tag{3}$$

Step 4. If $j < n$, then $j = j+1$. Repeat Step 2.

*Example 1.* To illustrate Algorithm LDR-Greedy, consider the following instance: $n = 5$, $a_1 = 5$, $a_2 = 4, a_3 = 3, a_4 = 2, a_5 = 1, B = 2, m = 2$ Due to FBLDR rule, job 1 and job 2 are placed in a batch, job 3 and job 4 are placed in a batch, and job 5 is placed in a batch. According to Algorithm LDR-Greedy, then we get that job 1 and job 2 are processed on machine 1, and job 3, job 4, and job 5 are processed on machine 2.

$$\begin{aligned} C_1(1) &= C_1(2) = 1\times(1+5) = 6, \\ G_1(1) &= C_1(0)(1+a_1) = 6, \\ G_1(2) &= C_1(1) = 6, \\ C_2(3) &= C_2(4) = 1\times(1+3) = 4, \\ G_2(3) &= C_2(2)(1+a_3) = 4, \\ G_2(4) &= C_2(3) = 4, \\ C_2(5) &= 8, \\ G_2(5) &= C_2(4)(1+a_5) = 8. \end{aligned} \tag{4}$$

Consider the schedule produced by Algorithm LDR-Greedy. It is worth noticing that the subschedule of the jobs processed on machine $M_i (1 \le i \le m)$ is consistent with the one produced by the scheduling policy $M_i$. Similar to Nong et al. [24], we obtain the following theorem.

**Theorem 1.** *Under the FBLDR coordination mechanism, there is a NE for the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$, and the NE is the schedule generated by the LDR-Greedy algorithm.*

*Proof.* We prove the theorem by contradiction. Let $\tau$ be the schedule produced by the LDR-Greedy algorithm. If $\tau$ is not a NE, then assume that $k$ is the smallest index which can complete earlier by switching from its current machine $i^*(k)$ to machine $i'(k)$. According to the FBLDR rule and the LDR rule, each job $k'$ that starts its processing before job $k$ has a deteriorating rate at least $a_k$, and it means that $G_{i'(j)}(k) \ge G_{i^*(j)}(k)$. But the assumption means that $G_{i'(j)}(k) < G_{i^*(j)}(k)$, which is a contradiction to $i^*(j) = \operatorname{argmin}_{1\le i\le m}\{G_i(j)\}$. Thus, $\tau$ is a NE. □

## 4. An Upper Bound on the LPOA

In the section, we present an upper bound on the LPOA of the FBLDR coordination mechanism. The following lemma plays an important role in the LPOA analysis.

**Lemma 1.** *There is an optimal schedule for the problem $P_m|B<n, p_j = a_j t|C_{\max}$ such that the jobs in $J$ are partitioned into batches by the FBLDR rule [21].*

**Theorem 2.** *For the scheduling problem $P_m|B<n, p_j = a_j t|C_{\max}$, the LPOA of the FBLDR coordination mechanism is at most* $2 - (1/3\max\{m,B\}) - (2/3B)$.

*Proof.* We prove the theorem by contradiction. Assume that there are counterexamples with the LPOA strictly larger than $2 - (1/3\max\{m,B\}) - (2/3B)$. Denote the counterexample $I'$ with the smallest number of jobs, and assume $I'$ has $n'$ jobs. Input $I'$ into the LDR-Greedy algorithm and let $\tau_1$ be the schedule returned, whose LPOA is strictly larger than $2 - (1/3\max\{m,B\}) - (2/3B)$. Assume that there are $b_i (1 \le i \le m)$ batches processed on machine $M_i (1 \le i \le m)$. Sort all the $b_i$ batches in nonincreasing order of their starting time on machine $M_i$, and then denote them by $B_{i1}, B_{i2}, \ldots, B_{ib_i}$, respectively.

According to the work of Nong et al. [24], when the FBLPT rule and LPT-Greedy algorithm replaced by the FBLDR rule and LDR-Greedy algorithm, respectively, we can obtain two similar claims with the proof omitted. □

*Claim 1.* There is only one batch completing at $C_{\max}(\tau_1)$, and the batch contains job $n'$ only.

