J Glob Optim (2018) 71:147–163
https://doi.org/10.1007/s10898-017-0536-7

Serial-batching group scheduling with release times and the combined effects of deterioration and truncated job-dependent learning

Wenjuan Fan[1,2] · Jun Pei[1,2] · Xinbao Liu[1,3] · Panos M. Pardalos[2] · Min Kong[1,3]

Received: 14 January 2017 / Accepted: 15 May 2017 / Published online: 1 June 2017

Abstract This paper investigates a single machine serial-batching scheduling problem considering release times, setup time, and group scheduling, with the combined effects of deterioration and truncated job-dependent learning. The objective of the studied problem is to minimize the makespan. Firstly, we analyze the special case where all groups have the same arrival time, and propose the optimal structural properties on jobs sequencing, jobs batching, batches sequencing, and groups sequencing. Next, the corresponding batching rule and algorithm are developed. Based on these properties and the scheduling algorithm, we develop a hybrid VNS–ASHLO algorithm incorporating variable neighborhood search (VNS) and adaptive simplified human learning optimization (ASHLO) algorithms to solve the general case of the studied problem. Computational experiments on randomly generated instances are conducted to compare the proposed VNS–ASHLO with the algorithms of VNS, ASHLO, Simulated Annealing (SA), and Particle Swarm Optimization (PSO). The results based on instances of different scales show the effectiveness and efficiency of the proposed algorithm.

Keywords Scheduling · Serial-batching · Group scheduling · Release time · Deterioration · Truncated job-dependent learning

✉ 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

1 Introduction

In many production scenarios, the effects of learning and deterioration are common and thus have received increasing attention from academy in recent years. Readers can find these works in Wang et al. [1,2,5], Cheng et al. [3], Kuo [4], Yang [6] and Yang et al. [7], etc. Specifically, there is also a growing interest in studying the truncated job-dependent learning effect. Wu et al. [8] investigated a two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function, and they proposed a branch-and-bound and a genetic heuristic-based algorithm to solve this problem. He et al. [9] studied the resource constrained scheduling problem with general truncated job-dependent learning effect, and provided the optimal resource allocation for each case. Wu and Wang [10] studied a single-machine scheduling problem with truncated sum-of-processing-times-based learning effect including proportional delivery times, and proposed several scheduling rules to solve this problem. More references can be found in Niu et al. [11], Wang et al. [12], Wu et al. [13], etc. In this paper, we follow the learning model in Niu et al. [11] and extend it into the group scheduling problem considering the batch processing way.

In our previous research, some scheduling problems with the effects of deterioration and learning were investigated [14–17]. Different from our previous research, we mainly focus on the combined effects of deterioration and truncated job-dependent learning in this paper, with group scheduling and group release times further investigated. Thus, the main contributions of this paper can be summarized as follows:

(1) We propose a novel integrated scheduling model which combines the features of serial batching, the combined effects of deterioration and truncated job-dependent learning, group scheduling, and setup time simultaneously.
(2) Specific to the situation of group scheduling, different release times are further investigated on the basis of the batching processing way.
(3) For the special case, we propose the optimal job batching policies, batches sequencing, and groups sequencing and develop an optimization algorithm to solve it. Based on this, an effective hybrid VNS–ASHLO algorithm is developed to solve the general case.

The reminder of this paper is organized as follows. We give notations and the problem statement in Sect. 2. In Sect. 3, the special case that all groups have the same arrival time is analyzed and an optimization algorithm is proposed to solve it. In Sect. 4, the general case where all groups have different arrival times is analyzed, and a hybrid meta-heuristic is proposed to solve it. Finally, the conclusion is given in Sect. 5.

2 Notations and problems statement

We first give the notations used throughout in this paper, which is shown in Table 1.

Table 1 Notations

Notation	Definition
n	The number of groups
G_i	The job set of group i, $i = 1, 2, \cdots, n$
N_i	The number of jobs in G_i, $i = 1, 2, \cdots, n$
N	The total number of jobs, i.e., $N = \sum_{i=1}^{n} N_i$
J_{ij}	Job j in G_i, $i = 1, 2, \cdots, n$, $j = 1, 2, \cdots, N_i$

Table 1 continued

Notation	Definition
α_i	The learning rate of all jobs in $G_i, i = 1, 2, \cdots, n$
β	A truncation parameter
b	The deteriorating rate of processing jobs
p_{ij}	The normal processing time of $J_{ij}, j = 1, 2, \cdots, N_i, i = 1, 2, \cdots, n$
p_{ij}^A	The actual processing time of $J_{ij}, j = 1, 2, \cdots, N_i, i = 1, 2, \cdots, n$
r_i	Release time of all jobs in $G_i, i = 1, 2, \cdots, n$
m_i	The number of batches in $G_i, i = 1, 2, \cdots, n$
b_{ik}	Batch k in $G_i, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
n_{ik}	The number of jobs in $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
θ_b^i	The deteriorating rate of batches' setup time in $G_i, i = 1, 2, \cdots, n$
θ_g	The deteriorating rate of groups' setup time
s_b^{ik}	The setup time of $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
s_g^i	The setup time of $G_i, i = 1, 2, \cdots, n$
c	The capacity of the batching machine
$S(b_{ik})$	The starting time of $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
$C(b_{ik})$	The completion time of $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
$P(b_{ik})$	The actual processing time of $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$
$P(G_i)$	The total actual processing time of $G_i, i = 1, 2, \cdots, n$
C_{max}	The makespan

