Ann Oper Res (2017) 249:175–195
DOI 10.1007/s10479-015-1824-6

Scheduling deteriorating jobs on a single serial-batching machine with multiple job types and sequence-dependent setup times

Jun Pei · Xinbao Liu · Panos M. Pardalos · Wenjuan Fan · Shanlin Yang

Published online: 6 March 2015

Abstract In this paper, we study a scheduling model in which the features of deteriorating jobs, serial batches, multiple job types, and setup times are considered simultaneously. In this proposed model, the jobs of each type are first partitioned into serial batches, and then all batches of different job types are processed on a single serial-batching machine. The actual job processing time is an increasing function of its starting time, and the setup time of the batches is sequence-dependent, i.e., setup time is required only when a new batch is processed first on the machine or immediately after a batch belonging to another job type. We develop optimization algorithms to solve the makespan minimization problem, the maximum tardiness minimization problem, the maximum lateness minimization problem, and the maximum earliness minimization problem, respectively. We also propose optimization algorithms to solve the problem of minimizing the number of tardy jobs under a certain agreeable condition. Finally, we discuss two special cases of the total completion time minimization problem and develop optimization algorithms to solve them.

Keywords Scheduling · Deteriorating jobs · Serial-batching · Sequence-dependent setup time

Dedicated to Dr. Panos Pardalos on the occasion of his 60th birthday **(Pardalos60)**.

J. Pei (✉) · X. Liu · W. Fan · S. Yang
School of Management, Hefei University of Technology, Hefei, China
e-mail: feiyijun.ufl@gmail.com

J. Pei · P. M. Pardalos
Department of Industrial and Systems Engineering, Center for Applied Optimization, University of Florida, Gainesville, FL, USA

X. Liu · S. Yang
Key Laboratory of Process Optimization and Intelligent Decision-making of Ministry of Education, Hefei, China

W. Fan
Department of Computer Science, North Carolina State University, Raleigh, NC, USA

1 Introduction

In classic scheduling problems, the job processing times are assumed to be fixed during the entire production. However, this assumption might not be appropriate in many real-world situations. One is known as job deterioration in scheduling, and a deteriorating job can be described as its processing time grows when it awaits to be processed. The scheduling problem of deteriorating jobs was first introduced by Gupta and Gupta (1988) and Browne and Yechiali (1990), and extensive reviews on deteriorating jobs models were provided by Cheng et al. (2004) and Gawiejnowicz (2008). In recent years, more and more researchers have paid attentions to the scheduling problems with deteriorating jobs (Zhang et al. 2001).

As another popular topic in scheduling research, batch production exists in many production situations. Batch scheduling includes two types of processing, referring to parallel-batching and serial-batching. There are some papers addressing the parallel-batching scheduling problems with deteriorating jobs in recent years (Qi et al. 2009; Li et al. 2011; Miao et al. 2011, 2012). Qi et al. (2009) considered the unbounded model for several single machine problems, and they developed effective algorithms for minimizing the maximum cost, the number of tardy jobs, and the total weighted completion time, respectively. Li et al. (2011) investigated the problem of scheduling deteriorating jobs with release times on a single batch machine, where both unbounded and bounded models were considered. Miao et al. (2011) studied the bounded parallel-batch scheduling problem on single and multiple machines for deteriorating jobs. Miao et al. (2012) investigated the scheduling problem of deteriorating jobs on a single machine, where all jobs have different release times and the objective is to minimize the maximum lateness. However, research on serial-scheduling problems with deteriorating jobs is relatively unexplored. In Pei et al. (2015), we investigated the problem of the serial-batching scheduling with deteriorating jobs in an aluminum manufacturing factory for the first time, where all the jobs are first partitioned into serial batches and then processed on a single serial-batching machine. Before each batch is processed, an independent constant setup time is required. After further research on aluminum manufacturing process, in this paper we propose another type of scheduling problem with deteriorating jobs. Aluminum ingots are of multiple types, each of which includes a certain number of aluminum ingots. Each type of aluminum ingots is first partitioned into multiple batches, and then all batches of different job types are processed on a single serial-batching machine. The setup time before processing a batch is sequence-dependent. That is, if a new batch is the first batch to be processed on the machine or its previous batch is of a different job type, then the setup time for this new batch is required. Based on these two production characteristics, i.e., different job types and sequence-dependent setup times, the problem studied in this paper is different from that in our previous work (Pei et al. 2015). Thus, we investigate a new type of serial-batching scheduling problem with deteriorating jobs in practical production, and this research can further enhance productivity of the aluminum manufacturing factory.

Many papers on batch scheduling problems have taken into account sequence-dependent setup time as in this study (Agnetis et al. 2001; Detti et al. 2007; Liao et al. 2011), while this particular sequence-dependent setup time is different from the past-sequence-dependent (p-s-d) setup time considered in most scheduling problems with deteriorating jobs, where the p-s-d setup time is usually determined by the job position (Lai et al. 2011; Huang et al. 2013; Lee 2014). In addition, the group scheduling problems with deteriorating jobs are similar to the problem of our study. In the group scheduling problems, the jobs are classified into certain groups in advance based on similar production requirements. Wu et al. (2008) recently investigated the framework of minimizing the makespan and total completion time in group

Table 1 Comparisons of group scheduling problems with deteriorating jobs in recent research

Publications	Job processing time	Set-up time	Objectives
Wang et al. (2009)	$p_{ij} = a_{ij} + b_{ij}t$	$s_i = c_i + d_i t$	C_{max}
Wang and Sun (2010)	$p_{ij} = a_{ij} - b_{ij}t$	$s_i = c_i - d_i t$	C_{max}; $\sum w_i C_i$
Huang et al. (2011)	$p_{ijr}(t) = p_{ij}\left(\alpha a_i^{r-1} + \beta\right)(bt + c)$	$s_i = f(u_i)$	C_{max}; $\sum u_j$
Yang (2011)	$p_{ij}^r = p_{ij} + c_i t_{ir}$; $p_{ij}^r = p_{ij} r^{a_i}$; $p_{ij}^r = \left(p_{ij} + c_i t_{ir}\right) r^{a_i}$	$s_{ik} = s_i k^b$	C_{max}; TC
Bai et al. (2012)	$p_{ijr}^A = p_{ij} f_i \left(\sum_{l=1}^{r-1} p_{i[l]}\right) g_i(r)$	$s_i = a_i + b_i t$	C_{max}; $\sum C_j$
Lee and Lu (2012)	$p_j = \alpha_j t$	$s_i = \theta_i t$	$\sum w_j U_j$
Wang et al. (2012)	$p_{ij} = \alpha_{ij} t$	Constant	C_{max}
Wang et al. (2014)	$p_{ij}^A = \left(\frac{p_{ij}}{u_{ij}}\right)^k + \alpha t$	$s_i^A = s_i + \beta t - \beta_i$; $s_i^A = \left(\frac{s_i}{u_i}\right)^k + \beta t$	$\delta_1 C_{max} + \delta_2 \sum_{i=1}^{m} \sum_{j=1}^{n_j} V_{ij} u_{ij} + \delta_3 \sum_{i=1}^{m} W_i u_i$

scheduling problems with deteriorating jobs on a single-machine. Wang et al. (2008) recently studied the single-machine scheduling problem with deteriorating jobs and group technology assumptions and considered the makespan minimization problem and the total weighted completion time minimization problem, where the group setup times are described by a linear function. More recent papers that studied group scheduling problems with deteriorating jobs include (Wang et al. 2009; Wang and Sun 2010; Huang et al. 2011; Yang 2011; Bai et al. 2012; Lee and Lu 2012; Wang et al. 2012, 2014), which are compared in Table 1. There is some similarity between the serial-batching problems and group scheduling problems, that is, the jobs in a batch or group are processed one after another. However, there are also several significant differences between them which should be addressed here, as discussed in Table 2.

In this paper, we further extend our previous research, focusing instead on a general linear deterioration with different job types and sequence-dependent setup times. To the best of our knowledge, the problem of scheduling deteriorating jobs of different types on a serial-batching machine with sequence-dependent setup times has not been considered by other researchers, but it inevitably happens in many production areas. Based on the derived structural properties of the studied problem in this paper, some optimization algorithms are developed to solve the problems of minimizing the makespan, the maximum tardiness, the maximum lateness, and the maximum earliness, the number of tardy jobs, the total completion time of jobs, respectively. In the real production, the schedulers usually cannot well analyze the characteristics of these scheduling problems before making schedule plans, thus result in sub-optimal schedule plans. Furthermore, since the schedule rules are not specific and the workers may operate optionally when performing the schedule plans, so the productivity is low. This paper analyzes the problems with different objectives, and develops algorithms based on the properties of the problem. The procedures of these algorithms are clear and

Table 2 Differences between group and serial-batching scheduling problems

Factors	Group scheduling problems (Wang et al. 2009; Wang and Sun 2010; Huang et al. 2011; Yang 2011; Bai et al. 2012; Lee and Lu 2012; Wang et al. 2012, 2014)	Serial-batching scheduling problems
Classifying/batching	Job has been classified into certain groups beforehand	Both decisions on job batching and batch sequencing need to be made for each type of jobs
Job completion time	The jobs in the same group have different completion times	The jobs in the same batch have the same completion time which is equal to the completion time of the last job in that batch
Machine capacity	The machine capacity doesn't need to be considered	The machine capacity should be taken into account
Setup time	Setup time is needed before each group if it is considered	Setup time is required when a new batch is the first batch to be processed on the machine or its previous batch is of a different type

definite with high operability, which can be directly used in the real production to enhance productivity.