Denote the longest deteriorating rate of jobs in batch $B_{ik}$ by $a_{ik}$. We construct a new instance $\Psi$ by adding $B-1$ jobs, each with a deteriorating rate of $a_{n'}$, to the smallest counterexample $I'$. Assume the set of jobs in $\Psi$ is $\{J_1, J_2, \ldots, J_n\}$, then we have$a_n = a_{n'}$.

*Claim 2.* $\Psi$ is a counterexample, and $\tau$ is a NE of $\Psi$ with LPOA strictly larger than $2 - (1/3\max\{m, B\}) - (2/3B)$.

Next, we analyze $\tau$, denote the completion time of $M_h$ is $C_{\max}$ in $\tau$. Therefore, $C_i \geq (C_{\max}/1 + a_{hb_h})$, for each $i \neq h$. Hence, we get that

$$\begin{aligned}\prod_{i=1}^{m}\prod_{k=1}^{b_i}(1+a_{ik}) &= \prod_{i=1}^{m} C_i \\ &= \prod_{i\neq h}^{m} C_i \times C_h \geq \prod_{i\neq h}^{m}\frac{C_{\max}}{1+a_{hb_h}} \times C_{\max} \\ &= \left(\prod_{i=1}^{m} C_{\max}\right)\left(\frac{1}{1+a_{hb_h}}\right)^{m-1} \\ &= (C_{\max})^m\left(\frac{1}{1+a_{hb_h}}\right)^{m-1}.\end{aligned} \tag{5}$$

We then provide lower bound on $C^*_{\max}$, denote the completion time of machine $M_i$ in an optimal schedule by $C^*_i$. Obviously,

$$\prod_{i=1}^{m} C^*_{\max} \geq \prod_{i=1}^{m} C^*_i \geq \left(\prod_{j=1}^{n}(1+a_j)\right)^{(1/B)}. \tag{6}$$

Hence,

$$C^*_{\max} \geq \left(\prod_{j=1}^{n}(1+a_j)\right)^{(1/mB)}. \tag{7}$$

When removing the job with longest deteriorating rate from each batch in $\tau$, then we obtain that the total processing time of the remaining jobs is

$$\frac{\prod_{j=1}^{n}(1+a_j)}{\prod_{i=1}^{m}\prod_{k=1}^{b_i}(1+a_{ik})}, \tag{8}$$

where $a_{ik}$ is the longest deteriorating rate of jobs in $B_{ik}$, for each $k = 1, 2, \ldots, b_i$.

Consider the resulting batches on machine $M_i$ $(1 \leq i \leq m)$. Note that there are $B-1$ jobs remaining in $B_{ik}$ $(1 \leq k \leq b_i)$ after removing the job with longest deteriorating rate from each batch in $\tau$. Since $B_{ik}$ $(1 \leq k \leq b_i)$ satisfy the FBLDR rule, we can see that for each $k \in \{1, 2, \ldots, b_i - 1\}$, the deteriorating rate of each job in batch $B_{lk}$ is at least $a_{i,k+1}$, and the deteriorating rate of each job in batch $B_{ib_i}$ is at least $a_{hb_h}$ Hence, the total processing time of the remaining jobs on machine $M_i$ is at least

$$\left(\prod_{k=2}^{b_i}(1+a_{ik}) \times (1+a_{hb_h})\right)^{B-1}. \tag{9}$$

Therefore, we deduce that

$$\begin{aligned}\frac{\prod_{j=1}^{n}(1+a_j)}{\prod_{i=1}^{m}\prod_{k=1}^{m}(1+a_{ik})} &\geq \left(\prod_{i=1}^{m}\left(\prod_{k=2}^{b_i}(1+a_{ik}) \times (1+a_{hb_h})\right)\right)^{B-1} \\ &= \left(\prod_{i=1}^{m}\prod_{k=1}^{b_i}(1+a_{ik})\right)^{B-1}\left(\prod_{i=1}^{m}\left(\frac{1+a_{hb_h}}{1+a_{i1}}\right)\right)^{B-1}.\end{aligned} \tag{10}$$

Thus, we get that

$$\prod_{j=1}^{n}(1+a_j) \geq \left(\prod_{i=1}^{m}\prod_{k=1}^{b_i}(1+a_{ik})\right)^{B}\left(\prod_{i=1}^{m}\frac{1+a_{hb_h}}{1+a_{i1}}\right)^{B-1}. \tag{11}$$