We start by proposing a combined deterioration and truncated position-based learning model for group scheduling in a serial-batching setting. Here the deterioration effect indicates that the raw materials to be processed deteriorate over time, and the jobs require more processing time if processed later. The learning effect indicates that the workers or machines can improve the production efficiency with more processing experiences, and the coefficient of learning effect is determined by the job's position and a truncation parameter in the truncated position-based learning model. Then, we investigate the following scheduling problem. There is a set of N non-preemptive jobs to be processed on a serial-batching machine, and these jobs are classified into n groups. Each group contains a certain number of jobs, that is, $G_i = \left\{J_{i1}, J_{i2}, \cdots, J_{N_i}\right\}, i = 1, 2, \cdots, n$. All jobs in each group are processed in the way of serial batches, which requires that all the jobs within the same batch are processed one after another in a serial way, and the completion time of any job is equal to that of its belonged batch, which is defined as the completion time of the last job in the batch [18]. The number of the jobs in each batch cannot exceed the machine capacity c. We further investigate the combined effects of deterioration and truncated job-dependent learning in this paper [11]. Due to the switch operations and different processing ways for different groups, the combined effects of deterioration and learning restart once a new group begins to be processed. If J_{ij} is scheduled in position r of certain group G_i, then its actual processing time is defined as [11]

$$p_{ij}^A = p_{ij} max\left\{r^{\alpha_i}, \beta\right\} + bt, \quad r, j = 1, 2, \cdots, N_i, i = 1, 2, \cdots, n$$

where p_{ij} is a normal processing time of J_{ij}, α_i is the learning rate of all jobs in $G_i (i = 1, 2, \cdots, n)$ with $\alpha_i < 0$, β is a truncation parameter with $0 < \beta < 1$, b is the deteriorating rate of processing jobs, t is the staring time for processing J_{ij}.

Both group and batch setup times are required before processing any group or batch, and the setup times of G_i and b_{ik} are defined as follows:

$$s_g^i = \theta_g t$$
$$s_b^{ij} = \theta_b^i t'$$

where θ_g is the deteriorating rate of groups' setup time, θ_b^i is the deteriorating rate of batches' setup time in G_i $(i = 1, 2, \cdots, n)$, and t and t' are the starting time of processing G_i and b_{ik}, respectively.

In this paper, we assume that different groups have distinct release times. We first investigate the special case that all groups have the identical release times and propose some important properties and algorithms, and then study the general case that all groups have different release times based on the investigation to the special case. The objective of the studied problems is to minimize the makespan. In the remaining sections of the paper, all the problems are denoted by the three-field notation schema $\alpha\,|\beta|\,\gamma$ introduced by Graham et al. [19].

3 Problem $1\left|s-batch,\ p_{ij}^A = p_{ij} max\left\{r^{\alpha_i},\beta\right\} + bt,\ r_i = t_0\right| C_{max}$

In this section, the special case that all groups have the same release times is studied. We first give some structural properties of this studied problem for the optimal schedules, and then a batching rule and an optimization algorithm are developed to solve this problem.

We first give the completion time of a certain group in the following property.

Lemma 1 *For the problem* $1\left|s-batch,\ p_{ij}^A = p_{ij} max\left\{r^{\alpha_i},\beta\right\} + bt,\ r_i = t_0\right| C_{max}$*, given any schedule* $\pi = (G_1, G_2, \cdots, G_n)$ *with all groups arriving at time* $t_0 > 0$*, if the starting time of* G_f $(f = 1, 2, \cdots, n)$ *is* T*, then the completion time of* G_f *is*

$$\begin{aligned} C\left(b_{fm_f}\right) = {} & \left(1+\theta_g\right)\left(1+\theta_b^f\right)^{m_f}(1+b)^{N_f}\,T \\ & + \sum_{k=1}^{m_f}(1+b)^{\sum_{d=k+1}^{m_f} n_{fd}}\left(1+\theta_b^f\right)^{m_f-k} \sum_{j=\sum_{h=1}^{k-1} n_{fh}+1}^{\sum_{h=1}^{k} n_{fh}} \\ & (1+b)^{\sum_{h=1}^{k} n_{fh}-j}\, p_{fj} max\left\{j^{\alpha_f},\beta\right\} \end{aligned} \tag{1}$$

Proof For the batch index $v = 1$ in G_f, there is

$$C\left(b_{f1}\right) = \left(1+\theta_g\right)\left(1+\theta_b^f\right)(1+b)^{n_{f1}}\,T + \sum_{j=1}^{n_{f1}}(1+b)^{n_{f1}-j}\, p_{fj} max\left\{j^{\alpha_f},\beta\right\}$$