The reminder of this paper is organized as follows. The notation and problem description are given in Sect. 2. The solution procedures for the makespan minimization problem, the maximum tardiness minimization problem, the maximum lateness minimization problem, and the maximum earliness minimization problem are described in Sects. 3 and 4, respectively. We present our solution procedure for the problem of minimizing the number of tardy jobs under a certain agreeable condition in Sect. 5. In Sect. 6, two special cases are analyzed for the total completion time minimization problem. Finally, the conclusion is given in Sect. 7.

2 Notation and problem description

We first describe the notation used in this paper in Table 3.

There are N jobs to be processed on a single serial-batching machine, each of which belongs to one of n job types. Each type of jobs is first partitioned into multiple serial batches. Then the batches of different job types are processed on a single machine. Serial batches require that all the jobs within the same batch are processed one after another in a serial fashion (Xuan and Tang 2007), and the completion times of all jobs are equal to that of their batch, which is defined as the completion time of the last job in the batch. When a new batch is processed first on the machine or immediately after a batch of another job type, a setup time s is required. The number of jobs in a batch cannot be more than the machine capacity b, that is, $n_k \leq b$. All jobs are available at time t_0, where $t_0 > 0$. As in Pei et al. (2015), the actual processing time of J_i is indicated as a linear function of its starting time t, that is,

$$p_i = a_i t, \quad i = 1, 2, \cdots, N$$

where a_i and t are the deterioration rate and the starting time of J_i, respectively.

Table 3 The list of the notation

n	The number of job types
F_l	The job set for job type l, $l = 1, 2, \cdots, n$
q_l	The number of jobs in F_l, $l = 1, 2, \cdots, n$
$a\,(F_l)$	The deteriorating rate of the jobs from job type l, $l = 1, 2, \cdots, n$
N	The total number of jobs, i.e., $N = q_1 + q_2 + \cdots + q_n$
J_i	Job i, $i = 1, 2, \cdots, N$
p_i	The actual processing time of job i, $i = 1, 2, \cdots, N$
m	The number of batches
b_k	Batch k, $k = 1, 2, \cdots, m$
n_k	The number of jobs in batch k, $k = 1, 2, \cdots, m$
$S\,(b_k)$	The starting time of batch k, $k = 1, 2, \cdots, m$
$C\,(b_k)$	The completion time of batch k, $k = 1, 2, \cdots, m$
a_i	The deteriorating rate of the processing time for job i,$a_i = a\,(F_l)$ for $J_i \in F_l, i = 1, 2, \cdots, N$, $l = 1, 2, \cdots, n$
$b\,(k)$	The deteriorating rate of batch k, $b\,(k) = a_i$ for $J_i \in b_k$,$i = 1, 2, \cdots, N$, $k = 1, 2, \cdots, m$
s	The setup time
d	The common due date of all jobs
b	The capacity of the serial-batching machine, i.e., the maximum number of jobs in a batch
x_{kj}	If batches k and j belong to different job types, then $x_{kj} = 1$; otherwise, $x_{kj} = 0$
π	A schedule of N jobs
$C_i\,(\pi)$	The completion time of job i in a given schedule π, $i = 1, 2, \cdots, N$
C_{max}	The makespan of all jobs
T_{max}	The maximum tardiness of all jobs
L_{max}	The maximum lateness of all jobs
E_{max}	The maximum earliness of all jobs
$\sum_{i=1}^{N} U_i$	The number of tardy jobs
$\sum_{i=1}^{N} C_i$	The total completion time of all jobs

For a schedule π, let $C_i\,(\pi)$ be the completion time of J_i. Using the traditional notation, we adopt $C_{max} = \max_{i=1,2,\cdots,N}\{C_i\}$, $T_{max} = \max_{i=1,2,\cdots,N}\{0, C_i - d\}$, $L_{max} = \max_{i=1,2,\cdots,N}\{C_i - d\}$, $E_{max} = \max_{i=1,2,\cdots,N}\{0, d - C_i\}$, $\sum_{i=1}^{N} U_i$, and $\sum_{i=1}^{N} C_i$ to represent the makespan, maximum tardiness, maximum lateness, maximum earliness, total number of tardy jobs, and total completion time, respectively, where $U_i = 1$ if $C_i > d$ and 0 otherwise. In the remaining sections of the paper, all the problems considered will be denoted using the three-field notation schema $\alpha\,|\beta|\,\gamma$ introduced by Graham et al. (1979).

3 Problem $1\,|s - batch, p_i = a_i t, s_{sd}|\,C_{max}$

In this section, the makespan minimization problem $1\,|s - batch, p_i = a_i t, s_{sd}|\,C_{max}$ is studied. Some properties of the makespan minimization problem for any given schedule π are given in the following.

Lemma 1 *For any given schedule $\pi = (b_1, b_2, \cdots, b_m)$, with the first batch b_1 starting at time $t_0 > 0$, the makespan of schedule π is*

$$C_{max}(\pi) = (t_0 + s)\prod_{k=1}^{m}(1 + b(k))^{n_k} + \sum_{k=2}^{m} x_{(k-1)k} s \prod_{f=k}^{m}(1 + b(f))^{n_f} \tag{1}$$

Proof We can use mathematical induction to prove this lemma based on the number of batches. Firstly for $m = 1$ we have

$$C(b_1) = (t_0 + s)(1 + b(1))^{n_1},$$

so Eq. (1) holds for $m = 1$. Suppose for all $2 \le j \le m - 1$, Eq. (1) is satisfied. We have

$$C(b_j) = (t_0 + s)\prod_{k=1}^{j}(1 + b(k))^{n_k} + \sum_{k=2}^{j} x_{(k-1)k} s \prod_{f=k}^{j}(1 + b(f))^{n_f}.$$

Then, for the $(j + 1)$th batch b_{j+1},

$$\begin{aligned}
C(b_{j+1}) &= \left[C(b_j) + x_{j(j+1)}s\right](1 + b(j+1))^{n_{(j+1)}} \\
&= \left[(t_0 + s)\prod_{k=1}^{j}(1 + b(k))^{n_k} + \sum_{k=2}^{j} x_{(k-1)k} s \prod_{f=k}^{j}(1 + b(f))^{n_f} \right. \\
&\quad \left. + x_{j(j+1)}s\right](1 + b(j+1))^{n_{(j+1)}} \\
&= (t_0+s)\prod_{k=1}^{j+1}(1 + b(k))^{n_k} + (1 + b(j+1))^{n_{(j+1)}} \cdot \sum_{k=2}^{j} x_{(k-1)k} s \prod_{f=k}^{j}(1+b(f))^{n_f} \\
&\quad + x_{j(j+1)}s \cdot (1 + b(j+1))^{n_{(j+1)}} \\
&= (t_0 + s)\prod_{k=1}^{j+1}(1 + b(k))^{n_k} + \sum_{k=2}^{j+1} x_{(k-1)k} s \prod_{f=k}^{j+1}(1 + b(f))^{n_f}.
\end{aligned}$$

Hence, Eq. (1) holds for $m = j + 1$. Note that $C_{max}(\pi) = C(b_m)$, the lemma is proved by the induction. □

Lemma 2 *For the problem $1|s - batch, p_i = a_i t, s_{sd}|C_{max}$, if there exits $x_{k(k+1)} = 0\ (k = 1, 2, \cdots, m - 1)$ in a given schedule, then the solution of this schedule remains unchanged when b_k and b_{k+1} are swapped.*

Lemma 3 *For the problem $1|s - batch, p_i = a_i t, s_{sd}|C_{max}$, if any two jobs are swapped in a batch for a given schedule, then the solution of this schedule remains unchanged.*

Based on Lemmas 2 and 3, we have the following corollary.