Since $C^*_{\max} \geq 1 + a_{i1}$, for $i = 1, 2, \ldots m$, we obtain that

$$\prod_{i=1}^{m}\frac{1}{C^*_{\max}} \leq \prod_{i=1}^{m}\frac{1}{1+a_{i1}}. \tag{12}$$

By inequalities (1)–(4), we deduce that

$$\begin{aligned}(C^*_{\max})^{mB} &\overset{(2)}{\geq} \prod_{j=1}^{n}\left(1+a_j\right)\\ &\overset{(3)}{\geq} \left(\prod_{i=1}^{m}\prod_{k=1}^{b_i}\left(1+a_{ik}\right)\right)^{B}\left(\prod_{i=1}^{m}\frac{1+a_{hb_h}}{1+a_{i1}}\right)^{B-1}\\ &\overset{(1)}{\geq} (C_{\max})^{mB}\left(\frac{1}{1+a_{hb_h}}\right)^{(m-1)B}\left(\prod_{i=1}^{m}\frac{1+a_{hb_h}}{1+a_{i1}}\right)^{B-1}\\ &\overset{(4)}{\geq} (C_{\max})^{mB}\left(\frac{1}{1+a_{hb_h}}\right)^{(m-1)B}\left(1+a_{hb_h}\right)^{m(B-1)}\left(\prod_{i=1}^{m}\frac{1}{C^*_{\max}}\right)^{B-1}\\ &= (C_{\max})^{mB}\left(1+a_{hb_h}\right)^{B-m}(C^*_{\max})^{m(1-B)},\end{aligned} \tag{13}$$

which implies that

$$(C^*_{\max})^{mB} \geq (C_{\max})^{mB}\left(1+a_{hb_h}\right)^{B-m}(C^*_{\max})^{m(1-B)}. \tag{14}$$

Hence, we get that

$$\begin{aligned} mB\ln C^*_{\max} &\geq mB\ln C_{\max} + (B-m)\ln\left(1+a_{hb_h}\right)\\ &\quad + m(1-B)\ln C^*_{\max}.\end{aligned} \tag{15}$$

And thus,

$$\frac{\ln C_{\max}}{\ln C^*_{\max}} \leq 1 + \frac{(m-B)}{mB}\frac{\ln\left(1+a_{hb_h}\right)}{\ln C^*_{\max}} + \frac{m(B-1)}{mB}. \tag{16}$$

In the remainder, we show that the LPOA at most $2 - (1/3\max\{m,B\}) - (2/3B)$. We consider three cases as follows:

Case 1. $B > m$ and $(1+a_{hb_h})^3 \leq C^*_{\max}$.

Then, $3\ln(1+a_{hb_h}) \leq \ln C^*_{\max}$. From inequality (5), we get that

$$\begin{aligned}\frac{\ln C_{\max}}{\ln C^*_{\max}} &\leq 2 - \frac{1}{B}\\ &= 2 - \frac{1}{3B} - \frac{2}{3B}\\ &= 2 - \frac{1}{3\max\{m,B\}} - \frac{2}{3B}.\end{aligned} \tag{17}$$

Case 2. $B \leq m$ and $(1+a_{hb_h})^3 \leq C^*_{\max}$.

Then, $3\ln(1+a_{hb_h}) \leq \ln C^*_{\max}$. From inequality (5), we obtain that

$$\begin{aligned}\frac{\ln C_{\max}}{\ln C^*_{\max}} &\leq 2 + \frac{(m-B)}{3mB} - \frac{1}{B}\\ &= 2 - \frac{1}{3m} - \frac{2}{3B}\\ &= 2 - \frac{1}{3\max\{m,B\}} - \frac{2}{3B}.\end{aligned} \tag{18}$$

Case 3. $(1+a_{hb_h})^3 > C^*_{\max}$.

We consider two cases as follows:

Case 3.1. If $n \leq mB$, since each batch is full in $\tau$, then we obtain that there is at most one batch on each machine. Therefore, $C_{\max} \leq \max_{1\leq i\leq m}\{1+a_{i1}\} \leq C^*_{\max}$ we receive that $(\ln C_{\max}/\ln C^*_{\max}) \leq 1$.

