Ann Oper Res (2019) 272:217–241
https://doi.org/10.1007/s10479-017-2481-8

Single-machine and parallel-machine serial-batching scheduling problems with position-based learning effect and linear setup time

Jun Pei[1,2,3] · **Bayi Cheng**[1,3] · **Xinbao Liu**[1,3] · **Panos M. Pardalos**[2] · **Min Kong**[1,3]

Published online: 4 April 2017

Abstract This paper introduces the serial batching scheduling problems with position-based learning effect, where the actual job processing time is a function of its position. Two scheduling problems respectively for single-machine and parallel-machine are studied, and in each problem the objectives of minimizing maximum earliness and total number of tardy jobs are both considered respectively. In the proposed scheduling models, all jobs are first partitioned into serial batches, and then all batches are processed on the serial-batching machine. We take some practical production features into consideration, i.e., setup time before processing each batch increases with the time, regarded as time-dependent setup time, and we formalize it as a linear function of its starting time. Under the single-machine scheduling setting, structural properties are derived for the problems with the objectives of minimizing maximum earliness and number of tardy jobs respectively, based on which optimization algorithms are developed to solve them. Under the parallel-machine scheduling setting, a hybrid VNS–GSA algorithm combining variable neighborhood search (VNS) and gravitational search algorithm (GSA) is proposed to solve the problems with these two objectives respectively, and the effectiveness and efficiency of the proposed VNS–GSA are demonstrated and compared with the algorithms of GSA, VNS, and simulated annealing (SA). This paper demonstrates that the consideration of different objectives leads to various optimal decisions on jobs assignment, jobs batching, and batches sequencing, which generates a new insight to investigate batching scheduling problems with learning effect under single-machine and parallel-machine settings.

✉ Jun Pei
feiyijun.ufl@gmail.com

✉ Xinbao Liu
lxb@hfut.edu.cn

1 School of Management, Hefei University of Technology, Hefei, China

2 Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL, USA

3 Key Laboratory of Process Optimization and Intelligent Decision-Making of Ministry of Education, Hefei, China

Keywords Scheduling · Serial-batching · Learning effect · Single-machine · Parallel-machine · Setup time

1 Introduction

In most scheduling literature, the processing time of jobs was often assumed to be fixed and known before the processing (Pinedo 2002). However, in many practical environments, the workers can acquire experience and the production efficiency improves continuously with time after they repetitiously operate the same or similar tasks. As a result, the later a job is processed, the shorter its processing time. This phenomenon is so-called the "learning effect" in the literature (Biskup 1999). Biskup (1999) was the pioneer introducing the learning effect into scheduling problems, where a position-dependent learning model was developed, and since then the learning effect has received extensive discussion in the scheduling field (Mosheiov and Sidney 2003; Janiak and Rudek 2009; Teyarachakul et al. 2011). In particular, much research on the batch scheduling problems with the learning effect can be found in the last decade (Yang and Kuo 2009; Kumar and Tan 2015; Paul et al. 2015; Tan et al. 2016; Tan and Carrillo 2017). However, all these studies focus on the parallel-batching scheduling problems. Whereas the other major type of batch scheduling problems, the serial-batching problems, with learning effect are relatively unexplored.

Our study was motivated by a practical production scenario of the aluminum-making process in an aluminum plant. In this particular scenario, cylindrical aluminum ingots are processed on an extrusion machine, which is a batch processing machine that processes batches of aluminum ingots for the extrusion process in the form of serial-batching, i.e., one after another in the same batch. Before processing each batch of aluminum ingots, a time-dependent linear setup time is required. The production efficiency improves continuously over time, and the processing time of an aluminum ingot depends on its processing position or sequence in a batch. That is, if an aluminum ingot is processed in a later position, then it takes less time to complete the extrusion process. Therefore, the processing times of aluminum ingots can be modeled as a decreasing function of their positions. In this production scenario, both single-machine and parallel-machine configurations are required for different production demands. Thus, in this paper we focus on both single-machine and parallel-machine scheduling situations according to the actual production requirements.

There have been many studies on the problems of single-machine scheduling with the learning effect. Cheng and Wang (2000) studied a single-machine scheduling problem where the job processing times decrease following a volume-dependent piecewise linear learning function, and the objective was to minimize the maximum lateness. Wang et al. (2008) considered a single-machine scheduling problem with the time-dependent learning effect, in which a job's processing time is assumed to be a function of the total normal processing time of all the other jobs scheduled in front of the job, and the objectives were to minimize the weighted sum of completion times, the maximum lateness, and the number of tardy jobs. Lee et al. (2010) investigated a single-machine problem with the learning effect and release times, and the objective was to minimize the makespan. A branch-and-bound algorithm was developed to derive the optimal solution. More recent papers which have addressed single-machine scheduling problems with learning effect include Lu et al. (2012), Zhu et al. (2013), Wang et al. (2014a), Wang and Wang (2014).

In addition to single-machine scheduling problem, some researchers also focus on the parallel-machines scheduling with the learning effect. Mosheiov (2001) studied parallel-machine scheduling problems flow-time minimization with a learning effect, and the objective was to minimize the flow-time. Eren (2009) investigated the bicriteria parallel machine scheduling problem with a learning effect of setup and removal times, and the objective function of the problem was to minimize of the weighted sum of total completion time and total tardiness. Three heuristic approaches were proposed to solve large-size problems. Eren and Güner (2009) also studied m-parallel machine scheduling problem for minimizing the weighted sum of total completion time and total tardiness based on the learning effect. Some other works on parallel-machine scheduling with learning effect can be found in Hsu et al. (2011), Okołowski and Gawiejnowicz (2010), etc. Although the scheduling problems under single-machine and parallel-machine settings have been separately studied, these two settings are rarely considered simultaneously. Meanwhile, the previous studies mainly focused on the general processing jobs way, and few papers studied the special processing way, e.g., serial-batch processing according to the practical production scenario.

The group scheduling problems with the learning effect are similar to the problem studied in this paper. In the group scheduling problems, the jobs are classified into certain groups in advance based on similar production requirements. Yang and Chard (2008) studied a group scheduling problem with learning and forgetting effects, and the objective was to minimize the total completion time of jobs. Three basic models were developed and some comparisons were made through computational experiments to analyze the impact of effects on learning and forgetting on this problem. Pan et al. (2014) studied an integrated single machine group scheduling problem, where the effects of learning and forgetting and preventive maintenance planning were simultaneously considered. More recent papers on the group scheduling problems with the learning effect include Bai et al. (2012), Huang et al. (2011), and Wang et al. (2012, 2014b), etc. Although some similarity can be found 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 several significant differences between them, as concluded in Pei et al. (2015a).

We have also studied the serial-batching scheduling problems with learning effect in the past research, while many key significant differences can be found in them. Different from Pei et al. (2017a), our paper only focuses on the learning effect, different argument jobs batching and jobs sequencing are obtained, and parallel-machine scheduling problems are studied and the hybrid VNS–GSA is proposed to solve them. Different from Pei et al. (2016), the machines are available all the time in this paper, and different learning effects are investigated. In addition, we also investigated the scheduling problems with the deteriorating jobs in the past research (Pei et al. 2015a, b, 2017b), and the deterioration phenomenon is also a typical characteristic during the practical production. We compare and contrast the related papers with our work in the following Table 1.

The main contributions of this paper can be summarized as follows:

(1) Both single-machine and parallel-machine scheduling problems with the learning effect using the processing way of serial-batching are studied, and the objectives of minimizing maximum earliness and total number of tardy jobs are considered in these two types of scheduling problems. The integer programming models for all the studied problems are constructed.
(2) For the single-machine scheduling problems, structural properties are derived for the problems of minimizing maximum earliness and number of tardy jobs, and optimization algorithms are developed to solve them.
(3) Based on the derived structural properties of the single-machine scheduling problems, the hybrid VNS–GSA algorithm is proposed to solve the parallel-machine scheduling problems with these two objectives.

Table 1 Key comparisons on our previous paper, other related research, and current study

Authors/year	Effect	Objective	Setup	Machine	Jobs processing	Algorithm
Pei et al. (2017a)	DE, LE	C_{max}	TD	1	Serial-batching	Heuristic algorithm
Pei et al. (2016)	LE	C_{max}	TD	1	Serial-batching	Heuristic algorithm
Pei et al. (2015a, b, 2017b)	DE	C_{max}	TD	1	Serial-batching	Heuristic algorithm
Cheng and Wang (2000)	LE	L_{max}	–	1	General processing	Polynomial approximation scheme
Wang et al. (2008)	LE	$\sum_{i=1}^{n} w_i C_i, \quad \sum_{i=1}^{n} U_i$	–	1	General processing	Polynomial time algorithm
Lee et al. (2010)	LE	C_{max}	–	1	General processing	Branch-and-bound algorithm
Wang et al. (2014a)	LE	$\sum_{i=1}^{n} (\alpha E_i + \beta T_i + \gamma d_i)$	–	1	General processing	Polynomial time algorithm
Eren (2009)	LE	$\sum_{i=1}^{n} (C_i + T_i)$	–	≥ 2	General processing	Heuristic algorithm
Eren and Güner (2009)	LE	$\alpha \sum_{i=1}^{n} C_i + \beta \sum_{i=1}^{n} T_i$	–	≥ 2	General processing	–
Hsu et al. (2011)	LE	$\sum_{i=1}^{n} C_i$	–	≥ 2	General processing	Heuristic algorithm
Current study	*LE*	$\sum_{i=1}^{n} U_i, \quad E_{max}$	*TD*	1; ≥ 2	*Serial-batching*	*Heuristic algorithm, Hybrid VNS–GSA*

DE and LE denote deteriorating effect and learning effect, TD denotes time-dependent

The reminder of this paper is organized as follows. The notation and problem description are given in Sect. 2. The single-machine and parallel-machine scheduling problems are studied in Sects. 3 and 4, respectively. Finally, the conclusion is given in Sect. 5.