Corollary 1 *For the problem $1|s - batch, p_i = a_i t, s_{sd}|C_{max}$, if there exits $x_{k(k+1)} = 0\ (k = 1, 2, \cdots, m - 1)$ in a given schedule, then the solution of this schedule is independent of the pattern of jobs batching and batches sequencing of the jobs in these two batches.*

Lemma 4 *For the problem $1|s - batch, p_i = a_i t, s_{sd}|C_{max}$, the generated batches from the same job type are processed consecutively in an optimal schedule.*

Proof Let π^* and π be an optimal schedule and a job schedule, respectively. The difference between the two schedules is the transfer of a batch b_q, that is, $\pi^* = (W_1, b_p, W_2, b_q, W_3)$, $\pi = (W_1, b_p, b_q, W_2, W_3)$. After b_p, b_q is the first batch of the same job type as b_p, W_1,W_2, and W_3 represent three partial sequences, W_1 or W_3 may be empty, and W_2 is not empty. It is easy to see that $x_{p(p+1)} = 1$ and $x_{(q-1)q} = 1$.

For π^*, the completion time of b_q is

$$\begin{aligned}
C\left(b_q\left(\pi^*\right)\right) &= \left[C\left(b_{q-1}\left(\pi^*\right)\right) + s\right](1 + b\,(q))^{n_q} \\
&= \left[C\left(b_{p+1}\left(\pi^*\right)\right) \prod_{k=p+2}^{q-1} (1 + b\,(k))^{n_k} \right. \\
&\quad \left. + \sum_{k=p+2}^{q-1} x_{(k-1)k} s \prod_{f=k}^{q-1} (1 + b\,(f))^{n_f} + s \right] (1 + b\,(q))^{n_q} \\
&= \left[\left(C\left(b_p\left(\pi^*\right)\right) + s\right)(1 + b\,(p+1))^{n_{p+1}} \prod_{k=p+2}^{q-1} (1 + b\,(k))^{n_k} \right. \\
&\quad \left. + \sum_{k=p+2}^{q-1} x_{(k-1)k} s \prod_{f=k}^{q-1} (1 + b\,(f))^{n_f} + s \right] (1 + b\,(q))^{n_q} \\
&= \left(C\left(b_p\left(\pi^*\right)\right) + s\right) \prod_{k=p+1}^{q} (1 + b\,(k))^{n_k} \\
&\quad + (1 + b\,(q))^{n_q} \sum_{k=p+2}^{q-1} x_{(k-1)k} s \prod_{f=k}^{q-1} (1 + b\,(f))^{n_f} \\
&\quad + s\,(1 + b\,(q))^{n_q}.
\end{aligned}$$

For π, the completion time of b_{p+1} and b_{q-1} are respectively

$$\begin{aligned}
C\left(b_{p+1}\left(\pi\right)\right) &= \left[C\left(b_q\left(\pi\right)\right) + s\right](1 + b\,(p+1))^{n_{p+1}} \\
&= C\left(b_p\left(\pi^*\right)\right)(1 + b\,(q))^{n_q}\,(1 + b\,(p+1))^{n_{p+1}} \\
&\quad + s\,(1 + b\,(p+1))^{n_{p+1}}, C\left(b_{q-1}\left(\pi\right)\right) \\
&= C\left(b_{p+1}\left(\pi\right)\right) \prod_{k=p+2}^{q-1} (1 + b\,(k))^{n_k} + \sum_{k=p+2}^{q-1} x_{(k-1)k} s \prod_{f=k}^{q-1} (1 + b\,(f))^{n_f} \\
&= C\left(b_p\left(\pi^*\right)\right) \prod_{k=p+1}^{q} (1 + b\,(k))^{n_k} + s \prod_{k=p+1}^{q-1} (1 + b\,(k))^{n_k} \\
&\quad + \sum_{k=p+2}^{q-1} x_{(k-1)k} s \prod_{f=k}^{q-1} (1 + b\,(f))^{n_f}.
\end{aligned}$$

Then, we have

$$C\left(b_q\left(\pi^*\right)\right) - C\left(b_{q-1}\left(\pi\right)\right) = s\left[\left(1+b\left(q\right)\right)^{n_q} - 1\right]\left[\prod_{k=p+1}^{q-1}\left(1+b\left(k\right)\right)^{n_k} + \sum_{k=p+2}^{q-1} x_{(k-1)k}\prod_{f=k}^{q-1}\left(1+b\left(f\right)\right)^{n_f}\right] + s\left(1+b\left(q\right)\right)^{n_q} > 0.$$

It can be deduced that $C\left(b_q\left(\pi^*\right)\right) > C\left(b_{q-1}\left(\pi\right)\right)$, which conflicts with the optimality of C^*_{max}. Thus, we can transfer the batches until the batches of the same job type are processed consecutively. This completes the proof. □

Lemma 5 *For the problem* $1\,|s-batch,\ p_i = a_i t, s_{sd}|\,C_{max}$*, if the generated batches of the same job type are processed consecutively in a given schedule* π*, then the makespan of schedule* π *is*

$$C_{max}\left(\pi\right) = t_0\prod_{l=1}^{n}\left(1+a\left(F_l\right)\right)^{q_l} + s\sum_{l=1}^{n}\prod_{f=l}^{n}\left(1+a\left(F_f\right)\right)^{q_f} \tag{2}$$

Proof The proof is similar to that of Lemma 1, and it is omitted. □

Lemma 6 *For the problem* $1\,|s-batch,\ p_i = a_i t, s_{sd}|\,C_{max}$*, considering two consecutive job sets of types* r *and* $r+1$*, if* $\rho\left(F_r\right) \geq \rho\left(F_{r+1}\right)$*, where* $\rho\left(F_r\right) = \left(1+a\left(F_r\right)\right)^{q_r}, r = 1, 2, \cdots, n-1$*, then it is optimal to process* F_r *before* F_{r+1}*.* □

Proof Let π^* and π be an optimal schedule and a job schedule. The difference between these two schedules is the pairwise interchange of these two job sets F_r and F_{r+1} $(r = 1, 2, \cdots, n-1)$, that is, $\pi^* = (W_1, F_r, F_{r+1}, W_2)$, $\pi = (W_1, F_{r+1}, F_r, W_2)$, where F_r and F_{r+1} are the job sets of types r and $r+1$, both of which may include one or multiple batches, W_1 and W_2 represent two partial sequences, and W_1 or W_2 may be empty. We assume that $\rho\left(F_r\right) < \rho\left(F_{r+1}\right)$, i.e., $\left(1+a\left(F_r\right)\right)^{q_r} < \left(1+a\left(F_{r+1}\right)\right)^{q_{r+1}}$. □

For π^*, the completion time of F_{r+1} is

$$C\left(F_{r+1}\left(\pi^*\right)\right) = t_0\prod_{l=1}^{r+1}\left(1+a\left(F_l\right)\right)^{q_l} + s\sum_{l=1}^{r+1}\prod_{f=l}^{r+1}\left(1+a\left(F_f\right)\right)^{q_f},$$

For π, the completion time of F_{r+1} and F_r are respectively

$$\begin{aligned}
&C\left(F_{r+1}\left(\pi\right)\right)\\
&= \left[t_0\prod_{l=1}^{r-1}\left(1+a\left(F_l\right)\right)^{q_l} + s\sum_{l=1}^{r-1}\prod_{f=l}^{r-1}\left(1+a\left(F_f\right)\right)^{q_f} + s\right]\left(1+a\left(F_{r+1}\right)\right)^{q_{r+1}}, C\left(F_r\left(\pi\right)\right)\\
&= \left[C\left(F_{r+1}\left(\pi\right)\right) + s\right]\left(1+a\left(F_r\right)\right)^{q_r}\\
&= \left\{\left[t_0\prod_{l=1}^{r-1}\left(1+a\left(F_l\right)\right)^{q_l} + s\sum_{l=1}^{r-1}\prod_{f=l}^{r-1}\left(1+a\left(F_f\right)\right)^{q_f} + s\right]\left(1+a\left(F_{r+1}\right)\right)^{q_{r+1}} + s\right\}\\
&\quad \left(1+a\left(F_r\right)\right)^{q_r}\\
&= t_0\prod_{l=1}^{r+1}\left(1+a\left(F_l\right)\right)^{q_l} + s\sum_{l=1}^{r}\prod_{f=l}^{r+1}\left(1+a\left(F_f\right)\right)^{q_f} + s\left(1+a\left(F_r\right)\right)^{q_r}.
\end{aligned}$$

Then,

$$\begin{aligned} & C\left(F_{r+1}\left(\pi^{*}\right)\right)-C\left(F_{r}\left(\pi\right)\right) \\ &= s\sum_{l=1}^{r+1}\prod_{f=l}^{r+1}\left(1+a\left(F_{f}\right)\right)^{q_f}-\left[s\sum_{l=1}^{r}\prod_{f=l}^{r+1}\left(1+a\left(F_{f}\right)\right)^{q_f}+s\left(1+a\left(F_{r}\right)\right)^{q_r}\right] \\ &= s\left[\left(1+a\left(F_{r+1}\right)\right)^{q_{r+1}}-\left(1+a\left(F_{r}\right)\right)^{q_r}\right]>0, \end{aligned}$$

which conflicts with the optimal schedule. Hence, $(1+a(F_r))^{q_r} \geq (1+a(F_{r+1}))^{q_{r+1}}$. This proves the lemma.