Thus, Eq. (1) holds for $v = 1$. For all $2 \le v < f$, if Eq. (1) holds, then

$$\begin{aligned} C\left(b_{fv}\right) = {} & \left(1+\theta_g\right)\left(1+\theta_b^f\right)^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\,T \\ & + \sum_{k=1}^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\left(1+\theta_b^f\right)^{v-k} \sum_{j=\sum_{h=1}^{k-1} n_{fh}+1}^{\sum_{h=1}^{k} n_{fh}} \\ & (1+b)^{\sum_{h=1}^{k} n_{fh}-j}\, p_{fj} max\left\{j^{\alpha_f},\beta\right\}. \end{aligned}$$

Furthermore, for the $(v+1)$th batch $b_{f(v+1)}$, there is

$$\begin{aligned} C\left(b_{f(v+1)}\right) &= C\left(b_{fv}\right)\left(1+\theta_b^f\right)(1+b)^{n_{f(v+1)}} \\ &\quad + \sum_{j=\sum_{h=1}^{k-1} n_{fv}+1}^{\sum_{h=1}^{k} n_{f(v+1)}} (1+b)^{\sum_{h=1}^{k} n_{f(v+1)}-j}\, p_{fj} max\left\{j^{\alpha_f}, \beta\right\} \\ &= \left(1+\theta_g\right)\left(1+\theta_b^f\right)^{v+1}(1+b)^{\sum_{d=k+1}^{v+1} n_{fd}}\, T + \sum_{k=1}^{v+1}(1+b)^{\sum_{d=k+1}^{v+1} n_{fd}} \\ &\quad \times \left(1+\theta_b^f\right)^{v-k} \sum_{j=\sum_{h=1}^{k-1} n_{fh}+1}^{\sum_{h=1}^{k} n_{fh}} (1+b)^{\sum_{h=1}^{k} n_{fh}-j}\, p_{fj} max\left\{j^{\alpha_f}, \beta\right\}. \end{aligned}$$

Thus, Eq. (1) holds for the $(v+1)$th batch $b_{f(v+1)}$, and it can be also derived that Eq. (1) holds for b_{fm_f}. The proof is completed. □

Based on the result of the batches' completion times, we develop some properties of jobs sequencing and jobs batching in the same batch from a certain batch by the job interchange operations as follows.

Lemma 2 *For the problem* $1\left|s-batch, p_{ij}^A = p_{ij} max\left\{r^{\alpha_i}, \beta\right\} + bt, r_i = t_0\right| C_{max}$, *all jobs in the same batch of a certain group should be sequenced in non-decreasing order of* p_{ij} *in the optimal schedule.*

Proof Here we assume that π^* and π are an optimal schedule and a job schedule, respectively. The difference of these two schedules is the pairwise interchange of these two jobs J_{fd} and $J_{f(d+1)}$ $\left(d = 1, 2, \cdots, N_f - 1\right)$ in the same batch, that is, $\pi^* = \left(W_1, J_{fd}, J_{f(d+1)}, W_2\right)$, $\pi = \left(W_1, J_{fd}, J_{f(d+1)}, W_2\right)$, where $J_{fd}, J_{f(d+1)} \in b_{fv}$, and $b_{fv} \subset G_f$, $n_{fv} \geq 2$, $f = 1, 2, \cdots, n$, $v = 1, 2, \cdots, m_f$. J_{fd} and $J_{f(d+1)}$ are in the dth and $(d+1)$th positions of G_f. W_1 and W_2 represent two partial sequences, and W_1 or W_2 may be empty. It is assumed that $p_{fd} \geq p_{f(d+1)}$.

If the starting time of G_f $(f = 1, 2, \cdots, n)$ is T, then we first give the completion time of b_{fv} in π^*,

$$\begin{aligned} C\left(b_{fv}\left(\pi^*\right)\right) &= \left(1+\theta_g\right)\left(1+\theta_b^f\right)^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\, T \\ &\quad + \sum_{k=1}^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\left(1+\theta_b^f\right)^{v-k} \sum_{j=\sum_{h=1}^{k-1} n_{fh}+1}^{\sum_{h=1}^{k} n_{fh}} \\ &\quad (1+b)^{\sum_{h=1}^{k} n_{fh}-j}\, p_{fj} max\left\{j^{\alpha_f}, \beta\right\} \end{aligned}$$

Then, the completion time of b_{fv} in π is

$$\begin{aligned} C\left(b_{fv}(\pi)\right) &= \left(1+\theta_g\right)\left(1+\theta_b^f\right)^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\, T \\ &\quad + \sum_{k=1}^{v}(1+b)^{\sum_{d=k+1}^{v} n_{fd}}\left(1+\theta_b^f\right)^{v-k} \sum_{j=\sum_{h=1}^{k-1} n_{fh}+1}^{\sum_{h=1}^{k} n_{fh}} (1 \end{aligned}$$

$$
\begin{aligned}
&+b)^{\sum_{h=1}^{k} n_{fh}-j} p_{fj} max\left\{j^{\alpha_f}, \beta\right\} \\
&-\left[(1+b)^{n_{fv}-d} p_{fd} max\left\{d^{\alpha_f}, \beta\right\}+(1+b)^{n_{fv}-d-1} p_{f(d+1)} max\left\{(d+1)^{\alpha_f}, \beta\right\}\right] \\
&+\left[(1+b)^{n_{fv}-d} p_{f(d+1)} max\left\{d^{\alpha_f}, \beta\right\}+(1+b)^{n_{fv}-d-1} p_{fd} max\left\{(d+1)^{\alpha_f}, \beta\right\}\right].
\end{aligned}
$$

