RAIRO-Oper. Res. 57 (2023) 1267–1284
https://doi.org/10.1051/ro/2023045

RAIRO Operations Research
www.rairo-ro.org

APPROXIMATION ALGORITHMS FOR SCHEDULING SINGLE BATCH MACHINE WITH INCOMPATIBLE DETERIORATING JOBS

Bayi Cheng[1,2,*], Haimei Yuan[1,2], Mi Zhou[1,2] and Tan Qi[3]

Abstract. Motivated by the soaking process under separate heating mode in iron and steel enterprises, we study the parallel batch machine scheduling problem with incompatible deteriorating jobs. The objective is to minimize makespan. A soaking furnace can be seen as a parallel batch processing machine. In order to avoid the thermal stress caused by excessive temperature difference, initial temperature is needed for the ingot before processing. With the increasing of waiting time, the ingot temperature decreases and the soaking time increases. This property is called deterioration. Setup time is needed between incompatible jobs. We show that if jobs have the same sizes, an optimal solution can be found within $O(n \log n)$ time. If jobs have identical processing times, the problem is proved to be NP-hard in the strong sense. We propose an approximate algorithm whose absolute and asymptotic worst-case ratios are less than 2 and 11/9, respectively. When the jobs have arbitrary sizes and arbitrary processing times, the model is also NP-hard in the strong sense. An approximate algorithm with an absolute and asymptotic worst-case ratio less than 2 is proposed. The time complexity is $O(n \log n)$.

Mathematics Subject Classification. 90B35.

Received August 16, 2022. Accepted March 30, 2023.

1. Introduction

Soaking is a typical batch process in iron and steel enterprises, which consumes a large amount of heat, usually accounting for two-thirds of the total energy consumption in the primary rolling zone. Low equipment utilization increases energy loss, therefore, it is important to improve the efficiency of the soaking process. Figure 1 shows the soaking process. A soaking furnace generally includes three soaking pits that can process multiple ingots at the same time. The soaking furnace has separate heating mode and centralized heating mode. Under the separate heating mode, ingots processed simultaneously in the same soaking pit are considered as a batch. When the temperature of the steel ingot reaches the rolling temperature, it will be taken out and supplied to the rolling mill for rolling. In order to reduce energy consumption, the ingot is filled as much as possible on the basis of not exceeding the furnace capacity. To ensure the quality of rolling, the maximum rolling temperature of all ingots in the soaking batch is usually taken as the discharge temperature of this batch. The

Keywords. Batch processing, Optimization, Incompatible, Deterioration, Approximation algorithms.

[1] School of Management, Hefei University of Technology, Hefei 230009, P.R. China.

[2] Key Laboratory of Process Optimization and Intelligent Decision-making, Ministry of Education, Hefei 230009, P.R. China.

[3] School of Electrical Engineering and Automation, Hefei University of Technology, Hefei 230009, P.R. China.

[*] Corresponding author: cheng_bayi@163.com

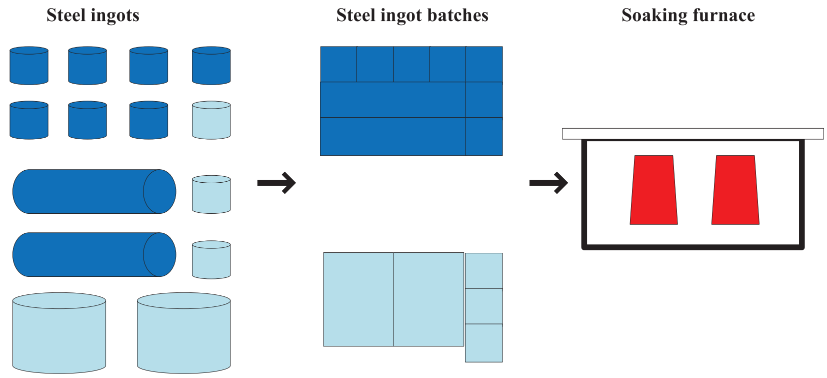

FIGURE 1. Batch processing of ingot soaking.

ingots in the same batch have the same entry and exit times. Therefore, the soaking time of ingots within the same batch is the same, which is equal to the maximum soaking time of ingots in the batch.

In the soaking process with hot chain characteristics, when the initial temperature of the soaking pit is much higher than that of the ingot, thermal stress will be generated due to the great temperature difference between the internal and external temperatures, resulting in surface cracking and internal cracking. Therefore, an initial temperature for the ingot is needed before soaking. The initial temperature is related to the waiting time. The increase in waiting time will reduce the ingot temperature and cause the soaking time to increase. The property that the processing time increases as job waiting time increases is called deterioration. Steel ingots of different materials and types cannot be processed in the same furnace. Such jobs that belong to different families and cannot be processed in the same batch are called incompatible jobs. In addition, setup time is required between incompatible ingots. When the number of changeovers increases, workers get more tired and need more time to prepare for the next changeover. The actual setup time of a changeover therefore varies according to its position in the schedule.

Based on the soaking process under separate heating mode in iron and steel enterprises, this paper studies the parallel batch machine scheduling problem with incompatible deteriorating job families so as to minimize the makespan.

2. Literature review

The first presentation of the deterioration can be traced back to 1988. Gupta and Gupta [1] studied the single machine scheduling problem considering the deterioration effect, where the job processing time is a monotonically increasing function of its starting time. Since then, the related models of time-dependent processing time have been widely studied from various perspectives. Ji *et al.* [2] considered parallel-machine scheduling with deteriorating jobs and proved the total completion time minimization problem is polynomially solvable. Gao *et al.* [3] presented more efficient algorithms to solve the two-agent scheduling problem on a parallel-batch machine, where jobs have release dates and linear deteriorating processing times. Tang *et al.* [4], Yin *et al.* [5] and Zhang *et al.* [6] study the linear deteriorating job scheduling problem under different environments. Liu *et al.* [7] investigate a specialized two-stage hybrid flow shop scheduling problem considering job-dependent deteriorating effect, in which the actual processing time is denoted as $p_{jlr} = p_{jl} r_{a_j}$. Pei *et al.* [8], Li *et al.* [9] and Ding *et al.* [10] studied the sequence dependent deteriorating effects under different models.

In most research, incompatibility mainly occurs in planned production, assembly line balancing and batch scheduling. Dauzère-Pérès and Mōnch [11], Li and Chen [12] studied the number of tardy jobs minimization problem with incompatible job families under different constraints. Geng and Yuan [13] presented an algorithm to solve family jobs scheduling on an unbounded parallel-batching machine. Cheng *et al.* [14] considered the scheduling problem of multiple job families on a batching machine and proposed two polynomial time heuristics. In recent years, more complicated incompatible job family scheduling problems have been studied. Sun *et al.* [15] gave polynomial time algorithms to solve the group scheduling job-dependent due date assignment problem with learning effect and resource allocation. Kramer *et al.* [16] explored the parallel machine scheduling problem with family setup time and introduced five novel mixed integer linear programs to solve it. Li *et al.* [17] investigated the scheduling problem of non-identical jobs from incompatible job families on a batch processing machine, proposed a lower bound and designed heuristics to solve this NP-hard problem. Alizadeh and Kashan [18] explored the scheduling of a single batch processing machine, where jobs are of different sizes and have a conflicting nature with each other. Molaee *et al.* [19] deal with the problem of single machine scheduling with family setup times and random machine breakdown. Mönch and Roob [20] discussed the parallel batch processing machines scheduling problem with incompatible jobs under an arbitrary regular sum objective, where a matheuristic framework is proposed to exploit this insight. Abu-Marrul *et al.* [21] developed an ILS and a GRASP algorithm to solve a batch scheduling problem with identical parallel machines and non-anticipatory family setup time.

Some researchers considered the incompatibility and deterioration simultaneously. Wu and Lee [22] investigated the two single-machine group scheduling problem where the group setup time and the job processing time are both increasing functions of their starting time, and prove that the makespan minimization problem remains polynomially solvable. Lee and Lu [23] considered the single machine scheduling problem with deteriorating jobs and setup time. Xu *et al.* [24] proposed a heuristic algorithm to solve single machine group scheduling problem with deterioration effect. Zhang *et al.* [25] proposed a position-dependent processing time for the single-machine group scheduling problem and presented polynomial-time algorithms to solve it.