Thus, Lemmas 4 and 6 imply that all jobs from the same job type are processed consecutively and the job sets of all types should be sequenced in non-increasing order of $\rho(F_r)$ in an optimal schedule. There is only once setup for each type of job sets.

Based on the above lemmas, the following Algorithm 1 is designed to solve the problem $1|s-batch, p_i=a_it, s_{sd}|C_{max}$.

Algorithm 1

Step 1.	Calculate $\rho(F_l)=(1+a(F_l))^{q_l}, l=1,2,\cdots,n$.
Step 2.	Sequence the jobs of the same type together, and then sequence the job sets of all types in non-increasing order of $\rho(F_l)$, i.e., $\rho(F_1)\geq\rho(F_2)\geq\cdots\geq\rho(F_n)$. Set $l=0$.
Step 3.	Set $l=l+1$.
Step 4.	If there are more than b jobs in F_l, then place the first b jobs in a batch and iterate. Otherwise, place the remaining jobs in a batch.
Step 5.	If $l=n$, then stop and schedule the batches in their generation order at timet_0. Otherwise, go to step 3.

Compared with Algorithm R-FBLDR proposed in Pei et al. (2015), the jobs of the same type should be first sequenced together and then the job sets of all types is sequenced based on the value of $\rho(F_l)$ in Algorithm 1.

Theorem 1 *For the problem* $1|s-batch, p_i=a_it, s_{sd}|C_{max}$, *an optimal schedule can be obtained by Algorithm 1 in* $O(N\log N)$ *time. If the job sets of all types are sequenced in non-increasing order of* $(1+a(F_l))^{q_l}$ $(l=1,2,\cdots,n)$, *then the optimal makespan is*

$$C_{max}^{*}=t_0\prod_{l=1}^{n}(1+a(F_l))^{q_l}+s\sum_{l=1}^{n}\prod_{f=l}^{n}\left(1+a\left(F_f\right)\right)^{q_f} \tag{3}$$

Proof Based on Lemmas 1–6, an optimal solution can be generated by Algorithm 1. We can also obtain the result of the optimal solution as Eq. (3) based on Lemma 5. The time complexity of step 1 is $O(n)$ and the time complexity of obtaining the optimal job set sequence of all types in step 2 is $O(n\log n)$, and the total time complexity of steps 3, 4, and 5 is $O(n)$. Then, we have $n\leq N$. Thus, the time complexity of Algorithm 1 is at most $O(N\log N)$.

Corollary 2 *Algorithm 1 can obtain an optimal schedule for the problem* $1|s-batch, p_i=a_it, s_{sd}|T_{max}$.

Proof It can be deduced that

$$T_{max} = \max_{i=1,2,\cdots,N} \{0, C_i - d\} = \max \left\{0, \max_{i=1,2,\cdots,N} \{C_i\} - d\right\} = \max \{0, C_{max} - d\}.$$

Since d is a constant, the maximum tardiness of all jobs is minimized when C_{max} is minimized. Based on Theorem 1, Algorithm 1 can obtain the minimum C_{max} for the problem $1\,|s - batch, p_i = a_i t, s_{sd}|\, C_{max}$. This completes the proof. □

Corollary 3 *Algorithm 1 can obtain an optimal schedule for the problem* $1\,|s - batch, p_i = a_i t, s_{sd}|\, L_{max}$.

Proof We have $L_{max} = \max_{i=1,2,\cdots,N} \{C_i - d\}$. Hence, the proof is similar to that of Corollary 2, and it is omitted. □

Corollaries 2 and 3 imply that Algorithm 1 can obtain an optimal schedule for the problems $1\,|s - batch, p_i = a_i t, s_{sd}|\, C_{max}$, $1\,|s - batch, p_i = a_i t, s_{sd}|\, T_{max}$, and $1\,|s - batch, p_i = a_i t, s_{sd}|\, L_{max}$ simultaneously.

4 Problem $1\,|s - batch, p_i = a_i t, s_{psd}|\, E_{max}$

In this section, we focus on the problem of minimizing the maximum earliness of all jobs. As commonly assumed in the previous work involving earliness (Yin et al. 2012), all jobs are restricted to be completed prior to the common due date d, otherwise each job can be trivially scheduled sufficiently late to avoid earliness cost. Thus, it should be $d \geq t_0 + \sum_{l=1}^{n} p_i$. Let π be a feasible schedule. The earliness of job J_i is given by $E_i = \{0, d - C_i\}$, and the maximum earliness is defined as $E_{max} = \max_{i=1,2,\cdots,N} E_i$. We first develop some properties as follows.

Lemma 7 *There is an optimal schedule with the first batch b_1 starting at time t_0 such that $(t_0 + s)\prod_{k=1}^{m}(1 + b(k))^{n_k} + \sum_{k=2}^{m} x_{(k-1)k} s \prod_{f=k}^{m}(1 + b(f))^{n_f} = d$, and there is no idle time between consecutive batches or consecutive jobs in the same batch.*

Proof All jobs need to be scheduled as late as possible and also satisfy the constraint that their completion times are prior to the common due date. Hence, the completion time of the last batch should be just equal to the common due date. Then, we have $(t_0 + s)\prod_{k=1}^{m}(1 + b(k))^{n_k} + \sum_{k=2}^{m} x_{(k-1)k} s \prod_{f=k}^{m}(1 + b(f))^{n_f} = d$ based on Lemma 1. In addition, if there is any idle time between consecutive batches or consecutive jobs in the same batch, then the maximum earliness will be increased as the starting time t_0 gets shorter. Thus, there is no idle time between them. □

Lemma 8 *There is an optimal schedule in which the number of jobs in the first batch b_1 is $n_1 = min\{b, q_r\}$, where the jobs in b_1 are of the job type r, and $r = 1, 2, \cdots, n$.*

Proof Since the completion time of a job is equal to the completion time of the batch it belongs to, the maximum earliness will be decreased as the processing time of the first batch b_1 becomes longer. It can be derived that the first batch should have the possible maximum job number. Thus, we have $n_1 = min\{b, q_r\}$. □

Based on Lemmas 4 and 6, we have the following property.

Lemma 9 *There is an optimal schedule in which all batches of the same job type are processed consecutively since the second batch, and the sets of all job types are processed in non-increasing order of* $\rho(F_r)$*, where* $\rho(F_r) = (1 + a(F_r))^{q_r}$, $r = 1, 2, \cdots, n$.

Proof We omit the proof as it is similar to that of Lemmas 4 and 6. □

Lemma 10 *For the problem* $1|s - batch, p_i = a_i t, s_{sd}|E_{max}$*, if there exists* $n_r > b$ *for a certain job type* r $(r = 1, 2, \cdots, n)$*, where* $b_1 \subset F_r$*, then the leftover jobs of job type* r *except all the jobs in* b_1 *are processed consecutively after* b_1.

Proof Let π^* and π be an optimal schedule and a job schedule. The difference between the two schedules is the insertion of the job set F_1^2 after F_1^1, that is, $\pi^* = (F_1^1, W_1, F_1^2, W_2)$, $\pi = (F_1^1, F_1^2, W_1, W_2)$, where F_1^1 and F_1^2 are two partial job sets of job type 1, both of them may include one or multiple batches and $b_1 \subseteq F_1^1$, W_1 and W_2 represent two partial sequences, and W_2 may be empty. In this case, we have $n_1 = b$ based on Lemma 8. Let the starting time of b_1 in π^* and π be t_0^* and t_0, respectively. Then,

$$E_{max}(\pi^*) = d - (t_0^* + s)(1 + a_1)^b$$

and

$$E_{max}(\pi) = d - (t_0 + s)(1 + a_1)^b.$$

Thus,

$$E_{max}(\pi^*) - E_{max}(\pi) = (t_0 - t_0^*)(1 + a_1)^b.$$

Based on Lemmas 5 and 9, it can be derived that

$$t_0 - t_0^* > \frac{s(1 + a_1)}{\prod_{l=1}^{n}(1 + a(F_l))^{q_l}} > 0.$$

Then,

$$E_{max}(\pi^*) > E_{max}(\pi),$$

Which conflicts with the optimal schedule, and this completes the proof. □

Based on the above lemmas, we propose the following Algorithm 2 to solve the problem $1|s - batch, p_i = a_i t, s_{sd}|E_{max}$.