Consequently,

$$
\begin{aligned}
&C\left(b_{fv}\left(\pi^{*}\right)\right)-C\left(b_{fv}(\pi)\right) \\
&=\left[(1+b)^{n_{fv}-d} p_{fd} max\left\{d^{\alpha_f}, \beta\right\}+(1+b)^{n_{fv}-d-1} p_{f(d+1)} max\left\{(d+1)^{\alpha_f}, \beta\right\}\right] \\
&\quad-\left[(1+b)^{n_{fv}-d} p_{f(d+1)} max\left\{d^{\alpha_f}, \beta\right\}+(1+b)^{n_{fv}-d-1} p_{fd} max\left\{(d+1)^{\alpha_f}, \beta\right\}\right] \\
&=\left[(1+b)^{n_{fv}-d} max\left\{d^{\alpha_f}, \beta\right\}-(1+b)^{n_{fv}-d-1} max\left\{(d+1)^{\alpha_f}, \beta\right\}\right]\left[p_{fd}-p_{f(d+1)}\right].
\end{aligned}
$$

Since $max\left\{d^{\alpha_f}, \beta\right\} \geq max\left\{(d+1)^{\alpha_f}, \beta\right\}$, we can obtain that $(1+b)^{n_{fv}-d} max\left\{d^{\alpha_f}, \beta\right\} > (1+b)^{n_{fv}-d-1} max\left\{(d+1)^{\alpha_f}, \beta\right\}$. Also, it is assumed $p_{fd} \geq p_{f(d+1)}$, we can derive that

$$
C\left(b_{fv}\left(\pi^{*}\right)\right) \geq C\left(b_{fv}(\pi)\right),
$$

which conflicts with the optimal schedule. Hence, it should be $p_{fd} \leq p_{f(d+1)}$.

The proof is completed. □

Similar to the proof of Lemma 2, we have the following property.

Lemma 3 *For the optimal schedule of the problem* $1\left|s-batch, p_{ij}^{A}=p_{ij} max\left\{r^{\alpha_i}, \beta\right\}+bt, r_i=t_0\right| C_{max}$, *the normal processing time of* J_{ij} *in a batch* b_{fv} *should be no more than that of any jobs in* $b_{f(v+1)}$, $f=1,2,\cdots,n$, $v=1,2,\cdots,m_f-1$.

Based on the jobs sequencing characteristic of Lemmas 2 and 3, we obtain the following corollary.

Corollary 1 *For the problem* $1\left|s-batch, p_{ij}^{A}=p_{ij} max\left\{r^{\alpha_i}, \beta\right\}+bt, r_i=t_0\right| C_{max}$, *all jobs in a certain group should be sequenced in the non-decreasing order of* p_{ij}.

As in Lemma 2, we can also use the similar jobs transferring operations to obtain the following property.

Lemma 4 *For the problem* $1\left|s-batch, p_{ij}^{A}=p_{ij} max\left\{r^{\alpha_i}, \beta\right\}+bt, r_i=t_0\right| C_{max}$, *there should be* $n_{fv} \leq n_{f(v+1)}$ *for a certain group in the optimal schedule, where* $f=1,2,\cdots,n$, $v=1,2,\cdots,m_f-1$.

The following property for jobs number argument can be further obtained by using the similar jobs transferring operations.

Lemma 5 *For the optimal schedule of the problem* $1\left|s-batch, p_{ij}^{A}=p_{ij} max\left\{r^{\alpha_i}, \beta\right\}\right.$ $\left.+bt, r_i=t_0\right| C_{max}$, *there should be* $\left\lceil \frac{N_i}{c} \right\rceil$ *batches in any group* G_i, *and all batches are full of jobs except possibly the first batch.*

For the problem $1\left|s-batch, p_{ij}^{A}=p_{ij}max\left\{r^{\alpha_i},\beta\right\}+bt, r_i=t_0\right|C_{max}$, we develop Batching-Rule 1 for the optimal jobs batching of each group based on the Lemmas 1–5 and Corollary 1. Based on the optimal jobs batching of each group, we have the following

Batching-Rule 1

Step 1. Set $i=1$
Step 2. All jobs in G_i are indexed in the non-decreasing order of p_{ij}, $j=1,2,\cdots,m_i$, then a job list of G_i is generated that $p_{i1}\le p_{i2}\le\cdots\le p_{im_i}$.
Step 4. Place the first $N_i-\left(\left\lceil\frac{N_i}{c}\right\rceil-1\right)c$ jobs in the first batch
Step 5. If there are more than c jobs in the job list of G_i, then place c jobs in a batch and iterate. Then, all batches are generated in G_i.
Step 6. If $i<n$, then set $i=i+1$, go to step 2. Otherwise, end.

property for the optimal sequencing of each group.