However, few studies have been done on batch operation optimization scheduling problems with deterioration and incompatible job families. Optimizing batch machine scheduling problems with effects are more complex than traditional scheduling problems. These problems exist widely in practice. In this paper, the optimization problem of single batch equipment with deterioration and incompatible job families is explored. Specific models are given for different problems, and effective optimization algorithms are provided respectively.

The remainder of this paper is organized as follows. Section 3 describes the meaning of notations and our problem. In Section 4, we propose an optimal algorithm for the first model, where $s_j = 1$. In Section 5, we propose an approximate algorithm for the second model where $p_j = 1$ and calculate the absolute and asymptotic worst-case ratios. In Section 6, we consider the general model, present an approximation algorithm and prove that the absolute and asymptotic worst-case ratios are strictly less than 2. In Section 7, we provide managerial insights for decision makers. Finally, in Section 8, we conclude this paper and give directions for future research.

3. Notations and problem description

The problem under investigation can be described as follows. A set of jobs $J = \{1, 2, \dots, n\}$ needs to be processed, where each job j has a size s_j and a processing time p_j. Jobs are divided into m incompatible families $F = \{F_1, F_2, \dots, F_m\}$, which means that jobs from different families cannot be processed together. Each family contains n_i jobs and $\sum_{i=1}^{m} n_i = n$. The capacity of a single batch processing machine is D, and therefore the total size of jobs in a batch cannot exceed it. Suppose jobs within family F_i are formed into z_i batches, let $B_i = \{B_{i1}, B_{i2}, \dots, B_{iz_i}\}$ denote the batch set of family F_i. Let $p_{\min} = \min\{p_j | j \in J\}$ and $p_{\max} = \max\{p_j | j \in J\}$. We define $\eta = \frac{p_{\min}}{p_{\max}}$, which represents the processing time difference among these jobs. Obviously we have $0 < \eta \leq 1$, where $\eta = 1$ means that jobs have the same processing time and η approaching 0 means there is a big difference between the processing time of jobs.

The normal processing time of batch B_{il} ($i = 1, 2, \dots, m$; $l = 1, 2, \dots, z_i$) is the largest processing time among all the jobs in B_{il}. Let P_{il} denotes the normal processing time of batch B_{il}, so $P_{il} = \max\{p_j | j \in B_{il}\}$. The actual processing time of B_{il} is a linear function of its starting time t. Let P_{il}^A denotes the actual processing time of batch B_{il}. We have

$$P_{il}^A = P_{il} + \alpha t, \quad i = 1, 2, \dots, m, \ l = 1, 2, \dots, z_i, \tag{1}$$

where α is the deteriorating rate of batch processing time and $0 < \alpha < 1$. All the jobs are available at time zero and jobs' preemption is not allowed. A setup time S_i is required if the machine switches to process family F_i. Jobs in the same family are processed consecutively and need no setup time. When the number of changeovers increases, workers get more tired and need more time to prepare for the next changeover. The actual setup time of a changeover therefore varies according to its position in the schedule. Therefore, the actual setup time for changeover to family F_i is sequence-dependent and as follows:

$$S_i^A = S_i r^{\beta}, \quad i = 1, 2, \ldots, m, \; r = 1, 2, \ldots, m, \tag{2}$$

where β is the deteriorating rate of setup time, $0 < \beta < 1$, r is the processing position of job family F_i. The objective is to minimize makespan

$$C_{\max} = \sum_{i=1}^{m} S_i^A + \sum_{i=1}^{m} \sum_{l=1}^{z_i} P_{il}^A. \tag{3}$$

Using the three-field notation in Lai and Lee [26], the models can be denoted respectively as follows.

$\psi_1 : 1|\text{p-batch}, D, \text{incompatible}, P_{il}^A = P_{il} + \alpha t,\; s_i = 1,\; p_j|C_{\max}$.

$\psi_2 : 1|\text{p-batch}, D, \text{incompatible},\; P_{il}^A = P_{il} + \alpha t,\; s_i,\; p_j = 1|C_{\max}$.

$\psi_3 : 1|\text{p-batch}, D, \text{incompatible},\; P_{il}^A = P_{il} + \alpha t,\; s_i,\; p_j|C_{\max}$.

In the above three models, p-batch means the processing time of a batch is equals to the longest processing time of jobs in the batch. One batch facility with capacity of D is used to process jobs and the objective is to minimize makespan $C_{\max}$. In ψ_1, jobs have identical sizes and arbitrary processing times, but in ψ_2, jobs have arbitrary sizes and identical processing times. ψ_3 is the general model.

We introduce the definitions of the absolute worst-case ratio and the asymptotic worst-case ratio. There is a given instance I and an approximation algorithm A, and we denote A_I and OPT_I as the solution obtained by algorithm A and an optimal algorithm, respectively, to solve I. Let $R_{A_I} = \frac{A_I}{\text{OPT}_I}$. So in the algorithm A, we define the absolute worst-case ratio as

$$R_A \equiv inf\{c \geq 1 : R_{A_I} \leq c \text{ for all } I\}$$

and the asymptotic worst-case ratio as

$$R_A^{\infty} \equiv inf\{c \geq 1 : \text{for some } N \geq 0,\; R_{A_I} \leq c \text{ for all } I \text{ with } \text{OPT}_I \geq N\}$$

In the following content, we use π to represent solutions obtained by our algorithms. For simplicity, we use X^* to represent the optimal variables. For example, π^* represents an optimal solution and Z^* represents the number of batches in an optimal solution.

4. Solving problem ψ_1

In this section, we study problem ψ_1, in which all jobs have arbitrary processing times but the same sizes $s_j = 1$. In this case, D jobs can be organized in one batch. We propose the Algorithm A_1 to solve ψ_1.

Algorithm A_1

Step 1. Sort the jobs within each family in non-increasing order of their processing time.

Step 2. Assign the jobs into batches using the following rule. Put the first D jobs in family F_i into the first batch B_{i1}. Put the second D jobs into the second batch B_{i2}, Continue the assignment and obtain z_i batches for each family.

Step 3. For each family F_i, order the batches in non-decreasing order of their processing time and then process the batches consecutively.

Step 4. Sort the families in non-increasing order of their setup time S_i, starting with the family with smallest S_i. □

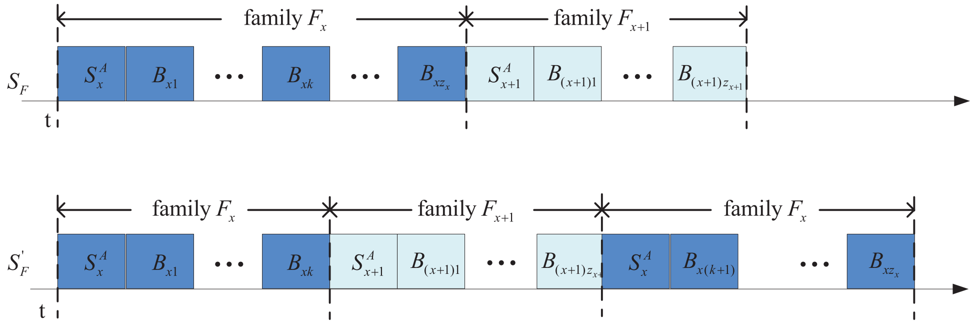

FIGURE 2. Structure of S_F and S'_F.

Proposition 1. *For problem ψ_1, the optimal number of batches for a family is $z_i^* = \lceil n_i/D \rceil$, where $\lceil a \rceil$ represents the smallest integer greater than or equal to a.*

Proposition 1 is easy to obtain and thus the proof is omitted.