Case 3.2. If $n > mB$, then there is at least one machine which processes two batches in the optimal schedule. Hence, $(1+a_{hb_h})^2 \leq C^*_{\max}$. Accordingly, we get that

$$2\ln\left(1+a_{hb_h}\right) \leq \ln C^*_{\max}. \tag{19}$$

*Claim 3.* $C^*_{\max} \geq (C_{\max}/1+a_{hb_h})$.

According to $n \leq 2mB$ and each batch is full in $\tau$, there are at most $2m$ batches in $\tau$. If $b_h = 2$, we get that $C_{\max} = (1+a_{h1})(1+a_{hb_h})$. Thus, we receive that

$$C^*_{\max} \geq \left(1+a_{h1}\right) = \frac{C_{\max}}{1+a_{hb_h}}. \tag{20}$$

If $b_h > 2$, then there is one machine $M_r$ processes only one batch, and the batch is completed at $1+a_{r1}$. Owing to the fact that the completion time of a machine is at least $(C_{\max}/1+a_{hb_h})$ in $\tau$, we obtain that $1+a_{r1} \geq (C_{\max}/1+a_{hb_h})$. According to $C^*_{\max} \geq 1+a_{r1}$, we get that

$$C^*_{\max} \geq 1+a_{r1} \geq \frac{C_{\max}}{1+a_{hb_h}}. \tag{21}$$

The claim holds. Thus, we receive that

$$\begin{aligned}\frac{\ln C_{\max}}{\ln C^*_{\max}} &\leq \frac{\ln\left(\left(1+a_{hb_h}\right)C^*_{\max}\right)}{\ln C^*_{\max}}\\ &= 1 + \frac{\ln\left(1+a_{hb_h}\right)}{\ln C^*_{\max}} \leq 1 + \frac{1}{2}\\ &= \frac{3}{2}.\end{aligned} \tag{22}$$

The above discussion implies that LPOA is no more than $2 - (1/3 \max\{m, B\}) - (2/3B)$. Hence, $\Psi$ is not a counterexample and causes a contradiction. The proof is completed.

We note that if $B = 1$, then the LPOA $\le (4/3) - (1/3m)$, i.e., the upper bound of the LPOA is exactly the worst case ratio of the LPT-Greedy algorithm.

## 5. Lower Bounds on the LPOA

In the section, we present lower bounds on the LPOA of the FBLDR coordination mechanism for the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$.

**Theorem 3.** *If* $m \le B$, *there is an instance with a NE such that* LPOA $\ge 2 - (1/m)$; *if* $m > B$, *there is an instance with a NE such that* LPOA $\ge 2 - (m + B - R/mB)$.

*Proof.* The proof of the theorem can be divided into the two cases.

Case 1. $m \le B$.

Consider the instance $I_1$. There are $(m-1)dB + B$ jobs, each of $B$ jobs has a deteriorating rate of $d$, and each of $(m-1)dB$ jobs has a deteriorating rate of $x = (1+d)^{(1/d)} - 1$, where $d = mk + 1$ for some positive integer $k$.

Let $\tau$ be an assignment of the instance $I_1$ is as follows: for each $1 \le i \le m-1$, the first batch on $M_i$ consists of a job with a deteriorating rate of $d$ and $B-1$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$. The first batch on $M_m$ consists of $B - m + 1$ jobs each with a deteriorating rate of $d$ and $m-1$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$. For each $1 \le i \le m-1$, $(m-1)kmB$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$ are evenly assigned to the $m$ machines; i.e., there are $(m-1)k$ batches each of which consists of $B$ jobs. It is easily seen that $\tau$ is a NE, and then, we obtain that $C_{\max} = (1+d)(1+x)^{(m-1)k} = (1+d)^{1+((m-1)k/d)}$.

Now, consider an assignment $\tau^*$ of the instance $I_1$ in which a machine processes a batch consisting of $B$ jobs each with a deteriorating rate of $d$ and $(m-1)dB$ jobs and each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$ are evenly assigned to the remaining machines. Hence, we get that $C^*_{\max} = 1 + d$.

Therefore, we deduce that

$$\begin{aligned}
\frac{\ln C_{\max}}{\ln C^*_{\max}} &= \frac{\ln (1+d)^{1+((m-1)k/d)}}{\ln (1+d)} \\
&= \frac{(1+((m-1)k/d))\ln (1+d)}{\ln (1+d)} \\
&= 1 + \frac{(m-1)k}{d} \\
&= 1 + \frac{(m-1)k}{mk+1} \longrightarrow 2 - \frac{1}{m}, (k \longrightarrow \infty).
\end{aligned} \tag{23}$$

Case 2. $m > B$.