Lemma 6 *For the problem* $1\left|s-batch, p_{ij}^{A}=p_{ij}max\left\{r^{\alpha_i},\beta\right\}+bt, r_i=t_0\right|C_{max}$, *considering two consecutive groups* G_r *and* G_{r+1}, *if* $\rho\left(G_r\right)\le\rho\left(G_{r+1}\right)$, *where* $\rho\left(G_r\right)=\frac{\sum_{k=1}^{m_r}(1+b)^{\sum_{d=k+1}^{m_r}n_{rd}}\left(1+\theta_b^r\right)^{m_r-k}\sum_{j=\sum_{h=1}^{k-1}n_{rh}+1}^{\sum_{h=1}^{k}n_{rh}}(1+b)^{\sum_{h=1}^{k}n_{rh}-j}p_{rj}max\{j^{\alpha_r},\beta\}}{\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}(1+b)^{N_r}-1}$, $r=1,2,\cdots,n-1$, *then it is optimal to process* G_r *before* G_{r+1}.

Proof Let $.\pi^*.$ and π be an optimal schedule and a job schedule, and their difference is the pairwise interchange of these two job sets G_r and G_{r+1} $(r=1,2,\cdots,n-1)$, that is, $\pi^*=(W_1,G_r,G_{r+1},W_2)$, $\pi=(W_1,G_{r+1},G_r,W_2)$, where both G_r and G_{r+1} may include one or multiple batches, W_1 and W_2 represent two partial sequences, and W_1 or W_2 may be empty. Here it is assumed that $\rho\left(G_r\right)>\rho\left(G_{r+1}\right)$, i.e.,

$$\frac{\sum_{k=1}^{m_r}(1+b)^{\sum_{d=k+1}^{m_r}n_{rd}}\left(1+\theta_b^r\right)^{m_r-k}\sum_{j=\sum_{h=1}^{k-1}n_{rh}+1}^{\sum_{h=1}^{k}n_{rh}}(1+b)^{\sum_{h=1}^{k}n_{rh}-j}p_{rj}max\left\{j^{\alpha_r},\beta\right\}}{\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}(1+b)^{N_r}-1}$$
$$>\frac{\sum_{k=1}^{m_{r+1}}(1+b)^{\sum_{d=k+1}^{m_{r+1}}n_{(r+1)d}}\left(1+\theta_b^{r+1}\right)^{m_{r+1}-k}\sum_{j=\sum_{h=1}^{k-1}n_{(r+1)h}+1}^{\sum_{h=1}^{k}n_{(r+1)h}}(1+b)^{\sum_{h=1}^{k}n_{(r+1)h}-j}p_{(r+1)j}max\left\{j^{\alpha_{r+1}},\beta\right\}}{\left(1+\theta_g\right)\left(1+\theta_b^{r+1}\right)^{m_{r+1}}(1+b)^{N_{r+1}}-1},$$

and the starting time of processing G_r is T.

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

$$\begin{aligned}C\left(G_{r+1}\left(\pi^*\right)\right)=&\left(1+\theta_g\right)\left(1+\theta_b^{r+1}\right)^{m_{r+1}}(1+b)^{N_{r+1}}\\&\left[\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}(1+b)^{N_r}T+\sum_{k=1}^{m_r}(1+b)^{\sum_{d=k+1}^{m_r}n_{rd}}\right.\\&\left.\times\left(1+\theta_b^r\right)^{m_r-k}\sum_{j=\sum_{h=1}^{k-1}n_{rh}+1}^{\sum_{h=1}^{k}n_{rh}}(1+b)^{\sum_{h=1}^{k}n_{rh}-j}p_{rj}max\left\{j^{\alpha_r},\beta\right\}\right]\\&+\sum_{k=1}^{m_{r+1}}(1+b)^{\sum_{d=k+1}^{m_{r+1}}n_{(r+1)d}}\left(1+\theta_b^{r+1}\right)^{m_{r+1}-k}\sum_{j=\sum_{h=1}^{k-1}n_{(r+1)h}+1}^{\sum_{h=1}^{k}n_{(r+1)h}}\end{aligned}$$

$$\times \left(1+b\right)^{\sum_{h=1}^{k} n_{(r+1)h}-j} p_{(r+1)j} max\left\{j^{\alpha_{r+1}}, \beta\right\}.$$