Lemma 1. *Step 3 of A_1 can generate the optimal order of batches for each family.*

Proof. By contradiction. First, we prove that batches within a family should be processed consecutively. Consider a schedule S_F where batches of a family are processed consecutively. In schedule S_F, there are two adjacent families, F_x and F_{x+1}. F_{x+1} is processed after F_x. We assume that the completion time of the previous batch before family F_x is t. We use t_{il} to donate the processing start time of batch B_{il}. In this case, the completion time of family F_{x+1} is

$$\begin{aligned} C &= t + S_x^A + \sum_{l=1}^{z_x} P_{xl}^A + S_{x+1}^A + \sum_{l=1}^{z_{x+1}} P_{(x+1)l}^A \\ &= t + S_x x^\beta + \sum_{l=1}^{z_x} (P_{xl} + \alpha t_{xl}) + S_{x+1}(x+1)^\beta + \sum_{l=1}^{z_{x+1}} \big(P_{(x+1)l} + \alpha t_{(x+1)l}\big). \end{aligned} \tag{4}$$

Without loss of generality, we process batches of family F_{x+1} consecutively after the batch B_{xk}, where $k < z_x$. Other batches remain the same as schedule S_F. Then, we obtain a new schedule S'_F. Suppose schedule S'_F is better than schedule S_F. Under schedule S'_F, the completion time of the family F_x is

$$\begin{aligned} C' &= t + S_x^A + \sum_{l=1}^{k} P_{xl}^A + S_{x+1}^A + \sum_{l=1}^{z_{x+1}} P_{(x+1)l}^A + S_x^A + \sum_{l=k+1}^{z_x} P_{xl}^A \\ &= t + S_x x^\beta + \sum_{l=1}^{k} (P_{xl} + \alpha t_{xl}) + S_{x+1}(x+1)^\beta + \sum_{l=1}^{z_{x+1}} (P_{(x+1)l} + \alpha t_{(x+1)l}) \\ &\quad + S_x(x+2)^\beta + \sum_{l=k+1}^{z_x} (P_{xl} + \alpha t'_{xl}). \end{aligned} \tag{5}$$

Thus, we have

$$C' - C = S_x(x+2)^\beta + \alpha \sum_{l=k+1}^{z_x} (t'_{xl} - t_{xl}). \tag{6}$$

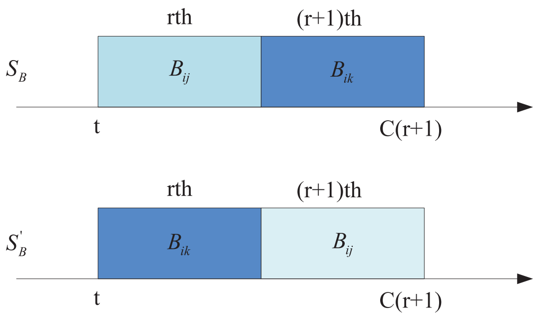

FIGURE 3. Structure of S_B and S'_B.

In schedule S'_F, the batches $\{B_{x(k+1)}, \ldots, B_{xz_x}\}$ have longer processing waiting time. So, we have t'_{xl} larger than t_{xl}. Therefore, $C' - C > 0$. which means that after the interruption, the total processing time became longer. This contradicts with that S'_F betters than S_F, which proves that batches of a family should be processed consecutively to minimize the makespan.

Now we prove that it is optimal to arrange the batches of every family in a non-decreasing order. For an arbitrary family F_i, consider an optimal schedule S_B where batches $S_B = (B_{i1}, B_{i2}, \ldots, B_{iz_i})$ are arranged by non-increasing order of their processing time. In this schedule, there must be two adjacent batches B_{ij} and B_{ik} scheduled in the rth and $(r+1)$th positions, respectively. Such that $P_{ij} \geq P_{ik}$. Furthermore, we assume that the starting time for the rth batch in schedule S_B is t. We now interchange the sequence of B_{ij} and B_{ik}, leaving the remaining batches in their original position. Thus, we form a new schedule S'_B. Let $C(r+1)$ and $C'(r+1)$ denote the completion time of $(r+1)$th batch under S_B and S'_B respectively. Figure 3 shows the structure of S_B and S'_B. Under S_B, we have

$$\begin{aligned} C(r+1) &= t + P_{ij}^A + P_{ik}^A \\ &= t + (P_{ij} + \alpha t) + [P_{ik} + \alpha(t + P_{ij} + \alpha t)] \\ &= (1 + 2\alpha + \alpha^2)t + (1+\alpha)P_{ij} + P_{ik}, \end{aligned} \tag{7}$$

whereas under S'_B, we can obtain

$$\begin{aligned} C'(r+1) &= t + P_{ik}^A + P_{ij}^A \\ &= t + (P_{ik} + \alpha t) + [P_{ij} + \alpha(t + P_{ik} + \alpha t)] \\ &= (1 + 2\alpha + \alpha^2)t + (1+\alpha)P_{ik} + P_{ij}. \end{aligned} \tag{8}$$

Thus, we have

$$C'(r+1) - C(r+1) = \alpha(P_{ik} - P_{ij}) \leq 0 \tag{9}$$

since $P_{ik} \leq P_{ij}$. It implies that the $(r+2)th$ batch under S_B has a later starting time than the same batch under S'_B, which means that the processing time of F_i under S_B is longer than that under S'_B. This contradicts the optimality of S_B and proves that batches should be ordered according to Step 3 of Algorithm A_1 rule. □

Lemma 2. *Step 4 of A_1 can generate the optimal order of families.*

Proof. By contradiction. Let $\pi = (F_1, \ldots, F_p, F_q, \ldots, F_m)$ denote the optimal schedule that families process in non-decreasing order of their setup time. F_p and F_q denote the family scheduled in the rth and $(r+1)$th position respectively, $S_p \leq S_q$. Let $\pi' = (F_1, , \ldots, F_q, F_p, , \ldots, F_m)$, where F_p and F_q are in an opposite order and the remaining families in their original positions. Let TS^* and TS' denote the total setup time consumption of π and π'. Then, under π' we have

$$TS' = \sum_{i=1}^{m} S_i^A = \sum_{i=1}^{r-1} S_i^A + S_q r^\beta + S_p(r+1)^\beta + \sum_{i=r+2}^{m} S_i^A, \tag{10}$$

and under π we have

$$TS^* = \sum_{i=1}^{m} S_i^{*A} = \sum_{i=1}^{r-1} S_i^{*A} + S_p r^\beta + S_q(r+1)^\beta + \sum_{i=r+2}^{m} S_i^{*A}. \tag{11}$$

Thus, we can obtain

$$\begin{aligned} TS^* - TS &= (S_p - S_q)r^\beta + (S_q - S_p)(r+1)^\beta \\ &= (S_q - S_p)[(r+1)^\beta - r^\beta] \\ &\geq 0. \end{aligned} \tag{12}$$

The total setup time consumption of optimal schedule π is greater than or equal to that of π' , which disproves the Lemma 2. □

Theorem 1. *Algorithm A_1 finds an optimal solution for problem ψ_1 in $O(n \log n)$ time.*

Proof. In Algorithm A_1, we use the following rule to generate batches. Assigned j_1 to B_{i1}. Since $s_j = 1$ and the machine capacity is D, we can assign D jobs into the first batch. Therefore, $P_{i1} = P_{i1}^* = p_1$. Assign the next D jobs to the second batch B_{i2}, we have $P_{i2} = P_{i2}^* = p_2$. Repeat the above operation until all jobs of F_i are allocated, we can obtain that $P_{il} = P_{il}^*$. Lemmas 1 and 2 prove an optimal batch processing sequence and an optimal family sequence, respectively. Therefore, we prove Algorithm A_1 finds an optimal solution for ψ_1. Step 1, Step 3 and Step 4 of Algorithm A_1 cost $O(n \log n)$ time, and Step 2 costs $O(n)$ time. Thus,the overall running time of Algorithm A_1 is $O(n \log n)$. □

5. Solving problem ψ_2

Proposition 2. *Problem ψ_2 is NP-hard in the strong sense.*

Proof. We first consider a relaxed problem, in which the setup time is not considered, and the job has an arbitrary size but identical processing time, that is, $p_j = 1$. Additionally, deterioration is not considered. In this case, the problem is equivalent to the Bin Packing Problem (BPP). Since BPP is known to be a NP-hard problem in the strong sense, ψ_2 is NP-hard in the strong sense. □

Now we propose an approximation Algorithm A_2 to solve it.