Consider the instance $I_2$. There are $m$ jobs each with a deteriorating rate of $d$ and $(m-y)dB + B - R$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$, where $d = mk + 1$ for some positive integer $k$. Set

$$R = \begin{cases} B, if m (\bmod) B = 0, \\ m (\bmod) B, \text{otherwise}. \end{cases} \tag{24}$$

Moreover, set $y = \lceil m/B \rceil$, and we can see that $y = (m + B - R/B)$.

Now, consider assignment $\tau'^*$ of the instance $I_2$ which is as follows: for each $1 \le i \le y-1$, machine $M_i$ processes a batch consists of $B$ jobs each with a deteriorating rate of $d$; machine $M_y$ processes a batch consists of $R$ jobs each with a deteriorating rate of $d$ and $B - R$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$; for each $y + 1 \le i \le m$, machine $M_i$ processes $d$ batches consists of $B$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$. Thus, we receive that $C^*_{\max} = 1 + d$.

Consider an assignment $\tau'$ of the instance $I_2$ as follows: for each $1 \le i \le m$, the first batch on machine $M_i$ consists of one job each with a deteriorating rate of $d$ and $B - 1$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$, and then, $(m-y)kmB$ jobs each with a deteriorating rate of $x = (1+d)^{(1/d)} - 1$ are evenly assigned to the $m$ machines; that is, there are $(m-y)k$ batches each of which consists of $B$ jobs. It is easily seen that $\tau'$ is a NE, and then, we get that $C_{\max} = (1+d)(1+x)^{(m-y)k} = (1+d)^{1+((m-y)k/d)}$.

Therefore, we deduce that

$$\begin{aligned}
\frac{\ln C_{\max}}{\ln C^*_{\max}} &= \frac{\ln (1+d)^{1+((m-y)k/d)}}{\ln (1+d)} \\
&= \frac{(1+((m-y)k/d))\ln (1+d)}{\ln (1+d)} \\
&= 1 + \frac{(m-y)k}{d} \\
&= 1 + \frac{(m-y)k}{mk+1} \longrightarrow 2 - \frac{y}{m}, (k \longrightarrow \infty) \\
&= 2 - \frac{m+B-R}{mB}.
\end{aligned} \tag{25}$$

According to instance $I_1$ and instance $I_2$, we provide the lower bound of the LPOA of the FBLDR coordination mechanism for the scheduling game problem $P_m|B<n, p_j = a_j t|C_{\max}$. We note that instance $I_1$ and instance $I_2$ are similar to Nong et al. [24], which has been used to provide the lower bound of the POA of the FBLPT coordination mechanism for the scheduling game problem $P_m|B<n|C_{\max}$. In addition, we find that the lower bound of the LPOA of the FBLDR coordination mechanism is exactly the lower bound of the POA of the FBLPT coordination mechanism.

In combination with Theorems 2 and 3, we can obtain the following conclusion. □

**Corollary 1.** *When $m = B$, the LPOA of FBLDR coordination mechanism in Theorem 2 is tight.*

## 6. Conclusion

In this paper, we consider the scheduling game problem $P_m|B < n, p_j = a_j t|C_{\max}$. For this scheduling game problem, we design fully batch longest deteriorating rate coordination mechanism and prove the existence of Nash equilibrium. We show that the fully batch longest deteriorating rate coordination mechanism has a logarithm price of anarchy at most $2 - (1/3 \max\{m, B\}) - (2/3B)$. It is an interesting problem to design other coordination mechanisms for the problem.

## Data Availability

The data used to support the findings of this study can be obtained from the corresponding author upon request.

## Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

## Acknowledgments

This research was partially supported by the Science and Technology Research Project of Jiangxi Provincial Education Department under Grant No. GJJ191156 and the National Natural Science Foundation of China under grant no. 11771251.

## References

[1] S. Gawiejnowicz and W. Kurc, "New results for an open time-dependent schedul- ing problem," *Journal of Scheduling*, vol. 23, no. 6, pp. 733–744, 2020.