2 Notation and problem definition

The notation used in this paper is first described as Table 2.

There is a given set of n non-preemptive jobs to be processed at time zero. Two types of scheduling problems are studied. In the first type of problem, all jobs are processed

Table 2 The list of the notation

n	The number of jobs
q	The number of machines in the parallel-machine scheduling problems
M_g	Machine g, $f = 1, 2, \ldots, q$
J_i	Job i, $i = 1, 2, \ldots, n$
p_i	The basic processing time of J_i, $i = 1, 2, \ldots, n$
a	The learning index of processing time
$[\,]$	The symbol which denotes the job order in a schedule
$p_{[r]}$	The actual processing time of a job assigned to the position r, $r = 1, 2, \ldots, n$
x_{ir} $\left(x_{gir}\right)$	1, if J_i is assigned to the position r (on M_g), $i, r = 1, 2, \ldots, n$, $g = 1, 2, \ldots, q$; 0, otherwise
p_{ir} $\left(p_{gir}\right)$	The actual processing time of J_i assigned to the position r (on M_g), $i, r = 1, 2, \ldots, n$, $g = 1, 2, \ldots, q$
m	The total number of batches
m_g	The number of batches on M_g, $g = 1, 2, \ldots, q$
b_k $\left(b_{gk}\right)$	Batch k (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$
n_k $\left(n_{gk}\right)$	The number of jobs in b_k (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$
$S\left(b_k\right)$ $\left(S\left(b_{gk}\right)\right)$	The starting time of b_k (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$
$C\left(b_k\right)$ $\left(C\left(b_{gk}\right)\right)$	The completion time of b_k (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$
θ	The deteriorating rate of the setup time
s_k $\left(s_{gk}\right)$	The setup time of b_k (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$
d	The common due date of all jobs
c	The capacity of the serial-batching machine, i.e., the maximum number of jobs in a batch
π	A schedule of n jobs
$C_i\left(\pi\right)$	The completion time of J_i in a given schedule π, $i = 1, 2, \ldots, n$
y_{ik} $\left(y_{gik}\right)$	1, if J_i is assigned to b_k $\left(b_{fk}\right)$ (on M_g), $i = 1, 2, \ldots, n$, $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$; 0, otherwise
z_{kl} $\left(z_{gkl}\right)$	1, if b_k $\left(b_{fk}\right)$ precedes b_l $\left(b_{fk}\right)$ (on M_g), $k = 1, 2, \ldots, m$, $g = 1, 2, \ldots, q$; 0, otherwise
$\sum_{i=1}^{n} U_i \left(\sum_{i=1}^{n} U_i^g\right)$	The number of tardy jobs (on M_g), $g = 1, 2, \ldots, q$
E_{max} $\left(E_{max}^g\right)$	The maximum earliness of all jobs (on M_g), $g = 1, 2, \ldots, q$

on a single serial-batching machine, while in the second one, all jobs are first assigned into parallel machines, and then the assigned jobs are processed on each machine with serial-batching processing way. For both single-machine and parallel-machine scheduling problems, the objectives to minimize maximum earliness and total number of tardy jobs are considered, respectively. The jobs are firstly partitioned into multiple serial batches on each serial-batching machine and then they are processed on the serial-batching machine. Serial batches require that all the jobs within the same batch are processed consecutively in a serial fashion (Xuan and Tang 2007), and the completion times of all jobs are equal to that of their belonged batch, which is defined as the completion time of the last job in that batch. The capacity of the serial-batching machine is defined as c, that is, the maximum number of jobs in a batch is equal to c, and each machine has the same capacity c in the parallel-machine scheduling problems. A time-dependent setup time is required before processing each batch. In this paper, we extend the application of Biskup's model (Biskup 1999) in the serial-batching scheduling problem. The learning effect is reflected on the varying job processing time. Specifically, the actual processing time of J_i is defined as

$$p_{ir} = p_{[r]} = p_i r^a, \quad i, r = 1, 2, \ldots, n$$

where r is the position of J_i, and a is the learning index of processing time and $a \leq 0$.

Moreover, as in Cheng et al. (2011), the batch's setup time is also defined as a simple linear function of its starting time t, that is,

$$s = \theta t$$

where $\theta > 0$ is the setup time's deterioration rate.

We adopt $E_{max} = \max\limits_{i=1,2,\ldots,n} \{0, d - C_i\}$ and $\sum_{i=1}^{n} U_i$ to denote the maximum earliness and total number of tardy jobs, respectively. All jobs have a common due date, where $U_i = 1$ if $C_i > d$ and 0 otherwise. In the remaining sections of the paper, we use the three-field notation schema $\alpha \left|\beta\right| \gamma$ introduced by Graham et al. (1979) to denote all the problems.

3 Single-machine scheduling problems

In this section, we firstly derive the completion time of each batch, which will be used in the problems with the objectives of minimizing maximum earliness and total number of tardy jobs. Then, the problems with the objectives of minimizing maximum earliness and total number of tardy jobs are studied in Sects. 3.1 and 3.2, respectively.

Lemma 1 *For any given schedule $\pi = \left(b_1, b_2, \ldots, b_f, \ldots, b_m\right)$, the completion time of b_f in schedule π is*

$$C\left(b_f\right) = \sum_{k=1}^{f} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} \left(1+\theta\right)^{f-k} \tag{3.1}$$

where $f = 1, 2, \ldots, m$.

Proof This lemma can be proved by the mathematical induction based on the number of batches. Firstly for $f = 1$, we have

$$C\left(b_1\right) = s_1 + \sum_{r=1}^{n_1} p_{[r]} = \sum_{r=1}^{n_1} \sum_{i=1}^{n} p_i i^a x_{ir}.$$

Thus, Eq. (3.1) holds for $f = 1$. Suppose for all $2 \leq f \leq m-1$, Eq. (3.1) is satisfied. We have

$$C\left(b_f\right) = \sum_{k=1}^{f} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{f-k}.$$

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

$$\begin{aligned} C\left(b_{f+1}\right) &= C\left(b_f\right) + s_{f+1} + \sum_{r=1+\sum_{j=1}^{f} n_j}^{\sum_{j=1}^{f+1} n_j} p_{[r]} \\ &= \sum_{k=1}^{f} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{f-k} \\ &\quad + \theta \sum_{k=1}^{f} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{f-k} \\ &\quad + \sum_{r=1+\sum_{j=1}^{f} n_j}^{\sum_{j=1}^{f+1} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} \\ &= \sum_{k=1}^{f+1} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{f+1-k}. \end{aligned}$$

Thus, Eq. (3.1) holds for $C\left(b_{l+1}\right)$, and the lemma is proved by the induction. □

3.1 Problem $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\, E_{max}$

In this section, we study the single-machine scheduling problem with the objective of minimizing the maximum earliness of all jobs. For this problem, all jobs are restricted to be completed no later than the common due date d, which is usually assumed in many previous works involving earliness (Yin et al. 2012), and otherwise if without this assumption each job may be trivially scheduled sufficiently late to avoid earliness cost. Hence, it should be $d \geq t + \sum_{k=1}^{m} s_k + \sum_{i=1}^{n} p_i$, where t is the starting time of processing jobs. We denote the earliness of job J_i and the maximum earliness of all jobs as $E_i = max\{0, d - C_i\}$ and $E_{max} = \max_{i=1,2,\ldots,n} E_i$, respectively. Next, the model is given in Sect. 3.1.1, and then some structural properties are derived and an optimization algorithm is proposed to solve the problem based on the properties.