For π, the completion time of G_r is

$$\begin{aligned} C\left(G_r\left(\pi\right)\right) = & \left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}\left(1+b\right)^{N_r} \\ & \left[\left(1+\theta_g\right)\left(1+\theta_b^{r+1}\right)^{m_{r+1}}\left(1+b\right)^{N_{r+1}} T + \sum_{k=1}^{m_{r+1}} \left(1+b\right)^{\sum_{d=k+1}^{m_{r+1}} n_{(r+1)d}}\right. \\ & \left.\times\left(1+\theta_b^{r+1}\right)^{m_{r+1}-k} \sum_{j=\sum_{h=1}^{k-1} n_{(r+1)h}+1}^{\sum_{h=1}^{k} n_{(r+1)h}} \left(1+b\right)^{\sum_{h=1}^{k} n_{(r+1)h}-j} p_{(r+1)j} max\left\{j^{\alpha_{r+1}}, \beta\right\}\right] \\ & + \sum_{k=1}^{m_r}\left(1+b\right)^{\sum_{d=k+1}^{m_r} n_{rd}}\left(1+\theta_b^r\right)^{m_r-k} \sum_{j=\sum_{h=1}^{k-1} n_{rh}+1}^{\sum_{h=1}^{k} n_{rh}} \left(1+b\right)^{\sum_{h=1}^{k} n_{rh}-j} p_{rj} max\left\{j^{\alpha_r}, \beta\right\}. \end{aligned}$$

It can be derived that

$$\begin{aligned} & C\left(G_{r+1}\left(\pi^*\right)\right) - C\left(G_r\left(\pi\right)\right) \\ & = \left[\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}\left(1+b\right)^{N_r} - 1\right]\left[\left(1+\theta_g\right)\left(1+\theta_b^{r+1}\right)^{m_{r+1}}\left(1+b\right)^{N_{r+1}} - 1\right] \\ & \left[\frac{\sum_{k=1}^{m_r}\left(1+b\right)^{\sum_{d=k+1}^{m_r} n_{rd}}\left(1+\theta_b^r\right)^{m_r-k}\sum_{j=\sum_{h=1}^{k-1} n_{rh}+1}^{\sum_{h=1}^{k} n_{rh}}\left(1+b\right)^{\sum_{h=1}^{k} n_{rh}-j} p_{rj} max\left\{j^{\alpha_r}, \beta\right\}}{\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}\left(1+b\right)^{N_r}-1}\right. \\ & \left. - \frac{\sum_{k=1}^{m_{r+1}}\left(1+b\right)^{\sum_{d=k+1}^{m_{r+1}} n_{(r+1)d}}\left(1+\theta_b^{r+1}\right)^{m_{r+1}-k}\sum_{j=\sum_{h=1}^{k-1} n_{(r+1)h}+1}^{\sum_{h=1}^{k} n_{(r+1)h}}\left(1+b\right)^{\sum_{h=1}^{k} n_{(r+1)h}-j} p_{(r+1)j} max\left\{j^{\alpha_{r+1}}, \beta\right\}}{\left(1+\theta_g\right)\left(1+\theta_b^{r+1}\right)^{m_{r+1}}\left(1+b\right)^{N_{r+1}}-1}\right] \\ & > 0. \end{aligned}$$

It conflicts with the optimal schedule. Consequently, the proof is completed. □

Based on the above lemmas and the optimal jobs batching rule, we develop the following Algorithm 1 to solve the problem $1\left|s-batch, p_{ij}^A = p_{ij} max\left\{r^{\alpha_i}, \beta\right\} + bt, r_i = t_0\right| C_{max}$.

Algorithm 1

Step 1. Execute Batching-Rule 1

Step 2. Calculate $\rho\left(G_r\right) = \dfrac{\sum_{k=1}^{m_r}\left(1+b\right)^{\sum_{d=k+1}^{m_r} n_{rd}}\left(1+\theta_b^r\right)^{m_r-k}\sum_{j=\sum_{h=1}^{k-1} n_{rh}+1}^{\sum_{h=1}^{k} n_{rh}}\left(1+b\right)^{\sum_{h=1}^{k} n_{rh}-j} p_{rj} max\left\{j^{\alpha_r}, \beta\right\}}{\left(1+\theta_g\right)\left(1+\theta_b^r\right)^{m_r}\left(1+b\right)^{N_r}-1}$,

$r = 1, 2, \cdots, n-1$.

Step 3. Sequence all groups in the non-increasing order of $\rho\left(G_l\right)$, i.e., $\rho\left(G_1\right) \le \rho\left(G_2\right) \le \cdots \le \rho\left(G_n\right)$.

Theorem 1 *For the problem* $1\left|s-batch, p_{ij}^A = p_{ij} max\left\{r^{\alpha_i}, \beta\right\} + bt, r_i = t_0\right| C_{max}$, *an optimal schedule can be obtained by Algorithm 1 in* $O\left(N \log N\right)$ *time. The optimal makespan is*

$$
\begin{aligned}
C^*_{max} = {} & t_0 \left(1+\theta_g\right)^n \prod_{r=1}^{n} \left(1+\theta_b^r\right)^{m_r} (1+b)^{N_r} \\
& + \sum_{r=1}^{n} \left(1+\theta_g\right)^{n-r} \sum_{k=1}^{m_r} (1+b)^{\sum_{d=k+1}^{m_r} n_{rd}} \left(1+\theta_b^r\right)^{m_r-k} \\
& \sum_{j=\sum_{h=1}^{k-1} n_{rh}+1}^{\sum_{h=1}^{k} n_{rh}} (1+b)^{\sum_{h=1}^{k} n_{rh}-j} \, p_{rj} max\left\{j^{\alpha_r}, \beta\right\} \prod_{l=r+1}^{n} \left(1+\theta_b^l\right)^{m_l} (1+b)^{N_l} .
\end{aligned} \tag{2}
$$