Algorithm A_2

Step 1. Sorting jobs for each family in a non-increasing order of job sizes.
Step 2. Assign jobs into batches by the First Fit Decreasing rule and obtain z_i batches for family F_i ($i = 1, 2, \ldots, m$).
Step 3. Sorting families in a non-increasing order of their setup time S_i. Then process every family of jobs in consecutive batches. □

Proposition 3. $z_i \leq \frac{11}{9} z_i^* + \frac{6}{9}$ *(Dósa et al. [27]).*

We can find two integers m and i to make $z_i^* = 9m + i$, where $1 \leq i \leq 9$ and $m \geq 0$. We consider the worst case of z_i, that is, $z_i = \left\lfloor \frac{11}{9} z_i^* + \frac{6}{9} \right\rfloor$. The values of z_i^* and the worst case of z_i are shown in Table 1.

TABLE 1. z_i and the worst case of z_i^*.

z_i^*	$9m+1$	$9m+2$	$9m+3$	$9m+4$	$9m+5$	$9m+6$	$9m+7$	$9m+8$	$9m+9$
$\frac{11}{9}z_i^*+\frac{6}{9}$	$11m+\frac{17}{9}$	$11m+\frac{28}{9}$	$11m+\frac{39}{9}$	$11m+\frac{50}{9}$	$11m+\frac{61}{9}$	$11m+\frac{72}{9}$	$11m+\frac{83}{9}$	$11m+\frac{94}{9}$	$11m+\frac{105}{9}$
z_i	$11m+1$	$11m+3$	$11m+4$	$11m+5$	$11m+6$	$11m+8$	$11m+9$	$11m+10$	$11m+11$

Lemma 3. *For* ψ_2, $R_{A_2} \leq 2$, $R_{A_2}^{\infty} = 11/9$.

Proof. From Lemma 2, we have proven that Step 3 generates the optimal family sequence. Now we consider the total processing time for each family.

Case 1: $z_i^* \leq 3$.
when $z_i^* = 1$, we have

$$\sum_{l=1}^{z_i} P_{il}^A = \sum_{l=1}^{z_i^*} P_{il}^{*A} = 1. \tag{13}$$

When $z_i^* = 2$, from Table 1, the worst case of z_i is $z_i = 3$, we have

$$\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} = \frac{3+2\alpha+\alpha^2}{2+\alpha} < 2. \tag{14}$$

When $z_i^* = 3$, from Table 1, the worst case of the z_i is $z_i = 4$, we have

$$\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} = \frac{4+3\alpha+2\alpha^2+\alpha^3}{3+2\alpha+\alpha^2} < 1.67. \tag{15}$$

Case 2: $z_i^* \geq 4$.

$$\begin{aligned}
\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &= \frac{\sum_{l=1}^{z_i}(1-\alpha^{z_i-l+1})P_{il}}{\sum_{l=1}^{z_i^*}(1-\alpha^{z_i^*-l+1})P_{il}^*} \\
&= \frac{z_i - \frac{\alpha}{1-\alpha}(1-\alpha^{z_i})}{z_i^* - \frac{\alpha}{1-\alpha}(1-\alpha^{z_i^*})} \\
&= 2 - \frac{(2z_i^*-z_i) + \frac{\alpha}{1-\alpha}(1-\alpha^{z_i}-2+2\alpha^{z_i^*})}{z_i^* - \frac{\alpha}{1-\alpha}(1-\alpha^{z_i^*})} \\
&\leq 2 - \frac{(2z_i^*-z_i) + \frac{\alpha}{1-\alpha}(\alpha^{z_i^*}-1)}{z_i^* - \frac{\alpha}{1-\alpha}(1-\alpha^{z_i^*})} \\
&= 2 - \frac{(1-\alpha)(2z_i^*-z_i) - \alpha(1-\alpha^{z_i^*})}{(1-\alpha)z_i^* - \alpha(1-\alpha^{z_i^*})} \\
&\leq 2 - \frac{\frac{7}{9}z_i^*(1-\alpha) - \frac{6}{9}(1-\alpha) - \alpha(1-\alpha^{z_i^*})}{(1-\alpha)z_i^* - \alpha(1-\alpha^{z_i^*})} \\
&\leq \frac{11}{9} - \frac{\frac{6}{9}(1-\alpha) + \frac{2}{9}\alpha(1-\alpha^{z_i^*})}{(1-\alpha)z_i^*} \\
&< \frac{11}{9}.
\end{aligned} \tag{16}$$

Hence, the asymptotic worst-case ratio of Algorithm A_2 is

$$\begin{aligned} R_{A_2} &= \frac{C_{\max}}{C^*_{\max}} \\ &= \frac{\sum_{i=1}^m S_i^A + \sum_{i=1}^m \sum_{l=1}^{z_i} P_{il}^A}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} \\ &< \frac{\sum_{i=1}^m S_i^{*A} + 2\sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} \\ &< 2. \end{aligned} \tag{17}$$

We now examine the asymptotic worst-case ratio. Since n approaches infinity, the number of jobs within F_i approaches infinity, we have

$$\begin{aligned} R_{A_2}^{\infty} &= \lim_{n\to\infty} \frac{C_{\max}}{C^*_{\max}} \\ &= \lim_{n\to\infty} \frac{\sum_{i=1}^m S_i^A + \sum_{i=1}^m \sum_{l=1}^{z_i} P_{il}^A}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} \\ &< \frac{\sum_{i=1}^m S_i^{*A} + \frac{11}{9}\sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} \\ &< \frac{11}{9}. \end{aligned} \tag{18}$$

Theorem 2 follows. □

Theorem 2. *The running time of Algorithm A_2 is $O(n \log n)$ time. The absolute worst-case ratio $R_{A_2} < 2$, and the asymptotic worst-case ratio $R_{A_2}^{\infty} < 11/9$.*

6. Solving problem ψ_3

In this section, we consider the general case ψ_3 where the jobs have arbitrary sizes and processing times. Since ψ_3 is more difficult than ψ_2, ψ_3 is also NP-hard in the strong sense. We have the following proposition.

Proposition 4. *Problem ψ_3 is NP-hard in the strong sense.*

Algorithm A_3

Step 1. Sorting jobs for each family in a non-increasing order of their processing times.
Step 2. Assign the jobs into batches by the First Fit Decreasing rule and obtain z_i batches for each family.
Step 3. For each family F_i, order the batches in non-decreasing order of their processing time and then process the batches consecutively.
Step 4. Sorting families in non-increasing order of their setup time S_i and starting with the smallest S_i.

After the execution of Step 1 and Step 2 of Algorithm A_3, batches are sorted in non-increasing order of their processing times. For simplicity, we denote batches and their processing time as H_{il} and K_{il}, respectively. By contrast, in Step 3, batches are ordered in the reverse order and batches and processing times are denoted as B_{il} and P_{il}, respectively.