Theorem 2 *For the problem* $1|s - batch, p_i = a_i t, s_{sd}|E_{max}$*, an optimal schedule can be obtained by Algorithm 2 in* $O(N^2 \log N)$ *time.*

Proof Based on Lemmas 7–10, Algorithm 2 can generate an optimal solution. The time complexity of step 1 is $O(1)$, the total time complexity of steps 2, 3, 4, and 5 is $O(n^2 \log n)$, and the total time complexity of steps 6, 7, 8, and 9 is $O(n)$. We also have $n \leq N$. Thus, the time complexity of Algorithm 2 is at most $O(N^2 \log N)$. □

5 Problem $1|s - batch, p_i = a_i t, s_{sd}|\sum_{i=1}^{N} U_i$

In the following section, we first give some properties for the general problem of minimizing the number of tardy jobs $1|s - batch, p_i = a_i t, s_{sd}|\sum_{i=1}^{N} U_i$ and a property for this problem

Algorithm 2

Step 1. Set $\theta = d$ and $l = 0$, and let σ be the sequence of the job sets of all types.
Step 2. Set $l = l + 1$.
Step 3. Set the job set of the lth type as the first place in the sequence. Sequence the leftover jobs of the same type together, and then sequence the job sets of these leftover $(n-1)$ types in non-increasing order of $\rho(F_l)$, i.e., $\rho(F_2) \geq \rho(F_3) \geq \cdots \geq \rho(F_{n-1})$, where $\rho(F_l) = (1 + a(F_l))^{q_l}$, $l = 1, 2, \cdots, n-1$. Calculate t_0 based on Lemmas 5 and 10, and then calculate $E_{max}(l) = d - (t_0 + s)(1 + a_1)^{min\{q_1, b\}}$.
Step 4. If $\theta < E_{max}(l)$, then set σ be the sequence of the job sets of all types in step 3.
Step 5. If $l = n$, then go to step 6. Otherwise, go to step 2.
Step 6. Let set σ be the final sequence of the job sets of all types and set $l = 0$.
Step 7. Set $l = l + 1$.
Step 8. If there are more than b jobs in F_l, then place the first b jobs in a batch and iterate. Otherwise, place the remaining jobs in a batch.
Step 9. If $l = n$, then stop and schedule the batches in their generation order at timet_0. Otherwise, go to step 7.

under an agreeable condition, and then develop an optimization algorithm to solve it. The job sets of which the completion times are no more than and more than the common due date d are denoted as O (i.e., ordinary jobs) and L (i.e., late jobs), respectively. Let the job set and the job number of type l be F_l^O and q_l^O in O, and F_l^L and q_l^L in L, respectively, where $F_l^O \cup F_l^L = F_l$ and $q_l^O + q_l^L = q_l \quad (l = 1, 2, \cdots, n)$. Let the total number of job types be n^O in O.

Lemma 11 *For the problem* $1|s-batch, p_i = a_i t, s_{sd}|\sum_{i=1}^{N} U_i$, *the solution of a given schedule remains unchanged when: (1) two batches* b_k *and* b_{k+1} *satisfying that* $x_{k(k+1)} = 0\,(k = 1, 2, \cdots, m-1)$ *are swapped;(2) any two jobs are swapped in a batch.*

Lemma 12 *For the problem* $1|s-batch, p_i = a_i t, s_{sd}|\sum_{i=1}^{N} U_i$, *an optimal schedule satisfies the following properties:*

(1) The generated batches of the same job type are processed consecutively in O;
(2) If $\sigma(F_r^O) \geq \sigma(F_{r+1}^O)$ *for two consecutive job sets of job types r and* $r+1$ *in O, where* $\sigma(F_r^O) = (1 + a(F_r^O))^{q_r^O}$, $r = 1, 2, \cdots, n-1$, *then it is optimal to process* F_r^O *before* F_{r+1}^O;

Proof The proof of (1) and (2) is similar to that of Lemmas 4 and 6, and it is omitted here. □

We focus on the agreeable condition that $a(F_r) < a(F_u)$ implies $\rho(F_r) > \rho(F_u)$ for two job types r and u, where $\rho(F_l) = (1 + a(F_l))^{q_l}$ and $r, u = 1, 2, \cdots, n$. This agreeable condition is denoted as ac for simplicity. Then we have the following property.

Lemma 13 *For the problem* $1|s-batch, p_i = a_i t, s_{sd}, ac|\sum_{i=1}^{N} U_i$, *the deterioration rate of an arbitrary job in O is no more than that of any jobs in L in an optimal schedule.*

Proof Assume there exist two job sets $F_r^O, F_u^O \subseteq O$ in an optimal schedule, satisfying that $a(F_r) < a(F_u)$, $\rho(F_r) > \rho(F_u)$, and $q_r^O, q_r^L, q_u^O > 0$. Since $a(F_r) < a(F_u)$ and $\rho(F_r) > \rho(F_u)$, it can be derived that $n_r > n_u$. Let the completion time of all jobs in O be $C(O)$. Based on Lemmas 11 and 12, we have two cases in an optimal schedule as follows.

Case 1. $O = (W_1, F_r^o, W_2, F_u^o, W_3)$, where W_1,W_2, and W_3 represent three partial sequences.

(1) when $q_r^L \geq q_u^O$, we can replace all jobs of F_u^o with the jobs of F_r^L, and O is updated to $O' = \left(W_1, F_r^O, W_2, F_r^{L'}, W_3\right)$. It is easy to see that $C(O) > C\left(O'\right)$, which conflicts with the optimal schedule.

(2) when $q_r^L < q_u^O$, we can add all jobs of F_r^L following F_r^o and transfer the jobs of F_u^O with the job number of q_r^L into L. O is updated to $O' = \left(W_1, F_r^{O'}, W_2, F_u^{O'}, W_3\right)$, where $q_r^{O'} = q_r$ and $q_u^{O'} = q_u^O - q_r^L$. Thus,

$$\begin{aligned} C(O) - C\left(O'\right) = & \left[t_0 \prod_{l=1}^{n^O} (1 + a(F_l))^{q_l^O} + s \sum_{l=1}^{r} \prod_{f=l}^{n^O} \left(1 + a\left(F_f\right)\right)^{q_f^O} \right] \\ & \cdot \left[1 - \frac{(1 + a(F_r))^{q_r^L}}{(1 + a(F_u))^{q_r^L}} \right] \\ & + s \sum_{l=r+1}^{u} \prod_{f=l}^{n^O} \left(1 + a\left(F_f\right)\right)^{q_f^O} \cdot \left[1 - \frac{1}{(1 + a(F_u))^{q_r^L}} \right]. \end{aligned}$$

We have

$$1 - \frac{(1 + a(F_r))^{q_r^L}}{(1 + a(F_u))^{q_r^L}} > 0 \text{ and } 1 - \frac{1}{(1 + a(F_u))^{q_r^L}} > 0.$$

It can be derived that $C(O) > C\left(O'\right)$, which conflicts with the optimal schedule.

Case 2. $O = \left(W_1, F_u^o, W_2, F_r^o, W_3\right)$.

(1) when $q_r^L \geq q_u^O$, this situation is similar to (1) in case 1.

(2) when $q_r^L < q_u^O$, we can add all jobs of F_r^L following F_r^o and transfer the jobs of F_u^O with the job number of q_r^L into L, and then swap the updated $F_r^{O'}$ and $F_u^{O'}$. O is updated to $O' = \left(W_1, F_r^{O'}, W_2, F_u^{O'}, W_3\right)$, where $q_r^{O'} = q_r$ and $q_u^{O'} = q_u^O - q_r^L$. Thus,

$$\begin{aligned} C(O) - C\left(O'\right) = & \left[t_0 \prod_{l=1}^{n^O} (1 + a(F_l))^{q_l^O} + s \sum_{l=1}^{u} \prod_{f=l}^{n^O} \left(1 + a\left(F_f\right)\right)^{q_f^O} \right] \\ & \cdot \left[1 - \frac{(1 + a(F_r))^{q_r^L}}{(1 + a(F_u))^{q_r^L}} \right] \\ & + s \sum_{l=u+1}^{r} \prod_{f=l}^{n^O} \left(1 + a\left(F_f\right)\right)^{q_f^O} \cdot \left[1 - \frac{(1 + a(F_u))^{q_u^O - q_r^L}}{(1 + a(F_r))^{q_r^O}} \right]. \end{aligned}$$

Since $\rho(F_r) > \rho(F_u)$, that is, $(1 + a(F_r))^{q_r} > (1 + a(F_u))^{q_u}$, we have

$$(1 + a(F_u))^{q_u^O} \leq (1 + a(F_u))^{q_u} < (1 + a(F_r))^{q_r}.$$

Hence,

$$(1 + a(F_u))^{q_u^O - q_r^L} < (1 + a(F_r))^{q_r - q_r^L} = (1 + a(F_r))^{q_r^O}.$$

It can be derived that

$$1 - \frac{(1 + a(F_u))^{q_u^O - q_r^L}}{(1 + a(F_r))^{q_r^O}} > 0.$$

We have

$$1-\frac{(1+a(F_r))^{q_r^L}}{(1+a(F_u))^{q_r^L}}>0.$$

Therefore, we obtain that $C(O)>C\left(O'\right)$, which also conflicts with the optimal schedule.