Proof An optimal solution can be generated by Algorithm 1 based on Lemmas 1–6 and Corollary 1. Similar to the proof of Lemma 1, the result of the optimal solution can be also obtained as Eq. (2). The time complexity of step 1 is at most $O\left(N \log N\right)$, and the total time complexity of step 2 is $O\left(1\right)$. Then, for step 3, the time complexity of obtaining the optimal group sequence is $O\left(n \log n\right)$. Since we have $n \leq N$, the time complexity of Algorithm 1 is at most $O\left(N \log N\right)$. □

4 Problem $1\left|s-batch, p_{ij}^A = p_{ij} max\left\{r^{\alpha_i}, \beta\right\} + bt, r_i\right| C_{max}$

In this section, we first give the key steps of VNS–ASHLO algorithm in Sect. 4.1, and then computational experiments are conducted to test the performance of the proposed algorithm compared with another four algorithms in Sect. 4.2.

4.1 Key steps of VNS–ASHLO algorithm

In this section, a hybrid VNS–ASHLO algorithm combing variable neighborhood search (VNS) and adaptive simplified human learning optimization (ASHLO) algorithm is proposed to solve the studied problem. VNS has been widely used in various combinatorial optimization problems since it was developed by Hansen and Mladenović [20]. There are lots of variants of VNS and the detailed overview can be found in Hansen et al. [21]. In order to enhance the effectiveness of the local search procedure in VNS, we adopt the the adaptive simplified human learning optimization (ASHLO) algorithm to replace the local search procedure in traditional VNS. ASHLO algorithm was proposed by Wang et al. [22], which is inspired by the behavior of human learning. The human learning process can be divided into three methods: (i) random learning, (ii) individual learning, and (iii) social learning. Random learning is common at the beginning of learning, because of the lack of prior knowledge, and individual usually acquires knowledge randomly [22]. In the following studying process, to avoid mistakes and improve learning efficiency, individuals refer to their own experiences and knowledge during the process of study [22]. This phenomenon can be abstracted as individual learning. However, the experiences or knowledge of an individual is limited, individual needs to gain more knowledge from other people through social learning to further improve their learning performance [22]. In each iteration, individual i randomly selects a method to learn and then updates current individual knowledge data (IKD) for itself and social knowledge data (SKD), where IKD is used to save a certain number of historical optimal solutions for each individual, and SKD is uesd to store a certain number of historical optimal solutions for the whole population. The algorithm framework of VNS–ASHLO is described in Table 2, and the part of ASHLO is demonstrated in the 5th line to the 23th line in the pseudocode. The flow chart of VNS–ASHLO is also given in Fig. 1.

Table 2 The pseudocode of VNS–ASHLO

Pseudocode of VNS-ASHLO	
1.	Generate initial solution $X = \{x_1, \cdots, x_g, \cdots, x_n\}$ randomly, set pr, pi, max_it, $hmax$, $hmin$, and $hstep$
2.	Set $h = hmin$ and $it = 0$, randomly generate IKD for each individual and SKD from $N_h(X)$
3.	While ($h \leq hmax$)
4.	Generate N_pop solutions $pop = \{X_1, \cdots, X_l, \cdots X_{N\_pop}\}$ from $N_h(X)$ randomly, and each solution includes n elements, $X_l = \{x_{l1}, \cdots, x_{lg}, \cdots, x_{ln}\}$
5.	While ($it \leq max\_it$)
6.	For each X_l, $l = 1$ to N_pop
7.	For each x_{lg}, $g = 1$ to n
8.	Generate a random number $rand()$ in [0,1]
9.	If $rand() \leq pr$ then
10.	Perform random learning operator for x_{lg}
11.	else
12.	If $rand() \leq pi$ then
13.	Perform individual learning operator for x_{lg}
14.	else
15.	Perform social learning operator for x_{lg}
16.	End if
17.	End if
18.	End for
19.	Update IKD for individual l
20.	End for
21.	Update SKD, pr and pi
22.	$it++$
23.	End while
24.	Obtain best solution X' in SKD
25.	If X' is better than X, then
26.	Replace X with X'
27.	$h = hmin$
28.	else
29.	$h = h + hstep$
30.	End if
31.	End while
32.	Output X

4.1.1 Coding and encoding

In this paper, a solution of the studied problem is resprented by an array of group number, and the processing sequence of groups is determined by the array. For any solution $X = \{x_1, \cdots, x_g, \cdots, x_n\}$, the fitness of X is calculated as follows.

Calculation of the fitness

Step 1. Set $g = 1$, $C_{g-1} = C_g = 0$.
Step 2. Apply Scheduling Rule 1 to calculate the completion time of group x_g as C_g, if $g = n$, then output C_n and stop.
Step 3. If $C_g < max\{r_i | i = 1, 2 \cdots, n\}$, then set $g = g + 1$ and go to step 2. Otherwise, go to step 4.
Step 4. Apply algorithm 1 to solve the completion time of the remain groups and output C_n.