[2] C. X. Miao, "Complexity of Scheduling with Proportional Deterioration and Release Dates," *Iranian Journal of Science and Technology, Transactions A: Science*, vol. 42, no. 3, pp. 1337–1342, 2018.

[3] J. Qian and H. Y. Han, "The due date assignment scheduling problem with the deteriorating jobs and delivery time," *Journal of applied mathematics and computing*, vol. 68, no. 4, pp. 2173–2186, 2021.

[4] X. Chai, W. H. Li, H. Yuan, and L. Wang, "Online scheduling on a single machine with linear deteriorating processing times and delivery times," *Journal of combinatorial optimization*, vol. 44, no. 3, pp. 1900–1912, 2020.

[5] P. Brucker, A. Gladky, H. Hoogeveen et al., "Scheduling a batching machine," *Journal of scheduling*, vol. 1, pp. 31–54, 1998.

[6] Y. J. LI and S. G. Li, "Scheduling jobs with sizes and delivery times on identical parallel batch machines," *Theoretical computer science*, vol. 841, pp. 1–9, 2020.

[7] İ. Muter, "Exact algorithms to minimize makespan on single and parallel batch processing machines," *European journal of operational research*, vol. 285, no. 2, pp. 470–483, 2020.

[8] J. Xu, J. Q. Wang, and Z. X. Liu, "Parallel batch scheduling: impact of increasing machine capacity," *Omega*, vol. 108, Article ID 102567, 2022.

[9] J. W. Fowler and L. Monch, "A survey of scheduling with parallel batch(p-batch) processing," *European journal of operational research*, vol. 298, no. 1, pp. 1–24, 2022.

[10] R. Graham, E. L. Lawler, J. K. Lenstra, and A. R. Kan, "Optimal and approximation in deterministic sequencing and scheduling: A survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[11] E. Koutsoupias and C. H. Papadimitriou, "Worst-case equilibria," *Computer Scien- ce Review*, vol. 3, no. 2, pp. 65–69, 2009.

[12] G. Christodoulou, E. Koutsoupias, and A. Nanavati, "Coordination mechanisms," *Theoretical Computer Science*, vol. 410, no. 36, pp. 3327–3336, 2009.

[13] K. Lee, J. Y. T. Leung, and M. L. Pinedo, "Coordination mechanisms for parallel machine scheduling," *European Journal of Operational Research*, vol. 220, no. 2, pp. 305–313, 2012.

[14] Q. Q. Chen and Z. Y. Tan, "Mixed coordination mechanisms for scheduling games on hierarchical machines," *International transactions in operational research*, vol. 28, no. 1, pp. 419–437, 2021.

[15] D. S. Ye, L. Chen, and G. C. Zhang, "On the price of anarchy of two-stage machine scheduling games," *Journal of combinatorial optimization*, vol. 42, no. 3, pp. 616–635, 2019.

[16] C. Chen and Y. F. Xu, "Coordination mechanisms for scheduling selfish jobs with favorite machines," *Journal of combinatorial optimization*, vol. 40, no. 2, pp. 333–365, 2020.

[17] C. Y. He and Z. Y. Tan, "Coordination mechanisms for scheduling games with machine modification," *International transactions in operational research*, vol. 28, no. 2, pp. 904–925, 2020.

[18] M. Liu, F. Zheng, S. Wang, and Y. Xu, "Approximation algorithms for parallel machine scheduling with linear deterioration," *Theoretical Computer Science*, vol. 497, pp. 108–111, 2013.

[19] B. J. Lageweg, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy, *Computer-aided Complexity Classification of Deterministic Scheduling Problems*, Stichting Mathematisch Centrum, Amsterdam, Netherlands, 1981.

[20] 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.

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

[22] Q. Q. Chen, L. Lin, Z. Y. Tan, and Y. J. Yan, "Coordination mechanisms for scheduling games with proportional deterioration," *European Journal of Operational Research*, vol. 263, no. 2, pp. 380–389, 2017.

[23] G. Q. Fan and Q. Q. Nong, "A coordination mechanism for a scheduling game with uniform-batching machines," *Asia-Pacific of operational Research*, vol. 5, no. 5, pp. 567–579, 2018.

[24] Q. Q. Nong, G. Q. Fan, and Q. Z. Fang, "A coordination mechanism for a scheduling game with parallel-batching machines," *J Comb Optim*, vol. 33, no. 2, pp. 567–579, 2017.