Lemma 4. *The processing time of batches in F_i satisfies $P_{i(2l-1)} \le P_{i(2l)} \le P^*_{il}$.*

Proof. Consider a family F_i which contains a set of jobs $J_i \in \{1, 2, \ldots, n_i\}$. Assume batches in optimal schedule order in non-increasing sequence, that is, $K^*_{i1} \geq K^*_{i2} \geq \cdots \geq K^*_{iz_i}$. Consider an arbitrary job f in this family, which satisfies

$$\sum_{j=1}^{f-1} s_j \leq (l-1)D \tag{19}$$

and

$$\sum_{j=1}^{f} s_j > (l-1)D. \tag{20}$$

Then, in the optimal solution, the jobs in $\{1, 2, \ldots, \mathrm{f}\}$ cannot all be assigned to the first $l-1$ batches. If job f is assigned to H^*_{il}, we have $K^*_{il} = p_f$. If job f is assigned to a batch later than H^*_{il}, then since the batches are in non-increasing order of their processing time, we have $K^*_{il} \geq p_f$. In both cases, we can conclude that

$$K^*_{il} \geq p_f. \tag{21}$$

Now, we consider problem ψ_3. The worst case occurs when only one job can be put in each batch. In this case, job f is assigned to H_{if}. The batching result is the same as the case when each job has the same size s_0, where $D/2 < s_0 \leq D$. We have $(f-1)D/2 < (f-1)s_0 \leq (l-1)D$. So $f < 2l-1$, which indicates that job f can be assigned to a batch before $B_{i(2l-1)}$. Since the batches are in non-increasing order of their processing times, we have

$$K_{i(2l-1)} \leq p_f. \tag{22}$$

By (21) and (22), we can obtain

$$K_{i(2l)} \leq K_{i(2l-1)} \leq K^*_{il}. \tag{23}$$

In Step 3 of Algorithm A_3, the batches are assigned in non-decreasing order of their processing times, and now we use B_{il} and P_{il} to denote the batches and their processing times respectively. Obviously, we have

$$P_{i(2l-1)} \leq P_{i(2l)} \leq P^*_{il}. \tag{24}$$

□

Lemma 5. *For ψ_3, $R_{A_3} \leq 2$, $R^\infty_{A_2} = 2$.*

Proof. From Lemma 2, we have proven Step 3 generates the optimal family sequence. Now we consider the ratio of total processing time for family F_i.

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P^A_{il}}{\sum_{l=1}^{z_i^*} P^{*A}_{il}} &= \frac{\sum_{l=1}^{z_i} (1-\alpha^{z_i-l+1})P_{il}}{\sum_{l=1}^{z_i^*} (1-\alpha^{z_i^*-l+1})P^*_{il}} \\ &= 2 - \frac{2\sum_{l=1}^{z_i^*} (1-\alpha^{z_i^*-l+1})P^*_{il} - \sum_{l=1}^{z_i} (1-\alpha^{z_i-l+1})P_{il}}{\sum_{l=1}^{z_i^*} (1-\alpha^{z_i^*-l+1})P^*_{il}}.\end{aligned} \tag{25}$$

Case 1: $z_i^* \leq 3$.

If $z_i^* \leq 3$, $z_i - z_i^* - l + 1 \leq 0 (l = 1, 2, \ldots, \frac{z_i}{2})$, we have $\alpha^{z_i^*-l+1} - \alpha^{z_i-2l+2} \geq 0$.

When $z_i^* = 1$, we have

$$\sum_{l=1}^{z_i} P^A_{il} = \sum_{l=1}^{z_i^*} P^{*A}_{il} = P^A_{i1} = \max\{p_j | j \in F_i\}. \tag{26}$$

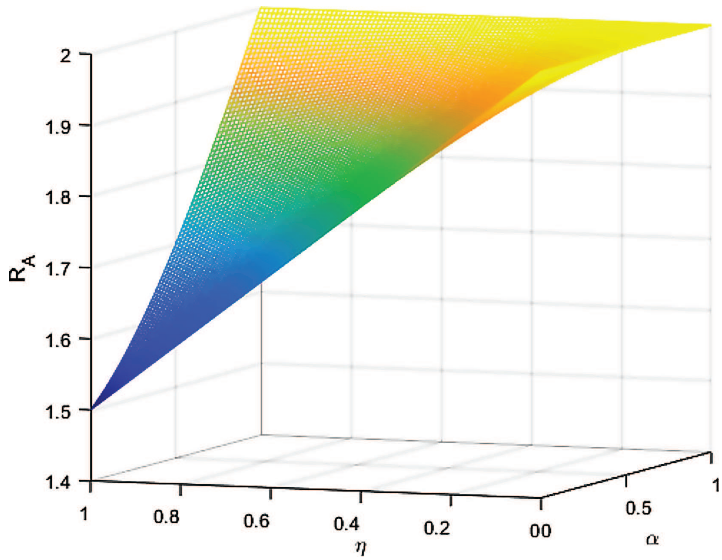

FIGURE 4. The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when $z_i = 2$.

When $z_i^* = 2$, consider the worst case that $z_i = 3$,

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &= \frac{(1+\alpha+\alpha^2)P_{i1} + (1+\alpha)P_{i2} + P_{i3}}{(1+\alpha)P_{i1}^* + P_{i2}^*} \\ &\le \frac{(2+2\alpha+\alpha^2)P_{i1}^* + P_{i2}^*}{(1+\alpha)P_{i1}^* + P_{i2}^*} \\ &= 2 - \frac{P_{i2}^* - \alpha^2 P_{i1}^*}{(1+\alpha)P_{i1}^* + P_{i2}^*} \\ &\le 2 - \frac{(1-\alpha^2)\eta}{2+\alpha}.\end{aligned} \tag{27}$$

When $z_i^* = 3$, consider the worst case that $z_i = 4$, we have

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &= 2 - \frac{2\sum_{l=1}^{3}(1-\alpha^{4-l})P_{il}^* - 2\sum_{l=1}^{4}(1-\alpha^{5-l})P_{il}}{\sum_{l=1}^{3}(1-\alpha^{4-l})P_{i1}^*} \\ &\le 2 - \frac{(\alpha^4-\alpha^3)P_{i1}^* + (\alpha-\alpha^2)P_{i2}^* + 2(1-\alpha)P_{i3}^*}{(1-\alpha^3)P_{i1}^* + (1-\alpha^2)P_{i2}^* + (1-\alpha)P_{i3}^*} \\ &= 2 - \frac{-\alpha^3 P_{i1}^* + (\alpha+\alpha^2)P_{i2}^* + 2(1-\alpha)P_{i3}^*}{(1+\alpha+\alpha^2)P_{i1}^* + (1+\alpha)P_{i2}^* + P_{i3}^*} \\ &\le 2 - \frac{(2+\alpha+\alpha^2-\alpha^3)\eta}{3+2\alpha+\alpha^2}.\end{aligned} \tag{28}$$

The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when $z_i^* = 2$ and $z_i^* = 3$ are shown in Figures 4 and 5, respectively.

Case 2: $z_i^* \ge 4$.

If $z_i^* \ge 4$, there exit k, when $l \ge k, z_i^* - k + 1 \ge z_i - 2k + 2,\ \ k = z_i - z_i^*$.

Case 2.1: When z_i belongs to even.

By Lemma 4, we have $P_{i(2l-1)} \le P_{i(2l)} \le P^*_{il}$, therefore,

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &\le 2 - \frac{2\sum_{l=z_i/2+1}^{z_i^*}\left(1-\alpha^{z_i^*-l+1}\right)P^*_{il} + 2\sum_{l=1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right)P^*_{il}}{\sum_{l=1}^{z_i^*} P^*_{il}} \\ &= 2 - \frac{2\sum_{l=z_i/2+1}^{z_i^*}\left(1-\alpha^{z_i^*-l+1}\right)P^*_{il}}{\sum_{l=1}^{z_i^*} P^*_{il}} + \frac{2\sum_{l=1}^{z_i-z_i^*}\left(\alpha^{z_i^*-l+1}-\alpha^{z_i-2l+2}\right)P^*_{il}}{\sum_{l=1}^{z_i^*} P^*_{il}} \\ &\quad - \frac{2\sum_{l=z_i-z_i^*+1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right)P^*_{il}}{\sum_{l=1}^{z_i^*} P^*_{il}} \\ &\le 2 - \frac{2(1-\alpha)\sum_{l=z_i/2+1}^{z_i^*} P^*_{il}}{\sum_{l=1}^{z_i^*} P^*_{il}} + \frac{2P^*_{i(z_i-z_i^*)}\sum_{l=1}^{z_i-z_i^*}\left(\alpha^{z_i^*-l+1}-\alpha^{z_i-2l+2}\right)}{\sum_{l=1}^{z_i^*} P^*_{il}} \\ &\quad - \frac{2P^*_{i(z_i-z_i^*+1)}\sum_{l=z_i-z_i^*+1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right)}{\sum_{l=1}^{z_i^*} P^*_{il}}.\end{aligned} \tag{29}$$