Combining cases 1 and 2, there should be no jobs of F_r in L when some jobs from F_u are included in O. The proof is completed.

Based on the above lemmas, the following Algorithm 3 is designed to solve the problem $1|s-batch, p_i=a_it, s_{sd}, ac|\sum_{i=1}^{N}U_i$.

Algorithm 3

Step 1. Sequence the jobs of the same type together, and then sequence the job sets of all types in non-decreasing order of $a(F_l)$ such that $a(F_1)\le a(F_2)\le\cdots\le a(F_n)$. Set $k=1$ and $O=\{J_1\}$.

Step 2. Set $k=k+1$.

Step 3. Set $O=O\cup\{J_k\}$. Sequence the jobs of the same type together, and sequence the job sets of all types in non-increasing order of $a\left(F_l^O\right)$, where $\sigma\left(F_l^O\right)=\left(1+a\left(F_r^O\right)\right)^{q_l^O}$ such that $a\left(F_1^O\right)\ge a\left(F_2^O\right)\ge\cdots\ge a\left(F_{n^O}^O\right)$. Calculate the job completion time $C(O)$ as Eq. (2). If $C(O)>d$, then set $O=O\backslash\{J_k\}$, and go to step 5. Otherwise, go to step 4.

Step 4. If $k=N$, then go to step 5. Otherwise, go to step 2.

Step 5. Sequence the jobs of the same type together, and sequence the job sets of all types in non-increasing order of $a\left(F_l^O\right)$. Set $l=0$.

Step 6. Set $l=l+1$.

Step 7. If there are more than b jobs in F_l^O, then place the first b jobs in a batch and iterate. Otherwise, place the remaining jobs in a batch.

Step 8. If $l=n^O$, then stop and schedule the batches in their generation order at timet_0. Otherwise, go to step 6.

Theorem 3 *For the problem* $1|s-batch, p_i=a_it, s_{sd}, ac|\sum_{i=1}^{N}U_i$, *an optimal schedule can be obtained by Algorithm 3 in* $O\left(N^2\log N\right)$ *time.*

Proof Based on Lemmas 11–13, Algorithm 3 can generate an optimal solution. The time complexity of step 1 is $O(n\log n)$. Since the execution time of step 2 is at most N and the time complexity of step 3 is at most $O(n\log n)$, the total time complexity of steps 2, 3, and 4 is at most $O(nN\log n)$. Similarly, the time complexity of step 5 is $O(n\log n)$ and the total time complexity of steps 6, 7, and 8 is at most $O(nN)$. Then, we have $n\le N$. Thus, the time complexity of Algorithm 1 is at most $O\left(N^2\log N\right)$.

6 Problem $1|s-batch, p_i=a_it, s_{sd}|\sum_{i=1}^{N}C_i$

In this section, we focus on two special cases of minimizing the total completion times of all jobs, and these two special cases are $a(F_l)=a$ and $s=0$, respectively. The property is given for the general problem as follows.

Lemma 14 *For the problem* $1\,|s-batch,\ p_i=a_it,\ s_{sd}|\sum_{i=1}^{N}C_i$, *there is only one job in any batch.*

Proof If there exists a batch with $n_k \geq 2$, then $\sum_{i=1}^{N}C_i$ is reduced after we separate any job from this batch as a single batch. This completes the proof. □

Corollary 4 *For the problem* $1\,|s-batch,\ p_i=a_it,\ s_{sd}|\sum_{i=1}^{N}C_i$, *there are N batches in an optimal schedule.*

6.1 Special case: $a\,(F_l)=a$

In this subsection, we focus on the special case that the jobs of different types have the same deteriorating rate, i.e., $a\,(F_l)=a, l=1,2,\cdots,n$. We first propose some useful lemmas for this case, and then an optimization algorithm is developed to solve it based on these lemmas.

Lemma 15 *There exist two consecutive batches* b_k *and* b_{k+1} *of different job types in a schedule, with the starting time of* b_k *at time* $T>0$, *then*

$$\sum_{J_i\in b_k}C_i+\sum_{J_i\in b_{k+1}}C_i=\frac{1}{a}\left[(T+s)\,(1+a)^{n_k+n_{k+1}+1}-(T+s)\,(1+a)+s\,(1+a)^{n_{k+1}+1}-s\,(1+a)\right].$$

Proof Based on Lemma 14, we have

$$\sum_{J_i\in b_k}C_i=\frac{1}{a}\left[(T+s)\,(1+a)^{n_k+1}-(T+s)\,(1+a)\right]$$

and

$$\sum_{J_i\in b_{k+1}}C_i=\frac{1}{a}\left[(T+s)\,(1+a)^{n_k+n_{k+1}+1}-(T+s)\,(1+a)^{n_k+1}+s\,(1+a)^{n_{k+1}+1}-s\,(1+a)\right].$$

Then,

$$\sum_{J_i\in b_k}C_i+\sum_{J_i\in b_{k+1}}C_i=\frac{1}{a}\left[(T+s)\,(1+a)^{n_k+n_{k+1}+1}-(T+s)\,(1+a)+s\,(1+a)^{n_{k+1}+1}-s\,(1+a)\right].$$

The proof is completed. □

Lemma 16 *For the problem* $1\,|s-batch,\ p_i=a_it,\ s_{sd},\ a\,(F_l)=a|\sum_{i=1}^{N}C_i$, *the generated batches of the same job type are produced consecutively in an optimal schedule.*

Proof The same notations and the sequences π^* and π are used as in the proof of Lemma 4. Let T denote the starting time of b_p. Based on Lemma 15, we have

$$\sum_{i=\sum_{k=1}^{p-1}n_k+1}^{\sum_{k=1}^{q}n_k}C_i\,(\pi)-\sum_{i=\sum_{k=1}^{p-1}n_k+1}^{\sum_{k=1}^{q}n_k}C_i\,(\pi^*)$$

$$
\begin{aligned}
&= \sum_{i=\sum_{k=1}^{p} n_k+1}^{\sum_{k=1}^{q-1} n_k} C_i(\pi) + \sum_{i=\sum_{k=1}^{q-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi) - \sum_{i=\sum_{k=1}^{p} n_k+1}^{\sum_{k=1}^{q-1} n_k} C_i(\pi^*) - \sum_{i=\sum_{k=1}^{q-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi^*) \\
&< \frac{(T+s)(1+a)^{n_p+n_q}}{a} \cdot \left[(1+a)^{\sum_{k=p+1}^{q-1} n_k+1} - (1+a)\right] \\
&\quad + \frac{(T+s)(1+a)^{n_p}}{a} \cdot \left[(1+a)^{n_q+1} - (1+a)\right] \\
&\quad - \frac{(T+s)(1+a)^{n_p}}{a} \cdot \left[(1+a)^{\sum_{k=p+1}^{q-1} n_k+1} - (1+a)\right] \\
&\quad - \frac{(T+s)(1+a)^{\sum_{k=p}^{q-1} n_k}}{a} \cdot \left[(1+a)^{n_q+1} - (1+a)\right] \\
&= \left[(1+a)^{n_q} - 1\right] \frac{(T+s)(1+a)^{n_p}}{a} \cdot \left[(1+a)^{\sum_{k=p+1}^{q-1} n_k+1} - (1+a)\right] \\
&\quad + \left[1 - (1+a)^{\sum_{k=p+1}^{q-1} n_k}\right] \cdot \frac{(T+s)(1+a)^{n_p}}{a} \cdot \left[(1+a)^{n_q+1} - (1+a)\right] \\
&= \frac{(T+s)(1+a)^{n_p}}{a} \cdot \left[(1+a)^{n_q+1} - (1+a)\right] \\
&\quad \cdot \left[(1+a)^{\sum_{k=p+1}^{q-1} n_k} - 1 + 1 - (1+a)^{\sum_{k=p+1}^{q-1} n_k}\right] = 0.
\end{aligned}
$$

Hence,

$$
\sum_{i=\sum_{k=1}^{p-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi) < \sum_{i=\sum_{k=1}^{p-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi^*).
$$

Based on Lemma 4, it can be deduced that

$$
\sum_{i=\sum_{k=1}^{q} n_k+1}^{N} C_i(\pi) < \sum_{i=\sum_{k=1}^{q} n_k+1}^{N} C_i(\pi^*).
$$

and

$$
\sum_{i=1}^{\sum_{k=1}^{p-1} n_k} C_i(\pi) = \sum_{i=1}^{\sum_{k=1}^{p-1} n_k} C_i(\pi^*).
$$

Thus,

$$
\begin{aligned}
&\sum_{i=1}^{N} C_i(\pi) - \sum_{i=1}^{N} C_i(\pi^*) \\
&= \sum_{i=1}^{\sum_{k=1}^{p-1} n_k} C_i(\pi) + \sum_{i=\sum_{k=1}^{p-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi) + \sum_{i=\sum_{k=1}^{q} n_k+1}^{N} C_i(\pi) \\
&\quad - \left[\sum_{i=1}^{\sum_{k=1}^{p-1} n_k} C_i(\pi^*) + \sum_{i=\sum_{k=1}^{p-1} n_k+1}^{\sum_{k=1}^{q} n_k} C_i(\pi^*) + \sum_{i=\sum_{k=1}^{q} n_k+1}^{N} C_i(\pi^*)\right] < 0,
\end{aligned}
$$

which conflicts with the optimal schedule. This completes the proof. □

Based on Lemma 16, we have the following property.