3.1.1 Mixed integer programming model for the problem $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\, E_{max}$

$$\text{Minimize } E_{max} \tag{3.2}$$

Subject to:

$$\sum_{i=1}^{n} y_{ik} \le c, \quad k=1,2,\ldots,m \tag{3.3}$$

$$\sum_{k=1}^{m} y_{ik} = 1, \quad i=1,2,\ldots,n \tag{3.4}$$

$$S\left(b_{k+1}\right) \ge C\left(b_k\right)(1+\theta), \quad k=1,2,\ldots,m-1 \tag{3.5}$$

$$d \ge t+\sum_{k=1}^{m} s_k+\sum_{i=1}^{n} p_i, \tag{3.6}$$

$$C\left(b_k\right)-C\left(b_l\right)+Mz_{kl}-C\left(b_k\right)(1+\theta)-P\left(b_k\right) \ge 0, \quad k,l=1,2,\ldots,m \tag{3.7}$$

$$y_{ik}, z_{kl}=0\, or\, 1, \quad k,l=1,2,\ldots,m_i, i,p=1,2,\ldots,n \tag{3.8}$$

The objective function (3.2) is to minimize the maximum earliness of all jobs. Constraint set (3.3) prohibits that the job number of any batch more than the machine capacity. Constraint set (3.3) guarantees that each job should be only assigned to one batch. Constraint set (3.4) makes sure that setup time is required before processing each batch. Constraint sets (3.6) ensures that all jobs are restricted to be completed no later than the common due date d in this studied problem. Constraint set (3.7) indicates that there is no overlapping situation between two continuous batches. Constraint set (3.8) defines the ranges of the variables.

3.1.2 The structural properties and the proposed optimization algorithm

The structural properties on the jobs sequencing and batching argument are proposed as follows.

Lemma 2 *For the problem* $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\, E_{max}$*, there is an optimal schedule in which there are no idle times between any consecutive batches or consecutive jobs in the same batch, and the first batch* b_1 *starts at time* t *such that*

$$t(1+\theta)^m+\sum_{k=1}^{m} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir}(1+\theta)^{m-k}=d \tag{3.9}$$

Then, the maximum earliness is

$$E_{max}=d-\frac{d-\sum_{k=2}^{m} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir}(1+\theta)^{m-k}}{(1+\theta)^{m-1}} \tag{3.10}$$

Proof Apart from satisfying the constraint that all jobs' completion times should be no later than the common due date, they need to be completed as late as possible to minimize the maximum earliness. Hence, the completion time of the last batch should be just equal to the common due date. Similar to the proof of Lemma 1, it can be derived that

$$t\left(1+\theta\right)^{m}+\sum_{k=1}^{m}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}=d.$$

If there is any idle time between consecutive batches or consecutive jobs in the same batch, then the starting time t will be earlier and the maximum earliness will be increased. Thus, there should be no idle times in the optimal schedule. Then, we have

$$E_{max}=d-t\left(1+\theta\right)-\sum_{r=1}^{n_1}\sum_{i=1}^{n}p_i i^a x_{ir}.$$

Based on Eq. (3.9), it should be

$$\begin{aligned}E_{max}&=d-\left(1+\theta\right)\\&\cdot\frac{d-\sum_{r=1}^{n_1}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-1}-\sum_{k=2}^{m}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}}{\left(1+\theta\right)^{m}}\\&-\sum_{r=1}^{n_1}\sum_{i=1}^{n}p_i i^a x_{ir}\\&=d-\frac{d-\sum_{k=2}^{m}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}}{\left(1+\theta\right)^{m-1}}.\end{aligned}$$

The proof is completed. □

Lemma 3 *For the problem* $1\left|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t\right|E_{max}$, *if* $m\geq 2$, *then the processing time of an arbitrary job in* b_1 *is no smaller than that in other batches.*

Proof Let π^* and π be an optimal schedule and a job schedule. The difference between the two schedules is the pairwise interchange of these two jobs $J_u\in b_1$ and $J_v\in b_p$, and J_u and J_v are assigned to the orders e and w in the optimal schedule π^*. $\pi^*=\left(b_1, W_1, b_p, W_2\right)$, $\pi=\left(\left(\left(b_1/\left\{J_u\right\}\right)\cup\left\{J_v\right\}\right), W_1, \left(\left(b_p/\left\{J_v\right\}\right)\cup\left\{J_u\right\}\right), W_2\right)$, where $n_p\geq 2$, $p=2,\ldots,m$, W_1 and W_2 represent two partial sequences, and W_1 or W_2 may be empty. We assume that $p_u<p_v$. For π^*, the maximum earliness is

$$E_{max}\left(\pi^*\right)=d-\frac{d-\sum_{k=2}^{m}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}}{\left(1+\theta\right)^{m-1}}.$$

For π, the maximum earliness is updated to

$$\begin{aligned}&E_{max}\left(\pi\right)\\&=d-\frac{1}{\left(1+\theta\right)^{m-1}}\left(d-\left(\sum_{k=2}^{p-1}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}\right.\right.\\&\quad+\sum_{r=1+\sum_{j=1}^{p-1}n_j}^{\sum_{j=1}^{p}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-p}-p_v w^a\left(1+\theta\right)^{m-p}\\&\quad\left.\left.+p_u w^a\left(1+\theta\right)^{m-p}+\sum_{k=p+1}^{m}\sum_{r=1+\sum_{j=1}^{k-1}n_j}^{\sum_{j=1}^{k}n_j}\sum_{i=1}^{n}p_i i^a x_{ir}\left(1+\theta\right)^{m-k}\right)\right)\end{aligned}$$

Then,

$$E_{max}\left(\pi^*\right) - E_{max}\left(\pi\right) = \frac{\left(p_v - p_u\right) w^a \left(1+\theta\right)^{m-p}}{\left(1+\theta\right)^{m-1}} > 0.$$

It can be derived that $E_{max}\left(\pi^*\right) > E_{max}\left(\pi\right)$, which conflicts with the optimal solution. Thus, it should be $p_u \geq p_v$. The proof is completed. □

Lemma 4 *For the problem* $1\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right| E_{max}$*, if* $n \geq c$*, then the first batch is full, i.e.,* $n_1 = c$*.*

Proof Let π^* and π be an optimal schedule and a job schedule. The difference between the two schedules is the transferring of a job J_u, and J_u is assigned to the position e in the optimal schedule π^*. That is, $\pi^* = \left(b_1, b_2, W\right)$, $\pi = \left(\left(b_1 \cup \left\{J_u\right\}\right), \left(b_2 / \left\{J_u\right\}\right), W\right)$, W represents a partial sequence, and it may be empty. We assume that $n \geq c$ and $n_1 < c$. J_u is the first one in b_2 and updated to the last one in b_1. After the transferring, we obtain that J_u is still in the position e.

For π^*, the maximum earliness is

$$E_{max}\left(\pi^*\right) = d - \frac{d - \sum_{k=2}^{m} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} \left(1+\theta\right)^{m-k}}{\left(1+\theta\right)^{m-1}}.$$

For π, the maximum earliness is updated to

$$E_{max}\left(\pi\right) = d - \frac{d - \left(\sum_{k=2}^{m} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} \left(1+\theta\right)^{m-k} - p_u e^a \left(1+\theta\right)^{m-2}\right)}{\left(1+\theta\right)^{m-1}}.$$

Then,

$$E_{max}\left(\pi^*\right) - E_{max}\left(\pi\right) = \frac{p_u e^a \left(1+\theta\right)^{m-2}}{\left(1+\theta\right)^{m-1}}.$$

It can be derived that $E_{max}\left(\pi^*\right) > E_{max}\left(\pi\right)$, which conflicts with the optimal solution. Thus, we can transfer the jobs from b_2 (or other batches) to b_1 until b_1 is full. The proof is completed. □

Based on the swapping and transferring operations of jobs, we obtain the following two lemmas.

Lemma 5 *For the problem* $1\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right| E_{max}$*, if* $n \geq c$*, then the jobs following* b_1 *belonged to the same batch should be sequenced in the non-decreasing order of* p_i *in an optimal schedule.*

Lemma 6 *For the problem* $1\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right| E_{max}$*, if* $n \geq c$*, then each batch following* b_1 *is full except possibly the second batch in an optimal schedule.*

Based on Lemma 6, we have the following corollary.

Corollary 1 *For the problem* $1\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right| E_{max}$*, if* $n \geq c$*, then* $n_1 = c, n_2 = n - \left(\lceil \frac{n}{c} \rceil - 1\right) c$*,* $n_k = c \; (k > 2)$ *in an optimal schedule.*

On the basis of the above lemmas and corollary, we design the following Algorithm 1 for solving the problem $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\, E_{max}$.