Since $(z_i - z_i^*) - \left(z_i^* - \frac{z_i}{2}\right) \le \frac{3}{2}\left(\frac{11}{9}z_i^* + \frac{6}{9}\right) - 2z_i^* = -\frac{3}{18}z_i^* + 1 < 0$, we have

$$\begin{aligned}&\frac{\sum_{l=1}^{z_i-z_i^*}\left(\alpha^{z_i^*-l+1}-\alpha^{z_i-2l+2}\right)}{\sum_{l=z_i-z_i^*+1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right)} \\ &= \frac{\sum_{l=1}^{z_i-z_i^*}\alpha^{z_i^*-l+1}\left(1-\alpha^{z_i-z_i^*-l+1}\right)}{\sum_{l=z_i-z_i^*+1}^{z_i/2}\alpha^{z_i-2l+2}\left(1-\alpha^{z_i^*-z_i+l-1}\right)} \\ &\le \frac{\alpha^{2z_i^*-z_i+1}\sum_{l=1}^{z_i-z_i^*}\left(1-\alpha^{z_i-z_i^*-l+1}\right)}{\alpha^{2z_i^*-z_i}\sum_{l=z_i-z_i^*+1}^{z_i/2}\left(1-\alpha^{z_i^*-z_i+l-1}\right)} \\ &= \frac{\alpha[(1-\alpha^{z_i-z_i^*})+\cdots+(1-\alpha^2)+(1-\alpha)]}{[(1-\alpha^{z_i^*-z_i/2-1})+\cdots+(1-\alpha^2)+(1-\alpha)]} \\ &= \frac{\alpha}{1+\left[\sum_{l=z_i-z_i^*+1}^{z_i^*-z_i/2-1}(1-\alpha^l)\Big/\sum_{l=1}^{z_i-z_i^*}\left(1-\alpha^{z_i-z_i^*-l+1}\right)\right]} \\ &< 1,\end{aligned} \tag{30}$$

therefore,

$$\sum_{l=1}^{z_i-z_i^*}\left(\alpha^{z_i^*-l+1}-\alpha^{z_i-2l+2}\right) < \sum_{l=z_i-z_i^*+1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right). \tag{31}$$

According to Step 3 of A_3, batches are in non-decreasing order of their processing time, thus,

$$P^*_{i(z_i-z_i^*)} \le P^*_{i\left(z_i-z_i^*+1\right)}. \tag{32}$$

By (31) and (32), we can conclude that

$$\begin{aligned}&\frac{2P^*_{i\left(z_i-z_i^*\right)}\sum_{l=1}^{z_i-z_i^*}\left(\alpha^{z_i^*-l+1}-\alpha^{z_i-2l+2}\right)}{\sum_{l=1}^{z_i^*} P^*_{il}} \\ &\quad - \frac{2P^*_{i(z_i-z_i^*+1)}\sum_{l=z_i-z_i^*+1}^{z_i/2}\left(\alpha^{z_i-2l+2}-\alpha^{z_i^*-l+1}\right)}{\sum_{l=1}^{z_i^*} P^*_{il}} \le 0.\end{aligned} \tag{33}$$

Therefore, we have

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &\le 2 - \frac{2(1-\alpha)\sum_{l=z_i/2+1}^{z_i^*} P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &= 2 - \frac{2(1-\alpha)}{1 + \sum_{l=1}^{z_i/2} P_{il}^* \Big/ \sum_{l=z_i/2+1}^{z_i^*} P_{il}^*} \\ &\le 2 - \frac{2(1-\alpha)}{1 + \left[P_{i\max}^*\left(\frac{z_i}{2}\right)\right] / \left[P_{i\min}^*\left(z_i^* - \frac{z_i}{2}\right)\right]} \\ &\le 2 - \frac{2(1-\alpha)\left(\frac{7}{9}z_i^* - \frac{6}{9}\right)\eta}{z_i^*\eta + \frac{11}{9}z_i^* + \frac{6}{9}} \\ &= 2 - \frac{2(1-\alpha)\left(\frac{7}{9} - \frac{6}{9z_i^*}\right)\eta}{\eta + \frac{11}{9} + \frac{6}{9z_i^*}} \\ &\le 2 - \frac{22(1-\alpha)\eta}{18\eta + 25}.\end{aligned} \tag{34}$$

Case 2.2: When z_i belongs to odd.

$$\begin{aligned}\frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} &\le 2 - \frac{2\sum_{l=1}^{z_i^*}\left(1-\alpha^{z_i^*-l+1}\right)P_{il}^* - 2\sum_{l=1}^{(z_i+1)/2}(1-\alpha^{z_i-2l+2})P_{il}^* + (1-\alpha)P_{i\frac{z_i+1}{2}}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &\le 2 - \frac{2\sum_{l=(z_i+1)/2+1}^{z_i^*}\left(1-\alpha^{z_i^*-l+1}\right)P_{il}^* + 2\sum_{l=1}^{(z_i+1)/2}\left(\alpha^{z_i-2l+2} - \alpha^{z_i^*-l+1}\right)P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &= 2 - \frac{2\sum_{l=(z_i+1)/2+1}^{z_i^*}\left(1-\alpha^{z_i^*-l+1}\right)P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} + \frac{2\sum_{l=1}^{z_i-z_i^*}(\alpha^{z_i^*-l+1} - \alpha^{z_i-2l+2})P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &\quad - \frac{2\sum_{l=z_i-z_i^*+1}^{(z_i+1)/2}\left(\alpha^{z_i-2l+2} - \alpha^{z_i^*-l+1}\right)P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &\le 2 - \frac{2(1-\alpha)\sum_{l=(z_i+1)/2+1}^{z_i^*} P_{il}^*}{\sum_{l=1}^{z_i^*} P_{il}^*} \\ &\le 2 - \frac{2(1-\alpha)}{1 + \left[\sum_{l=1}^{(z_i+1)/2} P_{il}^* \Big/ \sum_{l=(z_i+1)/2+1}^{z_i^*} P_{il}^*\right]} \\ &\le 2 - \frac{2(1-\alpha)}{1 + \left[P_{i\max}^*\left(\frac{z_i+1}{2}\right)\right] / \left[P_{i\min}^*\left(z_i^* - \frac{z_i+1}{2}\right)\right]} \\ &= 2 - \frac{2(1-\alpha)(2z_i^* - z_i - 1)\eta}{(2z_i^* - z_i - 1)\eta + z_i + 1} \\ &\le 2 - \frac{2(1-\alpha)\left(\frac{7}{9}z_i^* - \frac{15}{9}\right)\eta}{(z_i^* - 1)\eta + \frac{11}{9}z_i^* + \frac{6}{9}} \\ &= 2 - \frac{2(1-\alpha)\left(\frac{7}{9} - \frac{15}{9z_i^*}\right)\eta}{\eta + \frac{11}{9} + \frac{6}{9z_i^*}} \\ &\le 2 - \frac{13(1-\alpha)\eta}{18\eta + 25}.\end{aligned} \tag{35}$$

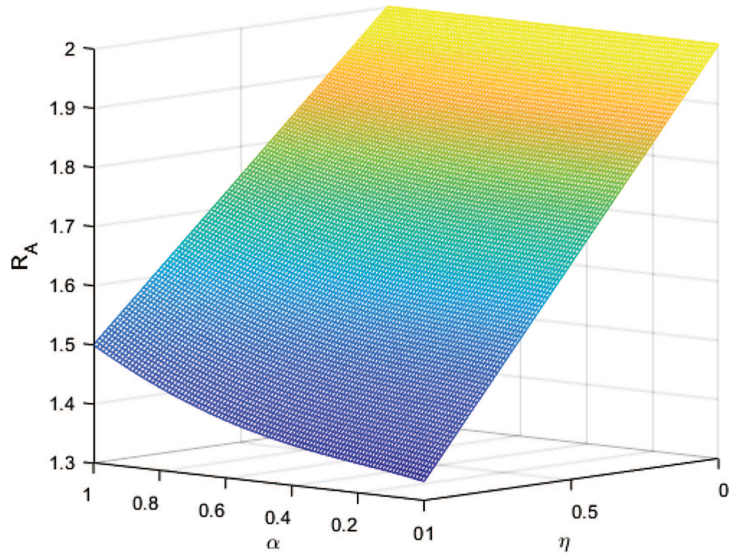

FIGURE 5. The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when $z_i = 3$.