Lemma 17 *For the problem* $1\,|s-batch, p_i = a_i t, s_{sd}, a\,(F_l) = a|\sum_{i=1}^{N} C_i$, *if* $q_r \geq q_{r+1}$ *for two consecutive job sets of job types* r *and* $r+1$, $r = 1, 2, \cdots, n-1$, *then it is optimal to process* F_r *before* F_{r+1}. □

Proof We use the same notations and the sequences π^* and π as in the proof of Lemma 6. We assume that there exists $q_r < q_{r+1}$ in π^*. Let the starting time of F_r in π^* be T.

For π^*, we have

$$\sum_{i=\sum_{l=1}^{r-1} q_l+1}^{\sum_{l=1}^{r+1} q_l} C_i\left(\pi^*\right) = (T+s)(1+a)\cdot\frac{(1+a)^{q_r}-1}{a} + \left[(T+s)(1+a)^{q_r}+s\right](1+a)\cdot\frac{(1+a)^{q_{r+1}}-1}{a}$$

And for π,

$$\sum_{i=\sum_{l=1}^{r-1} q_l+1}^{\sum_{l=1}^{r+1} q_l} C_i\left(\pi\right) = (T+s)(1+a)\cdot\frac{(1+a)^{q_{r+1}}-1}{a} + \left[(T+s)(1+a)^{q_{r+1}}+s\right](1+a)\cdot\frac{(1+a)^{q_r}-1}{a}.$$

Then,

$$\begin{aligned}
&\sum_{i=\sum_{l=1}^{r-1} q_l+1}^{\sum_{l=1}^{r+1} q_l} C_i\left(\pi^*\right) - \sum_{i=\sum_{l=1}^{r-1} q_l+1}^{\sum_{l=1}^{r+1} q_l} C_i\left(\pi\right)\\
&= (T+s)(1+a)\cdot\frac{(1+a)^{q_r}-1}{a} + \left[(T+s)(1+a)^{q_r}+s\right](1+a)\cdot\frac{(1+a)^{q_{r+1}}-1}{a}\\
&\quad - (T+s)(1+a)\cdot\frac{(1+a)^{q_{r+1}}-1}{a} - \left[(T+s)(1+a)^{q_{r+1}}+s\right](1+a)\cdot\frac{(1+a)^{q_r}-1}{a}\\
&= \frac{s(1+a)}{a}\left[(1+a)^{q_{r+1}}-(1+a)^{q_r}\right] > 0.
\end{aligned}$$

Based on Lemma 6, it can be derived that

$$\sum_{i=\sum_{l=1}^{r+1} q_l+1}^{\sum_{l=1}^{n} q_l} C_i\left(\pi^*\right) > \sum_{i=\sum_{l=1}^{r+1} q_l+1}^{\sum_{l=1}^{n} q_l} C_i\left(\pi\right).$$

And we have

$$\sum_{i=1}^{\sum_{l=1}^{r-1} q_l} C_i\left(\pi^*\right) = \sum_{i=1}^{\sum_{l=1}^{r-1} q_l} C_i\left(\pi\right).$$

Thus,

$$\sum_{i=1}^{N} C_i\left(\pi^*\right) > \sum_{i=1}^{N} C_i\left(\pi\right),$$

which conflicts with the optimal schedule. This completes the proof. □

Based on the above lemmas, we develop the following Algorithm 4 to solve the first special case.

Algorithm 4

Step 1.	Sequence the jobs of the same type together, and then sequence the job sets of all types in non-increasing order of q_l, i.e., $q_1 \geq q_2 \geq \cdots \geq q_n$. Set $l = 0$.
Step 2.	Set $l = l + 1$.
Step 3.	If there is any job in F_l, then place the first job as a batch and iterate. Otherwise, go to step 4.
Step 4.	If $l = n$, then stop and schedule the batches in their generation order at time t_0. Otherwise, go to step 3.

Thus, Algorithms 1 and 4 have some similarities, where the jobs of the same type are sequenced together and the same way of job batching is conducted in both algorithms.

Theorem 4 *For the problem* $1\,|s-batch, p_i = a_i t, s_{sd}, a\,(F_l) = a|\sum_{i=1}^{N} C_i$, *an optimal schedule can be obtained by Algorithm 4 in* $O\,(N \log N)$ *time. If the job sets of all types are sequenced in non-increasing order of* q_l $(l = 1, 2, \cdots, n)$, *then the optimal sum of job completion times is*

$$\sum_{i=1}^{N} C_i = (t_0 + s)\,\frac{(1+a)^{N+1} - (1+a)}{a} + s\left[\sum_{j=3}^{n}\sum_{l=2}^{j-1}(1+a)\,\frac{(1+a)^{q_j} - 1}{a}\prod_{x=l}^{j-1}(1+a)^{q_x} + \sum_{j=2}^{n} s\,(1+a)\,\frac{(1+a)^{q_j} - 1}{a}\right] \tag{4}$$

Proof Based on Lemmas 15–17, an optimal solution can be generated by Algorithm 4. It is seen that the time complexity of step 1 is $O\,(n \log n)$ to obtain the optimal job set sequence of all types, and the total time complexity of steps 2, 3, and 4 is $O\,(n)$. Then, we have $n \leq N$. Thus, the time complexity of Algorithm 4 is at most $O\,(N \log N)$.

Mathematical induction can be used to prove the optimal result of sum of job completion times based on the number of job types. Firstly for $n = 1$ we have

$$\sum_{i=1}^{q_1} C_i = (t_0 + s)\,\frac{(1+a)^{q_1+1} - (1+a)}{a},$$

so Eq. (4) holds for $n = 1$. Suppose that for all $2 \leq p \leq n-1$, Eq. (4) is satisfied. We have

$$\sum_{i=1}^{\sum_{f=1}^{p} q_f} C_i = (t_0 + s)\,\frac{(1+a)^{\sum_{f=1}^{p} q_f + 1} - (1+a)}{a} + s\left[\sum_{j=3}^{p}\sum_{l=2}^{j-1}(1+a)\,\frac{(1+a)^{q_j} - 1}{a}\prod_{x=l}^{j-1}(1+a)^{q_x} + \sum_{j=2}^{p}(1+a)\,\frac{(1+a)^{q_j} - 1}{a}\right]$$

Then, for the $(p+1)$th job set F_{p+1},

$$\begin{aligned}
\sum_{i=1}^{\sum_{f=1}^{p+1} q_f} C_i &= \sum_{i=1}^{\sum_{f=1}^{p} q_f} C_i + \sum_{i=\sum_{f=1}^{p} q_f+1}^{\sum_{f=1}^{p+1} q_f} C_i \\
&= \sum_{i=1}^{\sum_{f=1}^{p} q_f} C_i + \left[(t_0+s)\,(1+a)^{\sum_{f=1}^{p} q_f} + \sum_{j=2}^{p} s\,(1+a)^{\sum_{f=j}^{p} q_f} + s \right] (1+a) \\
&\quad \frac{(1+a)^{q_{p+1}}-1}{a} \\
&= (t_0+s)\,\frac{(1+a)^{\sum_{f=1}^{p+1} q_f+1}-(1+a)}{a} \\
&\quad + s\left[\sum_{j=3}^{p+1} \sum_{l=2}^{j-1} (1+a)\,\frac{(1+a)^{q_j}-1}{a} \prod_{x=l}^{j-1} (1+a)^{q_x} + \sum_{j=2}^{p+1} (1+a)\,\frac{(1+a)^{q_j}-1}{a} \right].
\end{aligned}$$

Note that $\sum_{i=1}^{N} C_i = \sum_{i=1}^{\sum_{f=1}^{n} q_f} C_i$, the result is proved by the induction. □

6.2 Special case: $s = 0$

Lemma 18 *For the problem* $1\,|s-batch, p_i = a_i t, s=0|\sum_{i=1}^{N} C_i$, *there is an optimal schedule, where the batches of the same job type are processed together, and all jobs are sequenced in non-decreasing order of* a_i. □

Proof For any given schedule, the makespan remains unchanged after the batches of the same job type are processed together. Hence, the batches of the same job type can be processed together in an optimal schedule. Let π^* and π be an optimal schedule and a job schedule. The difference between the two schedules is the pairwise interchange of two job batches $\{J_x\}$ and $\{J_{x+1}\}$ $(x = 1, 2, \cdots, m-1)$, that is, $\pi^* = (W_1, \{J_x\}, \{J_{x+1}\}, W_2)$, $\pi = (W_1, \{J_{x+1}\}, \{J_x\}, W_2)$, where J_x and J_{x+1} are of different job types l and f, $a(F_l) > a(F_f)$, $W_1$and W_2 represent two partial sequences, and W_1 or W_2 may be empty. Hence, we have $a_x > a_{x+1}$. Then,

$$\sum_{i=1}^{N} C_i(\pi^*) - \sum_{i=1}^{N} C_i(\pi) = \sum_{i=1}^{x+1} C_i(\pi^*) - \sum_{i=1}^{x+1} C_i(\pi) = t_0\,(a_x - a_{x+1}) \prod_{i=1}^{x-1} (1+a_i) > 0,$$

Which conflicts with the optimal schedule, and this completes the proof. □

Based on the above lemma, we develop the following Algorithm 5 to solve the second special case.