Algorithm 1

```
Step 1. All jobs are indexed in the non-decreasing order of $p_i$ such that $p_1 \le p_2 \le \cdots \le p_n$,
        and a job list is obtained.
Step 2. Place the last $c$ jobs in a batch.
Step 3. If there are unscheduled jobs in the job list, then place the first $n-\left(\left\lceil\frac{n}{c}\right\rceil-1\right)c$ jobs in a
        batch.
Step 4. If there are unscheduled jobs in the job list, then place the first $c$ jobs in a batch and
        iterate.
Step 5. The batches are scheduled in their generation order.
```

Theorem 1 *For the problem* $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\, E_{max}$*, an optimal schedule can be obtained by Algorithm 1 in* $O\,(n\log n)$ *time, and the optimal maximum earliness is*

$$E_{max}^{*} = \begin{cases} 0 & n>c \\ d-\dfrac{d-\left(\sum_{r=c+1}^{c+n-(\lceil \frac{n}{c}\rceil-1)c}\sum_{i=1}^{n} p_i i^a x_{ir}(1+\theta)^{\lceil \frac{n}{c}\rceil-2}+\sum_{k=3}^{\lceil \frac{n}{c}\rceil}\sum_{r=c+n-(\lceil \frac{n}{c}\rceil-1)c+(k-3)c+1}^{c+n-(\lceil \frac{n}{c}\rceil-1)c+(k-3)c+c}\sum_{i=1}^{n} p_i i^a x_{ir}(1+\theta)^{\lceil \frac{n}{c}\rceil-k}\right)}{(1+\theta)^{\lceil \frac{n}{c}\rceil-1}} & n\le c \end{cases} \tag{3.11}$$

Proof Based on Lemmas 2-6 and Corollary 1, an optimal solution can be generated by Algorithm 1, and also the result of the optimal solution can be obtained as Eq. (3.11). The time complexity of step 1 is $O\,(n\log n)$, and the time complexity of steps 2, 3, and 4 is at most $O\,(n)$. Thus, the total time complexity of Algorithm 1 is $O\,(n\log n)$. The proof is completed. □

3.2 Problem $1\,|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t|\sum_{i=1}^{n}U_i$

In this section, the problem with the objective of minimizing the number of tardy jobs is studied. We first give some properties for the optimal schedules, and then an optimization algorithm is developed to solve this problem. Here, the job sets of which the completion times are no later than and later than the common due date d are denoted as O (i.e., ordinary jobs) and L (i.e., late jobs), respectively. Similar to the problem in Sect. 3.1, the model, structural properties, and optimization algorithm are proposed as follows.

3.2.1 *Mixed integer programming model for the problem* $1\left|s - batch, p_{ir} = p_i r^a, s = \theta t\right|\sum_{i=1}^{n} U_i$

$$\text{Minimize } \sum\nolimits_{i=1}^{n} U_i \tag{3.12}$$
$$\text{Subject to } (3.3)-(3.5), (3.7)-(3.8)$$

The objective function (3.12) is to minimize the number of tardy jobs.

3.2.2 *The structural properties and the proposed optimization algorithm*

The structural properties on the jobs sequencing and batching argument are also proposed as follows.

Lemma 7 *For the problem* $1\,|s - batch, p_{ir} = p_i r^a, s = \theta t|\sum_{i=1}^{n} U_i$, *the following properties are satisfied in an optimal schedule:*

(1) All jobs should be sequenced in the non-decreasing order of p_i *in O, and all batches are full except possibly the first batch and the highest indexed batch in O;*
(2) The processing time of an arbitrary job in O is no more than that of other jobs in L.
(3) If there exists a batch b_k *in L satisfying that* $S(b_k) < d$ *and* $C(b_k) > d$*, then it should be* $C(b_{k-1})(1+\theta)(1+p_i w^a) > d$ *for an arbitrary job* $J_i \in L$*, where* $w = \sum_{i=1}^{k-1} n_i + 1$ *and it is the first position after* b_{k-1}*.*

Proof (1) This property can be proved by swapping and transferring operations of jobs. We omit the proof.

(2) Let π^* and π be an optimal schedule and a job schedule. The difference between the two schedules is the pairwise interchange of these two jobs J_u and J_v (suppose J_u and J_v are in the e-th and w-th orders, and $e < w$), that is, $\pi^* = \left(W_1, b_p, W_2, b_q, W_3\right)$, $\pi = \left(W_1, \left(b_p/\{J_u\}\right) \cup \{J_v\}, W_2, \left(b_q/\{J_v\}\right) \cup \{J_u\}, W_3\right)$. W_1, W_2, and W_3 represent three partial sequences, where W_1, W_2, or W_3 may be empty, $b_p \subseteq O$, and $b_q \subseteq L$. Here we assume that $p_u > p_v$.

The completion time of b_p in π^* is

$$C\left(b_p\left(\pi^*\right)\right) = \sum_{k=1}^{p} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{p-k}.$$

The completion time of b_p in π is

$$C\left(b_p(\pi)\right) = \sum_{k=1}^{p} \sum_{r=1+\sum_{j=1}^{k-1} n_j}^{\sum_{j=1}^{k} n_j} \sum_{i=1}^{n} p_i i^a x_{ir} (1+\theta)^{p-k} - p_u e^a + p_v e^a.$$

Then, $C\left(b_p\left(\pi^*\right)\right) - C\left(b_p(\pi)\right) = p_u e^a - p_v e^a$. Since $p_u > p_v$, it can be derived that $C\left(b_p\left(\pi^*\right)\right) > C\left(b_p(\pi)\right)$, which conflicts with the optimal solution. Hence, it should be $p_u \le p_v$ in an optimal schedule.

(3) If there exist a batch b_k and a job J_i in L satisfying that $S(b_k) < d$, $C(b_k) > d$, and $C(b_{k-1})(1+\theta)(1+p_i w^a) \le d$, then the solution can be improved after J_i is placed into

a single batch, which contradicts with the optimal schedule. Thus, it can be derived that $C(b_{k-1})(1+\theta)(1+p_i w^a) > d$ for any job $J_i \in L$.

This completes the proof. □

Based on Lemma 7, we develop the following Algorithm 2 to solve the problem $1|s-batch, p_{ir}=p_i r^a, s=\theta t|\sum_{i=1}^{n} U_i$.

Algorithm 2

```
Step 1. Index all jobs in the non-decreasing order of $p_i$ such that $p_1 \le p_2 \le \cdots \le p_n$, and
        obtain a job list. Initialize the first batch, and set $k=1$, $i=1$, $b_k=\{J_i\}$, $n_k=1$,
        $C(b_k)=p_i i^a$.
Step 2. If $i=n$, then go to step 6. Otherwise, update $i=i+1$ and go to step 3.
Step 3. If $n_k=c$, then go to step 4. Otherwise, go to step 5.
Step 4. If $C(b_k)(1+\theta)+p_i i^a \le d$, then update $k=k+1$, $b_k=\{J_i\}$, $n_k=1$, and
        $C(b_k)=C(b_k)(1+\theta)+p_i i^a$, go to step 2. Otherwise, go to step 7.
Step 5. If $C(b_k)+p_i i^a \le d$, then update $b_k=b_k \cup \{J_i\}$, $n_k=n_k+1$, and $C(b_k)=$
        $C(b_k)+p_i i^a$, go to step 2. Otherwise, go to step 6.
Step 6. Transfer the last job from $b_f$ into the first position of $b_{f+1}$ for $f=1,2,\cdots,k-1$,
        update each batch and re-calculate the completion time of each batch(job), go to step 3.
Step 7. Batch the remaining jobs arbitrarily and output the schedule of batches as their
        generation sequence.
```

Theorem 2 *For the problem* $1|s-batch, p_{ir}=p_i r^a, s=\theta t|\sum_{i=1}^{n} U_i$*, an optimal schedule can be obtained by Algorithm 2 in* $O(n\log n)$ *time.*

Proof Based on Lemma 7, an optimal solution can be obtained by Algorithm 2. The time complexity of step 1 is $O(n\log n)$, the total time complexity of steps 2, 3, 4, 5, and 6 is $O(n)$. Thus, the time complexity of Algorithm 2 is at most $O(n\log n)$. □

4 Parallel-machine scheduling problems

In this section, the hybrid VNS–GSA algorithm is used to solve the parallel-machine scheduling problems with the objective of minimizing the maximum earliness and total number of tardy jobs, respectively.

4.1 Mixed integer programming models for the parallel-machine scheduling problems

4.1.1 Mixed integer programming model for the problem $P\,|s-batch,\,p_{ir}=p_i r^a,\,s=\theta t|\,E_{max}$

Minimize (3.2)
Subjec to:

$$\sum_{i=1}^{n} y_{gik} \le c, \quad k=1,2,\ldots,m_g,\ g=1,2,\ldots,q \tag{4.1}$$

$$\sum_{k=1}^{m_g} y_{gik} = 1, \quad i=1,2,\ldots,n,\ g=1,2,\ldots,q \tag{4.2}$$

$$S\left(b_{g(k+1)}\right) \ge C\left(b_{gk}\right)(1+\theta), \quad k=1,2,\ldots,m-1,\ g=1,2,\ldots,q \tag{4.3}$$

$$d \ge t_g + \sum_{k=1}^{m} s_{gk} + \sum_{i=1}^{n} p_i, \quad g=1,2,\ldots,q \tag{4.4}$$

$$\begin{aligned} &C(b_{gk}) - C(b_{gl}) + Mz_{gkl} - C(b_{gk})(1+\theta)\\ &\quad -P(b_{gk}) \ge 0, \quad k,l=1,2\ldots,m,\ g=1,2,\ldots,q \end{aligned} \tag{4.5}$$

$$y_{gik}, z_{kl} = 0 \text{ or } 1, \quad k,l=1,2,\ldots,m_g,\ g=1,2,\ldots,q,\ i=1,2,\ldots,n \tag{4.6}$$

Constraint set (4.1) assures that the number of jobs in any batch should be no more than the machine capacity on each machine. Constraint set (4.2) guarantees that each job should be only assigned to one batch on each machine. Constraint set (4.3) makes sure that setup time is required before processing each batch on each machine. Constraint set (4.4) ensures that all jobs should be completed no later than the common due date d on each machine, and here t_g denotes the starting time to process the batches. Constraint set (4.5) indicates that there is no overlapping situation between two continuous batches on each batch. Constraint set (4.6) defines the ranges of the variables.