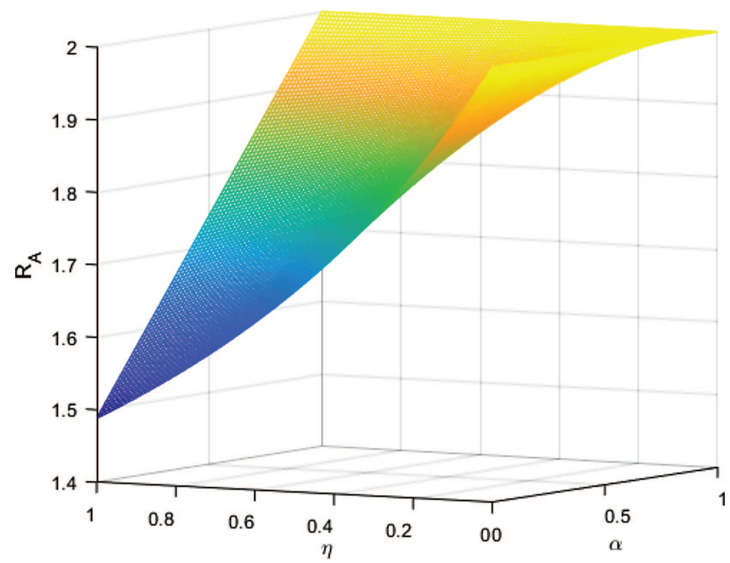

FIGURE 6. The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when z_i belongs to even.

The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when z_i belongs to even and odd are shown in Figures 6 and 7, respectively.

In all cases, the ratio of family processing time under Algorithm A_3 and the optimal schedule is less than 2. Therefore, we have

$$R_{A_3} = \frac{C_{\max}}{C_{\max}^*} = \frac{\sum_{i=1}^m S_i^A + \sum_{i=1}^m \sum_{l=1}^{z_i} P_{il}^A}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} \le \frac{\sum_{i=1}^m S_i^{*A} + 2\sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} < 2. \tag{36}$$

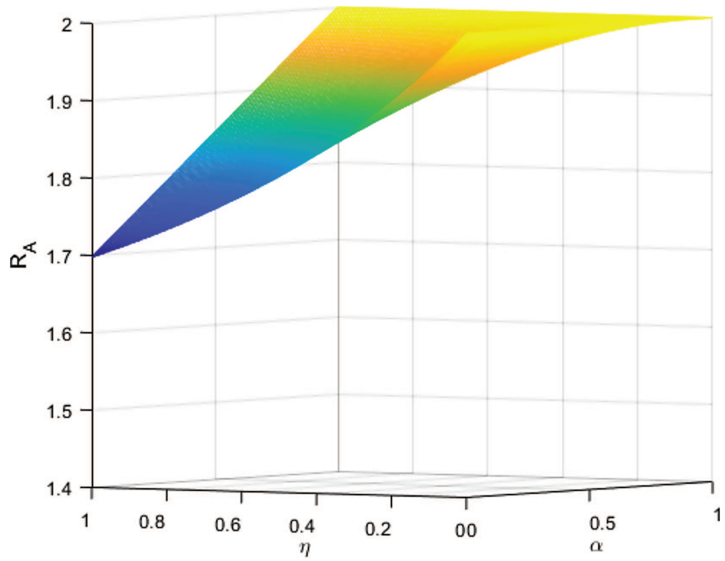

FIGURE 7. The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when z_i belongs to odd.

When the scale of ψ_3 approaches infinity, the asymptotic worst-case ratio are as follows. If z_i belongs to even, we have

$$\lim_{z_i\to\infty} \frac{\sum_{l=1}^{z_i} P_{il}^A}{\sum_{l=1}^{z_i^*} P_{il}^{*A}} \le \lim_{z_i\to\infty} 2 - \frac{2(1-\alpha)(\frac{7}{9}-\frac{15}{9z_i^*})\eta}{\eta+\frac{11}{9}+\frac{6}{9z_i^*}} = 2 - \frac{14(1-\alpha)\eta}{9\eta+11} < 2. \tag{37}$$

The same result can be obtained when z_i is odd. Figure 8 shows the value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when z_i approaching to infinity. Then we can find that $R_{A_3}^{\infty}$ is strictly less than 2.

$$R_{A_3}^{\infty} = \lim_{n\to\infty} \frac{C_{\max}}{C_{\max}^*} \le \lim_{n\to\infty} \frac{\sum_{i=1}^m S_i^A + \sum_{i=1}^m \sum_{l=1}^{z_i} P_{il}^A}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} < \frac{\sum_{i=1}^m S_i^{*A} + 2\sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}}{\sum_{i=1}^m S_i^{*A} + \sum_{i=1}^m \sum_{l=1}^{z_i^*} P_{il}^{*A}} < 2. \tag{38}$$

□

Theorem 3. *For ψ_3, the running time of A_3 is $O(n\log n)$, and $R_{A_3} < 2$, $R_{A_2}^{\infty} < 2$.*

7. Discussion

By the theoretical analysis of the parallel batch machine scheduling problem with incompatible deteriorating job families, we provide the following managerial insights to decision-makers of manufacturing enterprises.

First, a balance should be found between product categories and production costs. By comparing the three models, we find that the optimal scheduling can be obtained in polynomial time when the jobs have identical sizes. However, when the jobs have arbitrary sizes, the problem becomes NP-hard, which shows that the job size makes our problem complex. In addition, we find that the worst-case ratio is η decreasing function. When η approaches 1, the result of Algorithm A_3 is closer to the optimal solution. Based on the above, we propose that the diversity of products should be carefully considered in optimizing operation. Therefore, decision-makers should pay attention to the balance between product categories and cost. Moreover, measures should be taken

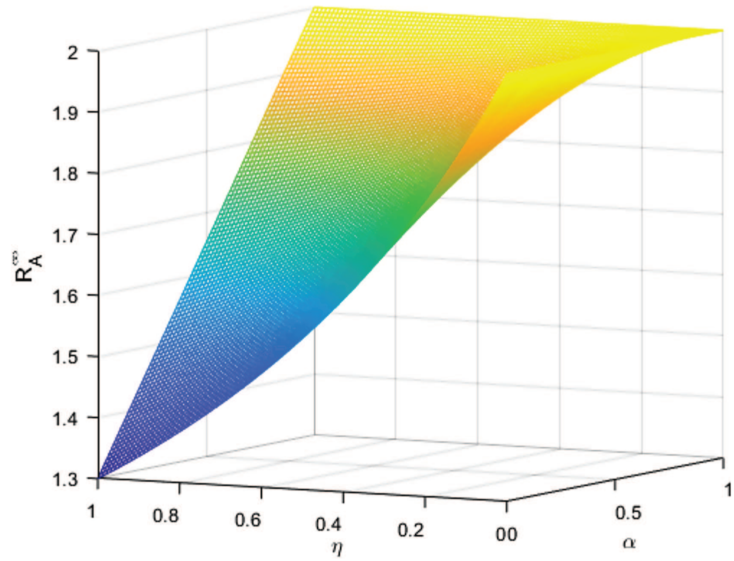

Figure 8. The value of $\sum_{l=1}^{z_i} P_{il}^A / \sum_{l=1}^{z_i^*} P_{il}^{*A}$ when z_i approaching to infinity.

to reduce the impact of job diversity on scheduling complexity, such as standardizing product size and adopting a delay strategy.

Second, it is important to improve the collaborative efficiency between processes. In order to minimize the impact of deterioration, decision-makers should take measures to improve the coordination efficiency between processes and reduce the waiting time of jobs. Firstly, manufacturers should optimize the layout of workshops and processing equipment, so as to reduce the transportation time in the workshop. Secondly, optimize the production scheduling. Develop a detailed scheduling plan to maintain workshop efficiency and operations between processes.