Theorem 5 *For the problem* $1\,|s-batch, p_i = a_i t, s=0|\sum_{i=1}^{N} C_i$, *an optimal schedule can be obtained by Algorithm 5 in* $O(N \log N)$ *time. If the job sets of all types are sequenced in non-decreasing order of a* (F_l) $(l = 1, 2, \cdots, n)$, *then the optimal total completion times of all jobs are* $t_0 \sum_{i=1}^{N} \prod_{j=1}^{i} (1+a_j)$.

Proof Based on Lemma 18, Algorithm 5 can obtain an optimal solution. It is obvious that the time complexity of Algorithm 5 is the same as that of Algorithm 4, so the time complexity of Algorithm 5 is also $O(N \log N)$. The result of total completion times of all jobs can be

Algorithm 5

Step 1.	Sequence the jobs of the same type together, and then sequence the job sets of all types in non-decreasing order of $a(F_l)$, i.e., $a(F_1) \leq a(F_2) \leq \cdots \leq a(F_n)$. Set $l = 0$.
Step 2.	Set $l = l + 1$.
Step 3.	If there is any job in F_l, then place the first job as a batch and iterate. Otherwise, go to step 4
Step 4.	If $l = n$, then stop and schedule the batches in their generation order at timet_0. Otherwise, go to step 3.

proved by induction. The proof is similar to that of Lemma 1, and it is omitted here. The proof is completed. □

7 Conclusions

In this paper, we study a new serial-batching scheduling model with deteriorating jobs, where multiple job types and sequence-dependent setup times are considered simultaneously. For the problems of minimizing the makespan, the maximum tardiness, the maximum lateness, and the maximum earliness, we develop optimization algorithms to solve them respectively. Moreover, an optimization algorithm is proposed to minimize the number of tardy jobs under a certain agreeable condition. Finally, for the problem of minimizing the total completion time of jobs, we also analyze its two special cases and propose optimization algorithms to solve them respectively. The algorithm can be applied in actual practices to increase the productivity. In future research, we may focus on developing effective meta-heuristic to solve some related scheduling problems, considering other objective functions, and proposing more general and practical models.

Acknowledgements We are grateful to the editors and the anonymous reviewers for their helpful suggestions on an earlier version of this paper. This work is supported by the National Natural Science Foundation of China (Nos. 71231004, 71171071, 71131002), and Panos M. Pardalos is partially supported by LATNA laboratory, NRU HSE, RF government grant, ag. 11.G34.31.0057.

References

Agnetis, A., Detti, P., Meloni, C., & Pacciarelli, D. (2001). Set-up coordination between two stages of a supply chain. *Annals of Operation Research*, *107*, 15–32.

Bai, J., Li, Z. R., & Huang, X. (2012). Single-machine group scheduling with general deterioration and learning effects. *Applied Mathematical Modelling*, *36*, 1267–1274.

Browne, S., & Yechiali, U. (1990). Scheduling deteriorating jobs on a single processor. *Operations Research*, *38*, 495–498.

Cheng, T. C. E., Ding, Q., & Lin, B. M. T. (2004). A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, *152*, 1–13.

Detti, P., Meloni, C., & Pranzo, M. (2007). Minimizing and balancing setups in a serial production system. *International Journal of Production Research*, *45*, 5769–5788.

Gawiejnowicz, S. (2008). *Time-dependent scheduling, monographs in theoretical computer science*. Berlin-Heidelberg: An EATCS Series, Springer.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, *5*, 287–326.

Gupta, J. N. D., & Gupta, S. K. (1988). Single facility scheduling with nonlinear processing times. *Computers and Industrial Engineering*, *14*, 387–393.

Huang, X., Wang, M. Z., & Wang, J. B. (2011). Single-machine group scheduling with both learning effects and deteriorating jobs. *Computers and Industrial Engineering*, *60*, 750–754.
Huang, X., Li, G., Huo, Y., & Ji, P. (2013). Single machine scheduling with general time-dependent deterioration, position-dependent learning and past-sequence-dependent setup times. *Optimization Letters*, *7*, 1793–1804.
Lai, P. J., Lee, W. C., & Chen, H. H. (2011). Scheduling with deteriorating jobs and past-sequence-dependent setup times. *International Journal of Advanced Manufacturing Technology*, *54*, 737–741.
Lee, W. C., & Lu, Z. S. (2012). Group scheduling with deteriorating jobs to minimize the total weighted number of late jobs. *Applied Mathematics and Computation*, *218*, 8750–8757.
Lee, W. C. (2014). Single-machine scheduling with past-sequence-dependent setup times and general effects of deterioration and learning. *Optimization Letters*, *8*, 135–144.
Li, S. S., Ng, C. T., Cheng, T. C. E., & Yuan, J. J. (2011). Parallel-batch scheduling of deteriorating jobs with release dates to minimize the makespan. *European Journal of Operational Research*, *210*, 482–488.
Liao, C. J., Tsai, Y. L., & Chao, C. W. (2011). An ant colony optimization algorithm for setup coordination in a two-stage production system. *Applied Soft Computing*, *11*, 4521–4529.
Miao, C. X., Zhang, Y. Z., & Cao, Z. G. (2011). Bounded parallel-batch scheduling on single and multi machines for deteriorating jobs. *Information Processing Letters*, *111*, 798–803.
Miao, C. X., Zhang, Y. Z., & Wu, C. L. (2012). Scheduling of deteriorating jobs with release dates to minimize the maximum lateness. *Theoretical Computer Science*, *462*, 80–87.
Pei, J., Liu, X., Pardalos, P. M., Fan, W., & Yang, S. (2015). Single machine serial-batching scheduling with independent setup time and deteriorating job processing times. *Optimization Letters, 9,* 91–104.
Qi, X. L., Zhou, S. G., & Yuan, J. J. (2009). Single machine parallel-batch scheduling with deteriorating jobs. *Theoretical Computer Science*, *410*, 830–836.
Wang, J. B., Lin, L., & Shan, F. (2008). Single-machine group scheduling problems with deteriorating jobs. *International Journal of Advanced Manufacturing Technology*, *39*, 808–812.
Wang, J. B., Gao, W. J., Wang, L. Y., & Wang, D. (2009). Single machine group scheduling with general linear deterioration to minimize the makespan. *International Journal of Advanced Manufacturing Technology*, *43*, 146–150.
Wang, J. B., & Sun, L. Y. (2010). Single-machine group scheduling with linearly decreasing time-dependent setup times and job processing times. *International Journal of Advanced Manufacturing Technology*, *49*, 765–772.
Wang, J. B., Huang, X., Wu, Y. B., & Ji, P. (2012). Group scheduling with independent setup times, ready times, and deteriorating job processing times. *International Journal of Advanced Manufacturing Technology*, *60*, 643–649.
Wang, D., Huo, Y. Z., & Ji, P. (2014). Single-machine group scheduling with deteriorating jobs and allotted resource. *Optimization Letters*, *8*, 591–605.
Wu, C. C., Shiau, Y. R., & Lee, W. C. (2008). Single-machine group scheduling problems with deterioration consideration. *Computers and Operations Research*, *35*, 1652–1659.
Xuan, H., & Tang, L. X. (2007). Scheduling a hybrid flowshop with batch production at the last stage. *Computers and Operations Research*, *34*, 2718–2733.
Yang, S. J. (2011). Group scheduling problems with simultaneous considerations of learning and deterioration effects on a single-machine. *Applied Mathematical Modelling*, *35*, 4008–4016.
Yin, Y., Cheng, S. R., & Wu, C. C. (2012). Scheduling problems with two agents and a linear non-increasing deterioration to minimize earliness penalties. *Information Sciences*, *189*, 282–292.
Zhang, X., Yan, G., Huang, W., & Tang, G. (2001). Single-machine scheduling problems with time and position dependent processing times. *Annals of Operations Research*, *186*, 345–356.