4.1.2 Mixed integer programming model for the problem $P\,|s-batch,\,p_{ir}=p_i r^a,\,s=\theta t|\sum_{i=1}^{n} U_i$

Minimize (3.12)
Subject to $(4.1)-(4.3)$, $(4.5)-(4.6)$

4.2 Key procedures of VNS–GSA

Variable neighborhood search (VNS) was proposed by Mladenović and Hansen (1997), since when this algorithm has been extensively studied by many researchers and applied in many practical combinatorial optimization problems. Gravitational search algorithm (GSA), proposed by Rashedi et al. (2009), is a population-based heuristic algorithm based on Newton's theory of gravity and mass interactions. In this paper, we combine the main features and steps of VNS and GSA to develop a novel hybrid VNS–GSA algorithm for solving the parallel-machine scheduling problems. The key procedures of VNS–GSA are described as follows:

4.2.1 Coding scheme

In the proposed VNS–GSA, for ensuring the continuity of search space in GSA algorithm, the random key encoding way is applied in our problem, which was first proposed by Bean (1994) to keep feasibility from parent to offspring in genetic algorithm. In this paper, a solution to the problem of assigning jobs to the parallel-machine is an array with the length equal to $n + q - 1$, and each position value is a decimal between 0 and 1.

4.2.2 Encoding scheme

We should map this decimal array into the jobs number array on each machine before encoding. Firstly, we map this decimal array into an integer array, and each decimal sequenced in the non-decreasing order of the value is transformed to the order number. For example, the smallest decimal is transformed to the integer 1, and the second smallest decimal is transferred to the integer 2, and so on. Then, if the value of any position is greater than n, then this position is marked as a flag to assign the following jobs into a different machine, and this flag mechanism has been applied in Wang and Chou (2010).

Assume that there are five jobs to be processed on two serial-batching machines. In Fig. 1 there is an example of a decimal array (0.67, 0.78, 0.54, 0.98, 0.45, 0.08) and the mapping process. The jobs $\{J_4, J_5, J_3\}$ are assigned on M_1 and $\{J_2, J_1\}$ on machine M_2.

Then, we can obtain the jobs set assigned on each machine, and different encoding strategies are designed for different objective functions. The detail of encoding strategy for problem $P\,|s-batch, p_{ir} = p_i r^a, s = \theta t|\, E_{max}$ is described as follows:

Encoding strategy for problem $P\,|s\text{-}batch, p_{ir} = p_i r^a, s = \theta t|\, E_{max}$

```
Step 1.   Set $g = 1, E_{max} = 0$
Step 2.   Apply Algorithm 1 to sequence and batch all jobs assigned on $M_g$, and $E^g_{max}$ is obtained for $M_g$
Step 3:   If $E_{max} < E^g_{max}$, then $E_{max} = E^g_{max}$
Step 4:   If $g < q$, set $g = g + 1$, then go to step 2. Otherwise, output the result of $E_{max}$
```

The detail of encoding strategy for problem $P\,|s-batch, p_{ir} = p_i r^a, s = \theta t|\sum_{i=1}^{n} U_i$ is described as follows:

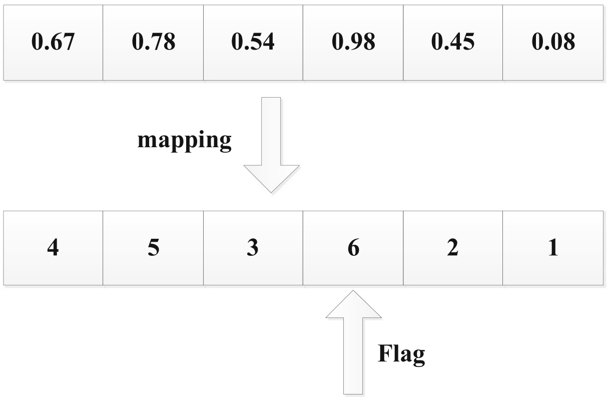


Fig. 1 An example of encoding mapping

Encoding strategy for problem $P\left|s\text{-}batch, p_{ir} = p_i r^a, s = \theta t\right| \sum_{i=1}^n U_i$

```
Step 1.  Set $g = 1, \sum_{i=1}^n U_i = 0$
Step 2.  Apply Algorithm 2 to sequence and batch all jobs assigned on $M_g$, and $\sum_{i=1}^n U_i^g$ is obtained for
           $M_g$ Set $\sum_{i=1}^n U_i = \sum_{i=1}^n U_i + \sum_{i=1}^n U_i^g$
Step 3.  If $g < q$, set $g = g + 1$, then go to step 2. Otherwise, output the result of $\sum_{i=1}^n U_i = 0$
```

4.2.3 Neighborhood structure

Swap operator is widely applied in neighborhood structure of combinatorial optimization problem. A multiple swap-based neighborhood structure is applied in this paper, which can be described as follows: randomly select two different positions from the solution and swap them to form a new solution, and the above process is defined as 1-swap, and then repeat to randomly swap two different positions from the new solution, it is defined as 2-swap, and so on.

4.2.4 GSA-based local search in variable neighborhood search

In order to improve the efficiency of local search in VNS, we use GSA as local search operator to strengthen the optimization ability of the algorithm. The detail of GSA-Based Local Search operation is described as follows (Rashedi et al. 2009):

GSA-Based Local Search operation

```
Step 1.  Randomly select $m$ solutions from a certain neighborhood of the current optimal solution in
           variable neighborhood search
Step 2.  Calculate the fitness of each solution
Step 3.  Update best and worst of the population
Step 4.  Calculate mass and acceleration for each solution
Step 5.  Update velocity and position for each solution
Step 6.  If termination condition is met, then stop the iteration
```

4.3 The main steps and framework of VNS–GSA

In our problem, the initial solution is denoted by x, the neighborhood of the solution is denoted by $N_o(x)$, and the current iteration step is denoted by it. We initialize the following parameters o, o_{min}, o_{max}, t_{max}, o_{step}, where o is the current neighborhood, o_{min} is the first neighborhood, o_{max} is the last neighborhood, t_{max} is the maximum number of iterations, o_{step} is the step length of iteration.

Main steps of VNS–GSA

```
Step 1.  Set $it = 1$, execute local search procedure to find a local optimal solution for initial solution and
           set it as the optimal solution $x_{best}$
Step 2.  Set $o = o_{min}$
Step 3.  Execute GSA-Based Local Search for $N_o(x_{best})$ to obtain a solution $x'$
Step 4.  If solution $x'$ is better than $x_{best}$, then set $x_{best} = x'$
           Otherwise, set $o = o + 1$, if $o \le o_{max}$ and go to step 3
Step 5.  If $it \le t_{max}$, then stop the iteration. Otherwise, set $it = it + 1$ and go to step 2
```