Third, measures should be taken to reduce the deteriorating rate. Based on the research in this paper, we find that when α increases, the worst-case ratio increases, indicating that deterioration not only reduces production efficiency, but also makes the scheduling problem more complex. Decision makers should take measures to reduce the deterioration rate. For example, in the soaking process, the initial temperature of the ingot can be maintained by thermal insulation packaging and increasing the ambient temperature.

8. Conclusions

In this paper, we study the single parallel batch machine scheduling problem with deteriorating incompatible jobs. The objective is to minimize makespan. Three models are considered and algorithms are proposed. In the first model, we propose an optimal polynomial time algorithm for the special case where the jobs have identical sizes. The optimality is proved. In the second model, we propose an approximate algorithm for the special case where the jobs have identical processing time. In the third model, an approximate algorithm is proposed for a more general case, that is, jobs have arbitrary sizes and arbitrary processing times. The latter two cases are proved to be NP-hard in the strong sense, and we show the absolute and asymptotic worst-case ratios of these two algorithms. All of the proposed algorithms run in $O(n \log n)$ time.

There are some interesting directions for future work. First, only single batch equipment is considered in this paper. Facility configuration is complex in practice and problems with other machine configurations, such as flow shops or parallel batch, are valuable to be researched. Since the single machine scheduling problem is NP-hard in the strong sense as studied in this paper, problems with complex facility configurations are also

NP-hard in the strong sense. Approximation algorithms and intelligent algorithms can be considered. Second, we only consider minimizing the makespan while multi-objective problems deserve study. For example, scheduling problems of purchasing, inventory and distribution. More objective functions are also interesting directions for future work, such as minimum service span or minimum total cost. Third, how to coordinate the scheduling of incompatible jobs and weaken the impact of deterioration is a direction worthy of research.

Acknowledgements. This work is partly supported by the National Natural Science Foundation of China under Grants 71671055, 72071056. This work is also partly supported by the National Key Research and Development Program of China 2019YFE0110300.

References

[1] J.N. Gupta and S.K. Gupta, Single facility scheduling with nonlinear processing times. *Comput. Ind. Eng.* **14** (1988) 387–393.

[2] M. Ji, X. Tang, X. Zhang and T.E. Cheng, Machine scheduling with deteriorating jobs and DeJong's learning effect. *Comput. Ind. Eng.* **91** (2016) 42–47.

[3] Y. Gao, J. Yuan, C.T. Ng and T.C.E. Cheng, A further study on two-agent parallel-batch scheduling with release dates and deteriorating jobs to minimize the makespan. *Eur. J. Oper. Res.* **273** (2019) 74–81.

[4] L. Tang, X. Zhao, J. Liu and J.Y.T. Leung, Competitive two-agent scheduling with deteriorating jobs on a single parallel-batching machine. *Eur. J. Oper. Res.* **263** (2017) 401–411.

[5] Y. Yin, Y. Wang, T.C.E. Cheng, W. Liu and J. Li, Parallel-machine scheduling of deteriorating jobs with potential machine disruptions. *Omega* **69** (2017) 17–28.

[6] X. Zhang, S.C. Liu, W.C. Lin and C.C. Wu, Parallel-machine scheduling with linear deteriorating jobs and preventive maintenance activities under a potential machine disruption. *Comput. Ind. Eng.* **145** (2020) 106482.

[7] S. Liu, J. Pei, H. Cheng, X. Liu and P.M. Pardalos, Two-stage hybrid flow shop scheduling on parallel batching machines considering a job-dependent deteriorating effect and non-identical job sizes. *Appl. Soft Comput.* **84** (2019) 105701.

[8] J. Pei, X. Liu, W. Fan, P.M. Pardalos and S. Lu, A hybrid BA-VNS algorithm for coordinated serial-batching scheduling with deteriorating jobs, financial budget, and resource constraint in multiple manufacturers. *Omega* **82** (2019) 55–69.

[9] K. Li, J. Chen, H. Fu, Z. Jia and J. Wu, Parallel machine scheduling with position-based deterioration and learning effects in an uncertain manufacturing system. *Comput. Ind. Eng.* **149** (2020) 106858.

[10] J. Ding, L. Shen, Z. Lü and B. Peng, Parallel machine scheduling with completion-time-based criteria and sequence-dependent deterioration. *Comput. Oper. Res.* **103** (2019) 35–45.

[11] S. Dauzère-Pérès and L. Mönch, Scheduling jobs on a single batch processing machine with incompatible job families and weighted number of tardy jobs objective. *Comput. Oper. Res.* **40** (2013) 1224–1233.

[12] S.S. Li and R.X. Chen, Single-machine parallel-batching scheduling with family jobs to minimize weighted number of tardy jobs. *Comput. Ind. Eng.* **73** (2014) 5–10.

[13] Z. Geng and J. Yuan, Pareto optimization scheduling of family jobs on a p-batch machine to minimize makespan and maximum lateness. *Theor. Comput. Sci.* **570** (2015) 22–29.

[14] B. Cheng, J. Cai, S. Yang and X. Hu, Algorithms for scheduling incompatible job families on single batching machine with limited capacity. *Comput. Ind. Eng.* **75** (2014) 116–120.

[15] L. Sun, A.J. Yu and B. Wu, Single machine common flow allowance group scheduling with learning effect and resource allocation. *Comput. Ind. Eng.* **139** (2020) 106126.

[16] A. Kramer, M. Iori and P. Lacomme, Mathematical formulations for scheduling jobs on identical parallel machines with family setup times and total weighted completion time minimization. *Eur. J. Oper. Res.* **289** (2021) 825–840.

[17] X. Li, Y. Li and Y. Huang, Heuristics and lower bound for minimizing maximum lateness on a batch processing machine with incompatible job families. *Comput. Oper. Res.* **106** (2019) 91–101.

[18] N. Alizadeh and A.H. Kashan, Enhanced grouping league championship and optics inspired optimization algorithms for scheduling a batch processing machine with job conflicts and non-identical job sizes. *Appl. soft Comput.* **83** (2019) 105657.

[19] E. Molaee, R. Sadeghian and P. Fattahi, Minimizing maximum tardiness on a single machine with family setup times and machine disruption. *Comput. Oper. Res.* **129** (2021) 105231.

[20] L. Mönch and S. Roob, A matheuristic framework for batch machine scheduling problems with incompatible job families and regular sum objective. *Appl. Soft Comput.*, **68** (2018) 835–846.

[21] V. Abu-Marrul, R. Martinelli, S. Hamacher and I. Gribkovskaia, Matheuristics for a parallel machine scheduling problem with non-anticipatory family setup times: application in the offshore oil and gas industry. *Comput. Oper. Res.* **128** (2021) 105162.

[22] C.C. Wu and W.C. Lee, Single-machine group-scheduling problems with deteriorating setup times and job-processing times. *Int. J. Prod. Econ.* **115** (2008) 128–133.

[23] W.C. Lee and Z.S. Lu, Group scheduling with deteriorating jobs to minimize the total weighted number of late jobs.*Appl. Math. Comput.* **218** (2012) 8750–8757.

[24] Y.T. Xu, Y. Zhang and X. Huang, Single-machine ready times scheduling with group technology and proportional linear deterioration. *Appl. Math. Model.* **38** (2014) 384–391.

[25] X. Zhang, L. Liao, W. Zhang, T.C.E. Cheng, Y. Tan and M. Ji, Single-machine group scheduling with new models of position-dependent processing times. *Comput. Ind. Eng.* **117** (2018) 1–5.

[26] P.J. Lai and W.C. Lee, Single-machine scheduling with general sum-of-processing-time-based and position-based learning effects. *Omega* **39** (2011) 467–471.

[27] G. Dósa, R. Li, X. Han and Z. Tuza, Tight absolute bound for first fit decreasing bin-packing: $FFD \leq (L)11/9OPT(L) + 6/9$. *Theo. Comput. Sci.* **510** (2013) 13–61.