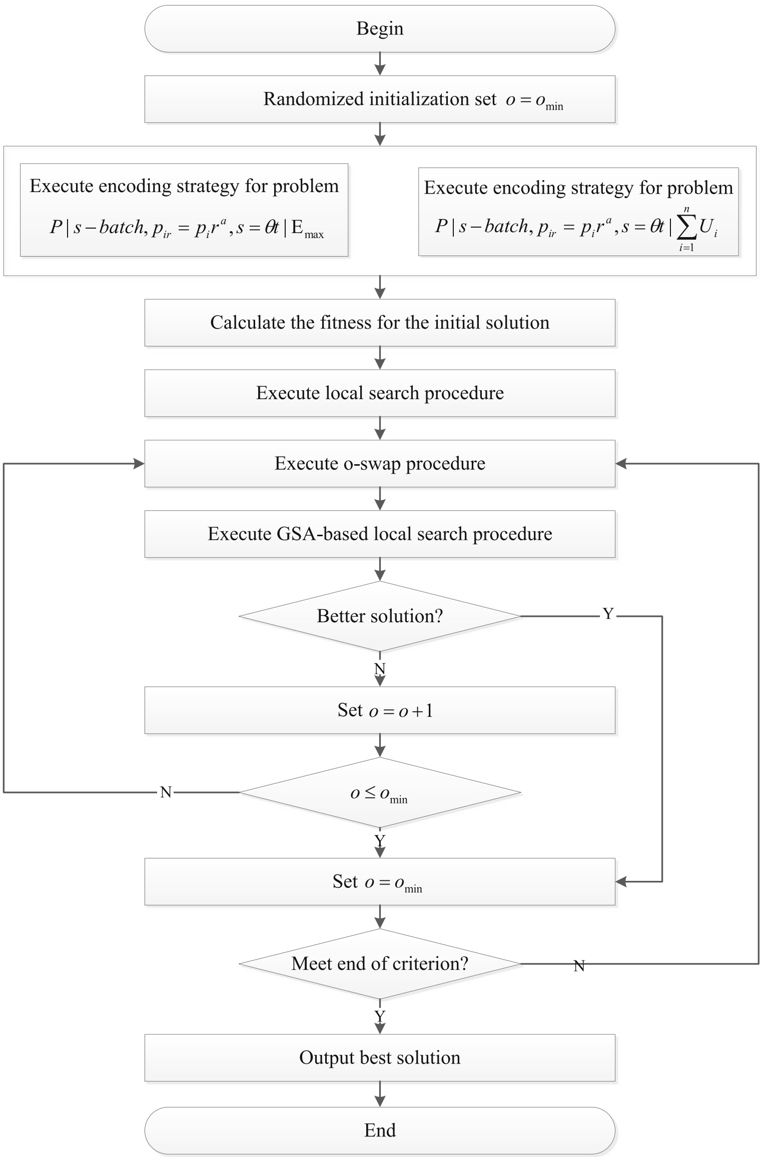

Begin
Randomized initialization set $o = o_{\min}$
Execute encoding strategy for problem $P \mid s-batch, p_{ir} = p_i r^a, s = \theta t \mid E_{\max}$
Execute encoding strategy for problem $P \mid s-batch, p_{ir} = p_i r^a, s = \theta t \mid \sum_{i=1}^{n} U_i$
Calculate the fitness for the initial solution
Execute local search procedure
Execute o-swap procedure
Execute GSA-based local search procedure
Better solution?
Y
N
Set $o = o + 1$
$o \le o_{\min}$
N
Y
Set $o = o_{\min}$
Meet end of criterion?
N
Y
Output best solution
End

Fig. 2 The framework of VNS–GSA

The framework of VNS–GSA is shown in Fig. 2 to solve the problem $P\,|s - batch,\, p_{ir} = p_i r^a,\, s = \theta t|\, E_{max}$ and $P\,|s - batch,\, p_{ir} = p_i r^a,\, s = \theta t|\, \sum_{i=1}^{n} U_i$.

4.4 Computational experiments and comparison

In this sub-section, a serial of computational experiments are conducted to test the performance of our proposed algorithm VNS–GSA, compared with VNS (Lei and Guo 2016), GSA (Rashedi et al. 2009), and SA (Damodaran and Vélez-Gallego 2012). We also conducted computational experiments for the problems of $P\,|s - batch,\, p_{ir} = p_i r^a,\, s = \theta t|\, E_{max}$ and

Table 3 Parameters setting

Parameters	Description	Value
n	The number of jobs in the small-size problems	60,70, 80, 90, 100
	The number of jobs in the large-size problems	200, 300, 400, 500, 600
q	The number of machines	U[5,10]
p_i	The basic processing time of $J_i, i = 1, 2, \ldots, n$	U[1,10]
α	The learning index of processing time	U[−1,0]
d	The common due date of all jobs in the problem $P\left\|s - batch, p_{ir} = p_i r^a, s = \theta t\right\| E_{max}$ with small size	U[800,900]
	The common due date of all jobs in the problem $P\left\|s - batch, p_{ir} = p_i r^a, s = \theta t\right\| E_{max}$ with large size	U[80000,90000]
	The common due date of all jobs in the problem $P\left\|s - batch, p_{ir} = p_i r^a, s = \theta t\right\| \sum_{i=1}^{n} U_i$ with small size	U[20,30]
	The common due date of all jobs in the problem $P\left\|s - batch, p_{ir} = p_i r^a, s = \theta t\right\| \sum_{i=1}^{n} U_i$ with large size	U[100,300]
c	The capacity of the serial-batching machine	U[5,10]
θ	The deteriorating rate of the setup time	U[0,1]

$P\,|s - batch, p_{ir} = p_i r^a, s = \theta t|\,\sum_{i=1}^{n} U_i$. The parameters of the test problems were randomly generated according to the practical production situations of an aluminum factory, as shown in Table 3.

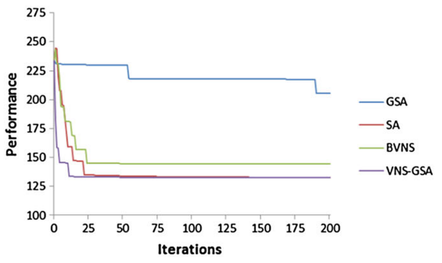
4.4.1 Experiments and comparison for the problem $P\,|s - batch, p_{ir} = p_i r^a, s = \theta t|\,E_{max}$

In order to test the performance of proposed VNS–GSA, the proposed algorithm and another three algorithms have been applied to solve the problem $P\,|s - batch, p_{ir} = p_i r^a, s = \theta t|\,E_{max}$ for comparison. In Table 4, the average objective value (Avg.Obj) and the maximum objective value (Max.Obj) for the problem $P\,|s - batch, p_{ir} = p_i r^a, s = \theta t|\,E_{max}$ are listed. It is easy to find that the VNS–GSA can obtain better solutions than compared algorithms, especially for the large-size problems.

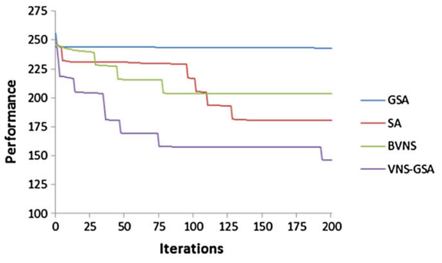
We also compared the proposed algorithm with other compared algorithms on the convergence speed and efficiency. Figures 3 and 4 show the convergence performance of VNS–GSA, GSA, VNS and SA for small-size and large-size problem instances, respectively. Compared to GSA, VNS, and SA, the proposed VNS–GSA converges faster and has better optimization capability.

Table 4 Computational results for the problem $P \left| s - batch, p_{ir} = p_i r^a, s = \theta t \right| E_{max}$

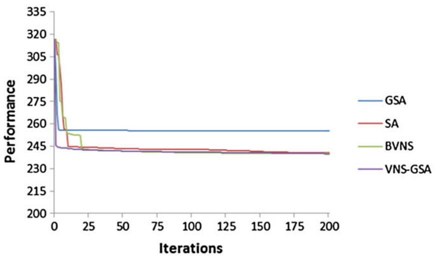
No.	n	GSA		SA		BVNS		VNS–GSA	
		Ave. obj	Max. obj	Ave. obj	Max. obj	Ave. obj	Max. obj	Ave. obj	Max. obj
1	60	205.383	271.797	132.639	273.832	144.183	272.041	132.637	271.086
2	70	243.046	273.158	180.506	272.426	203.374	275.649	146.366	270.239
3	80	255.177	372.453	240.504	375.098	240.030	373.118	239.916	272.890
4	90	265.983	374.700	242.790	374.739	261.287	375.402	242.371	370.601
5	100	315.890	375.151	255.193	455.804	282.060	454.403	246.022	371.486
6	200	34,988.966	46,591.530	27,867.785	46,588.821	28,907.504	46,586.855	27,838.218	46,582.954
7	300	47,714.626	53,822.620	41,779.008	59,842.115	41,918.831	59,842.046	40,187.705	59,809.492
8	400	58,123.428	77,848.042	51,440.756	69,044.059	51,439.285	69,042.407	48,433.178	64,862.430
9	500	64,520.017	75,431.733	58,071.212	75,431.058	58,282.718	75,430.865	58,067.995	72,529.272
10	600	69,250.738	79,865.588	65,267.097	79,865.436	63,873.380	79,865.335	63,177.735	79,864.733



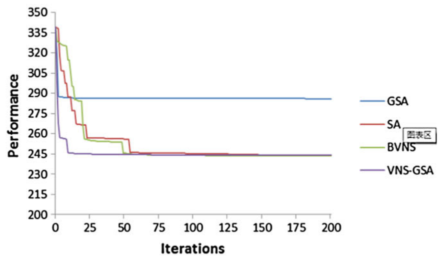
(a)



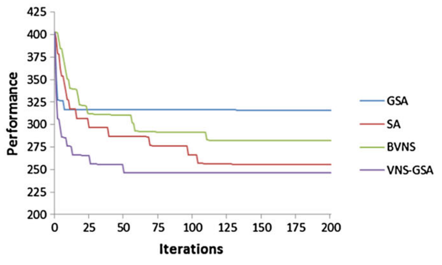
(b)



(c)



(d)



(e)

Fig. 3 Convergence curves for the problem $P\left|s-batch,\ p_{ir}=p_i r^a,\ s=\theta t\right|E_{max}$ with small size. **a** Convergence curves for problem 1, **b** convergence curves for problem 2, **c** convergence curves for problem 3, **d** convergence curves for problem 4, **e** convergence curves for problem 5

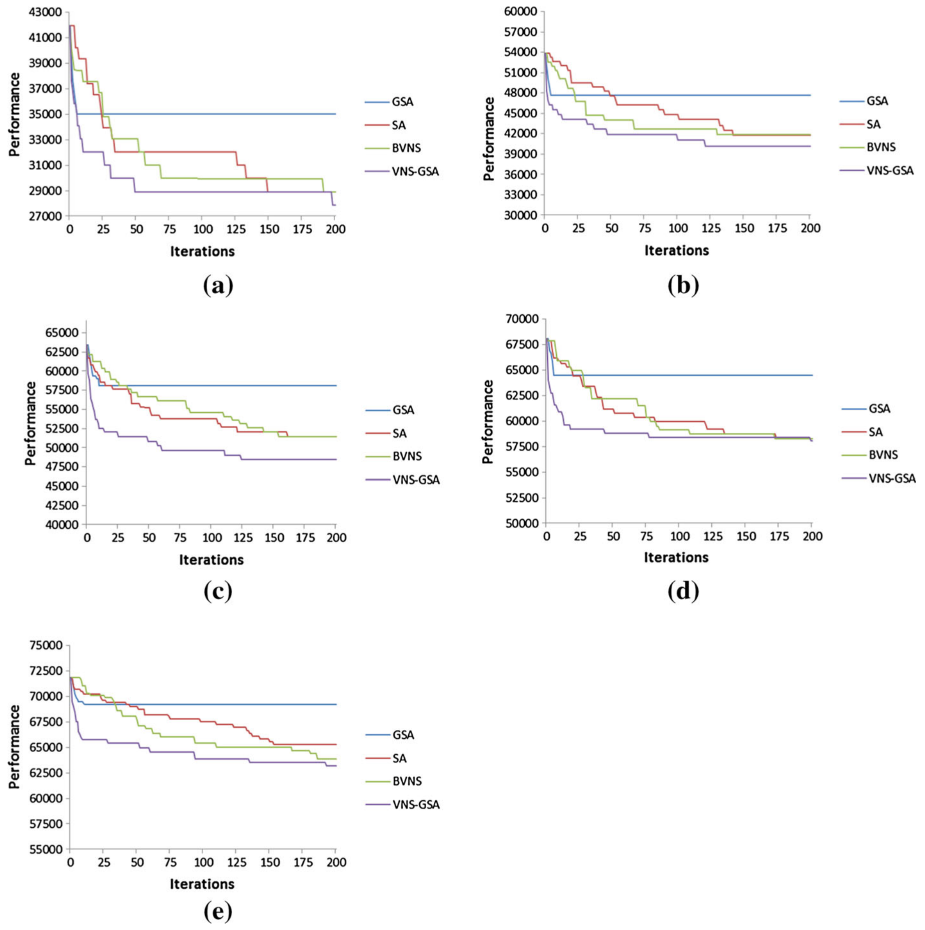


Fig. 4 Convergence curves for the problem $P\left|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t\right| E_{max}$ with large size. **a** Convergence curves for problem 6, **b** convergence curves for problem 7, **c** convergence curves for problem 8, **d** convergence curves for problem 9, **e** convergence curves for problem 10

4.4.2 Experiments and comparison for the problem $P\left|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t\right|\sum_{i=1}^{n}U_i$

In Table 5, we list the Ave.obj and Max.obj of the problem $P\left|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t\right|\sum_{i=1}^{n}U_i$ generated by the proposed algorithm and other algorithms. It is also obvious that VNS–GSA has better optimization capability than other algorithms.

Figures 5 and 6 show the convergence performance of the proposed algorithm and compared algorithms when solving the problem $P\left|s-batch,\, p_{ir}=p_i r^a,\, s=\theta t\right|\sum_{i=1}^{n}U_i$ with small and large problem instances, respectively. It is obvious that the proposed algorithm has better convergence than compared algorithms.

In all, from the above results we can conclude that the proposed VNS–GSA has faster convergence speed when solving problems compared with VNS (Lei and Guo 2016), GSA (Rashedi et al. 2009), and SA (Damodaran and Vélez-Gallego 2012). Then, it can be inferred that the proposed VNS–GSA is very stable and robust in terms of solution quality and convergence speed.

Table 5 Computational results for the problem $P\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right|\sum\limits_{i=1}^{n} U_i$

No.	n	GSA		SA		BVNS		VNS–GSA	
		Ave. obj	Max. obj	Ave. obj	Max. obj	Ave. obj	Max. obj	Ave. obj	Max. obj
11	60	21.4	27	20.2	29	19.8	27	19.7	25
12	70	26.7	40	25.3	38	25.2	39	25.2	38
13	80	34.7	44	33.3	48	33.4	44	32.8	44
14	90	41.7	55	40.2	54	39.4	51	39	51
15	100	48.9	61	46.2	57	46.3	59	46.1	57
16	200	14.3	46	12.3	28	6.5	31	5.4	27
17	300	63.1	106	57	99	54.4	98	49.8	88
18	400	46.8	90	28.7	72	15.2	42	12.1	41
19	500	47.4	104	29.7	60	10.3	46	6.8	35
20	600	91.2	189	66.1	126	42.5	99	33.5	95

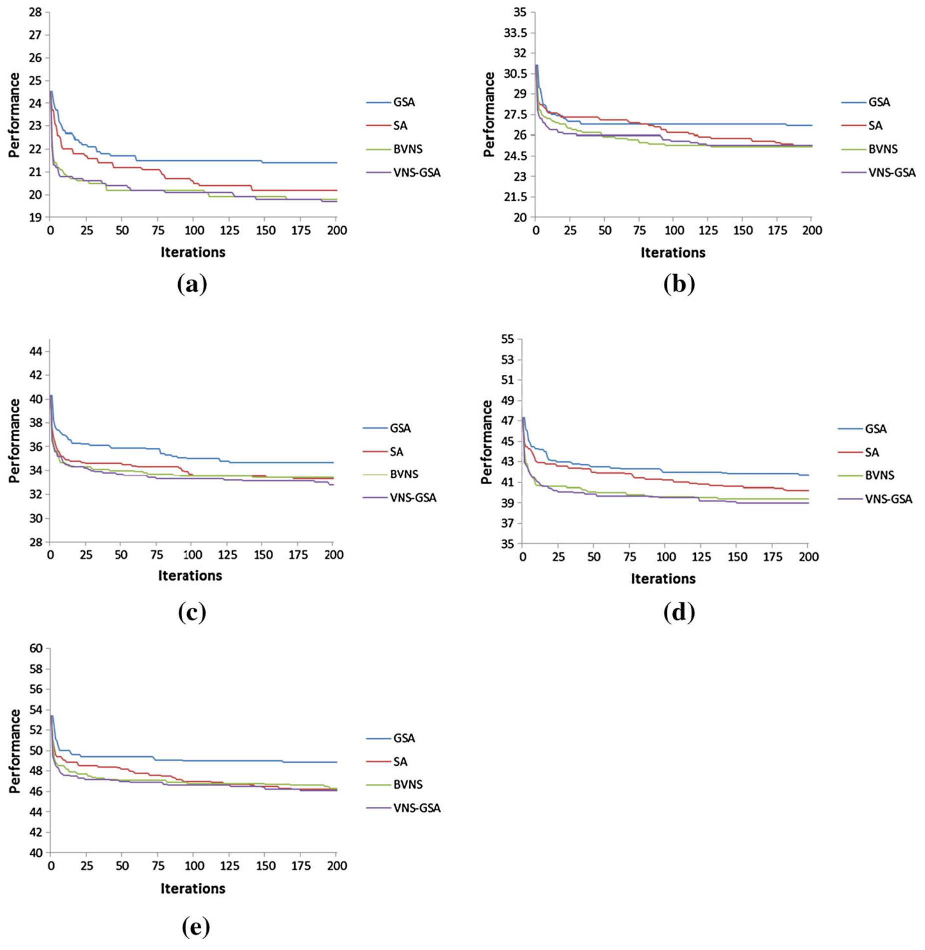


Fig. 5 Convergence curves for the problem $P\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right|\sum_{i=1}^{n} U_i$ with small size. **a** Convergence curves for problem 11, **b** convergence curves for problems 12, **c** convergence curves for problem 13, **d** convergence curves for problem 14, **e** convergence curves for problem 15

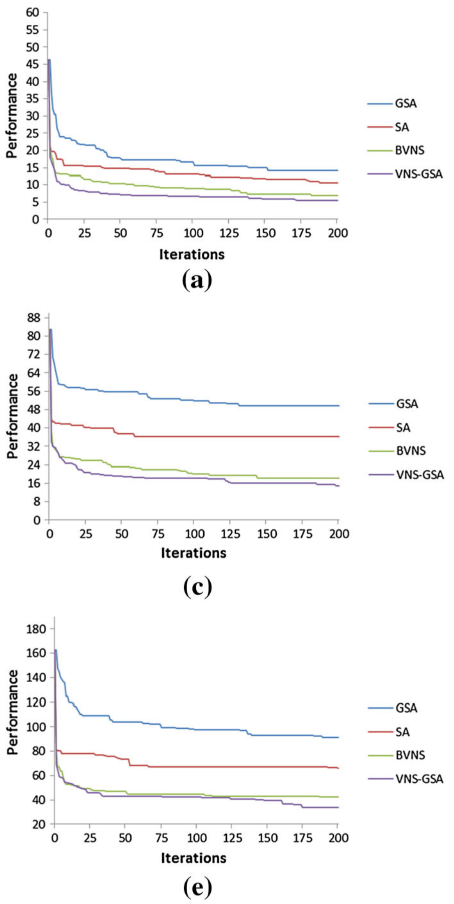


Fig. 6 Convergence curves for the problem $P\left|s-batch, p_{ir}=p_i r^a, s=\theta t\right|\sum_{i=1}^{n}U_i$ with large size. **a** Convergence curves for problem 16, **b** convergence curves for problem 17, **c** convergence curves for problem 18, **d** convergence curves for problem 19, **e** convergence curves for problem 20

5 Conclusions

In this paper, we study single-machine and parallel-machine serial-batching scheduling problems with learning effect, considering the time-dependent set-up time. Under the setting of single-machine, we propose the models and structural properties for the problems with the objectives of minimizing the maximum earliness and the number of tardy jobs, and develop optimization algorithms to solve these two problems, respectively. Under the setting of the parallel-machine, we propose a hybrid VNS–GSA algorithm to solve the scheduling problems with the same two objectives. In future research, we will investigate more general serial-batching scheduling models with the learning effect, take different objective functions into consideration, and extend our models to the background of supply chain scheduling.

Acknowledgements This work is supported by the National Natural Science Foundation of China (Nos. 71601065, 71231004, 71501058, 71671055, 71690235), and Innovative Research Groups of the National Natural Science Foundation of China (71521001), the Humanities and Social Sciences Foundation of the

Chinese Ministry of Education (No. 15YJC630097), Anhui Province Natural Science Foundation (No. 1608085QG167), and the Fundamental Research Funds for the Central Universities (Nos. JZ2016HGTA0709, JZ2016HGTB0727). Panos M. Pardalos is partially supported by the project of "Distinguished International Professor by the Chinese Ministry of Education" (MS2014HFGY026).

References

Bai, J., Li, Z. R., & Huang, X. (2012). Single-machine group scheduling with general deterioration and learning effects. *Applied Mathematical Modelling*, *36*(3), 1267–1274.

Bean, J. C. (1994). Genetic algorithms and random keys for sequencing and optimization. *Orsa Journal on Computing*, *6*, 154–160.

Biskup, D. (1999). Single-machine scheduling with learning considerations. *European Journal of Operational Research*, *115*(1), 173–178.

Cheng, T. C. E., Hsu, C. J., Huang, Y. C., & Lee, W. C. (2011). Single-machine scheduling with deteriorating jobs and setup times to minimize the maximum tardiness. *Computers & Operations Research*, *38*(12), 1760–1765.

Cheng, T. C. E., & Wang, G. (2000). Single machine scheduling with learning effect considerations. *Annals of Operations Research*, *98*, 273–290.

Damodaran, P., & Vélez-Gallego, M. C. (2012). A simulated annealing algorithm to minimize makespan of parallel batch processing machines with unequal job ready times. *E xpert Systems with Applications*, *39*(1), 1451–1458.

Eren, T. (2009). A bicriteria parallel machine scheduling with a learning effect of setup and removal times. *Applied Mathematical Modelling*, *33*, 1141–1150.

Eren, T., & Güner, E. (2009). A bicriteria parallel machine scheduling with a learning effect. *International Journal of Advanced Manufacturing Technology*, *40*(11), 1202–1205.

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.

Hsu, C.-J., Kuo, W.-H., & Yang, D.-L. (2011). Unrelated parallel machine scheduling with past-sequence-dependent setup time and learning effects. *Applied Mathematical Modelling*, *35*, 1492–1496.

Huang, X., Wang, M. Z., & Wang, J. B. (2011). Single-machine group scheduling with both learning effects and deteriorating jobs. *Computers & Industrial Engineering*, *60*, 750–754.

Janiak, A., & Rudek, R. (2009). Experience-based approach to scheduling problems with the learning effect. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *39*(2), 344–357.

Kumar, A., & Tan, Y. (2015). Demand effects of joint product advertising in online videos. *Management Science*, *61*, 1921–1937.

Lee, W.-C., Wu, C.-C., & Hsu, P.-H. (2010). A single-machine learning effect scheduling problem with release times. *Omega*, *38*, 3–11.

Lei, D., & Guo, X. (2016). Variable neighborhood search for the second type of two-sided assembly line balancing problem. *Computers & Operations Research*, *72*(C), 183–188.

Lu, Y.-Y., Wei, C.-M., & Wang, J.-B. (2012). Several single-machine scheduling problems with general learning effects. *Applied Mathematical Modelling*, *36*(11), 5650–5656.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100.

Mosheiov, G. (2001). Parallel machine scheduling with a learning effect. *Journal of the Operational Research Society*, *52*, 1165–1169.

Mosheiov, G., & Sidney, J. B. (2003). Scheduling with general job-dependent learning curves. *European Journal of Operational Research*, *147*(3), 665–670.

Okołowski, D., & Gawiejnowicz, S. (2010). Exact and heuristic algorithms for parallel-machine scheduling with DeJong's learning effect. *Computers & Industrial Engineering*, *59*(2), 272–279.

Pan, E., Wang, G., Xi, L., Chen, L., & Han, X. (2014). Single-machine group scheduling problem considering learning, forgetting effects and preventive maintenance. *International Journal of Production Research*, *52*(19), 5690–5704.

Paul, A., Tan, Y., & Vakharia, A. (2015). Inventory planning for a modular product family. *Production and Operations Management*, *24*(7), 1033–1053.

Pei, J., Liu, X., Pardalos, P. M., Fan, W., & Yang, S. (2015b). Single machine serial-batching scheduling with independent setup time and deteriorating job processing times. *Optimization Letters*, *9*(1), 91–104.

Pei, J., Liu, X., Pardalos, P. M., Fan, W., & Yang, S. (2017b). Scheduling deteriorating jobs on a single serial-batching machine with multiple job types and sequence-dependent setup times. *Annals of Operations Research*, *249*, 175–195.

Pei, J., Liu, X., Pardalos, P. M., Li, K., Fan, W., & Migdalas, A. (2016). Single-machine serial-batching scheduling with a machine availability constraint, position-dependent processing time, and time-dependent set-up time. *Optimization Letters*. doi:10.1007/s11590-016-1074-9.

Pei, J., Liu, X., Pardalos, P. M., Migdalas, A., & Yang, S. (2017a). Serial-batching scheduling with time-dependent setup time and effects of deterioration and learning on a single-machine. *Journal of Global Optimization*, *67*(1), 251–262.

Pei, J., Pardalos, P. M., Liu, X., Fan, W., & Yang, S. (2015a). Serial batching scheduling of deteriorating jobs in a two-stage supply chain to minimize the makespan. *European Journal of Operational Research*, *244*(1), 13–25.

Pinedo, M. (2002). *Scheduling: Theory, algorithms, and systems* (2nd ed.). Upper Saddle River, NJ: Prentice-Hall.

Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Science*, *179*(13), 223–2248.

Tan, Y., & Carrillo, J. (2017). Strategic analysis of the agency model for digital goods. *Production and Operations Management*. doi:10.1111/poms.12595.

Tan, Y., Carrillo, J., & Cheng, H. K. (2016). The agency model for digital goods. *Decision Science*, *4*, 628–660.

Teyarachakul, S., Chand, S., & Ward, J. (2011). Effect of learning and forgetting on batch sizes. *Production and Operations Management*, *20*(1), 116–128.

Wang, D., Huo, Y. Z., & Ji, P. (2014a). Single-machine group scheduling with deteriorating jobs and allotted resource. *Optimization Letters*, *8*, 591–605.

Wang, H.-M., & Chou, F.-D. (2010). Solving the parallel batch-processing machines with different release times, job sizes, and capacity limits by metaheuristics. *Expert Systems with Applications*, *37*, 1510–1521.

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, J. B., Ng, C. T., Cheng, T. C. E., & Liu, L. L. (2008). Single-machine scheduling with a time-dependent learning effect. *International Journal of Production Economics*, *111*, 802–811.

Wang, J.-B., & Wang, J.-J. (2014). Single machine scheduling with sum-of-logarithm-processing-times based and position based learning effects. *Optimization Letters*, *8*(3), 971–982.

Wang, X.-R., Wang, J.-B., & Jin, J. (2014b). Single machine scheduling with truncated job-dependent learning effect. *Optimization Letters*, *8*(2), 669–677.

Xuan, H., & Tang, L. (2007). Scheduling a hybrid flowshop with batch production at the last stage. *Computers & Operations Research*, *34*(9), 2718–2733.

Yang, D. L., & Kuo, W. H. (2009). A single-machine scheduling problem with learning effects in intermittent batch production. *Computers & Industrial Engineering*, *57*(3), 762–765.

Yang, W. H., & Chard, S. (2008). Learning and forgetting effects on a group scheduling problem. *European Journal of Operational Research*, *187*(3), 1033–1044.

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.

Zhu, Z., Chu, F., Sun, L., & Liu, M. (2013). Single machine scheduling with resource allocation and learning effect considering the rate-modifying activity. *Applied Mathematical Modelling*, *37*(7), 5371–5380.