4.1.2 Neighborhood structure

A simple neighborhood structure based on swap operator is applied in this paper, $N_h(X)$ denotes the hth neighborhood of solution X, and $N_h(X)$ is defined as follows.

Neighborhood structure

Step 1. Set $u = 1$.
Step 2. Randomly select two elements of solution X and swap them.
Step 3. If $u \leq h$, then go to step 2. Otherwise, output solution X.

4.1.3 Random learning operator

A simple random swap process is developed as random learning operator. When an element of solution X is selected to perform random learning operator, randomly swap this element with another element of solution X.

4.1.4 Individual learning operator

We assume that the gth position of current solution X for individual l is taken into consideration, and the individual learning operator can be described as follows.

Individual learning operator

Step 1. Randomly select a solution from the IKD of individual l as the selected solution X_select.
Step 2. The element of X which is equal to the one in the gth position of X_select is replaced by the gth position of X.
Step 3. Replace gth position of current solution X with the one in the gth position of X_select.

4.1.5 Social learning operator

Similar to individual learning operator, we also define the social learning operator, which can be described as follows.

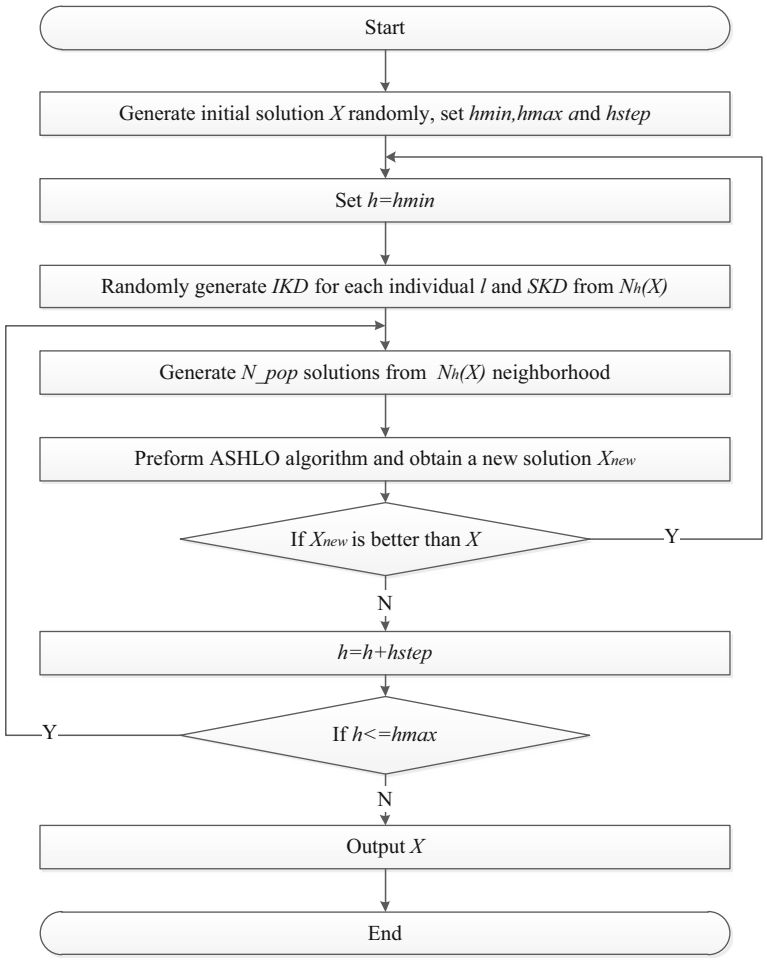

Fig. 1 The flowchart of VNS–ASHLO

Social learning operator

Step 1. Randomly select a solution from the SKD as selected solution X_select.
Step 2. The element of X which is equal to the one in the gth position of X_select is replaced by the gth position of X.
Step 3. Replace gth position of current solution X with the one in the gth position of X_select.

4.2 Computational experiments and comparison

In this sub-section, a serial of computational experiments are conducted to test the performance of our proposed algorithm VNS–ASHLO, compared with ASHLO [22], VNS [23], and Simulated Annealing (SA) [24], Particle Swarm Optimization (PSO) [25]. The parameters of the test problems were randomly generated as Table 3 according to the practical situations in an aluminum factory.

Table 3 Parameters setting

Notation	Definition	Value
n	The number of groups	10,20,30,40,50
N_i	The number of jobs in $G_i, i = 1, 2, \cdots, n$	U[5,15]
α_i	The learning rate of all jobs in $G_i, i = 1, 2, \cdots, n$	U[-1,-3]
β	A truncation parameter	U[0,0.5]
b	The deteriorating rate of processing jobs	U[0,0.1]
r_i	Release time of all jobs in $G_i, i = 1, 2, \cdots, n$	U[0,4000]
p_{ij}	The normal processing time of $J_{ij}, j = 1, 2, \cdots, N_i, i = 1, 2, \cdots, n$	U[0,20]
n_{ik}	The number of jobs in $b_{ik}, k = 1, 2, \cdots, m_i, i = 1, 2, \cdots, n$	U[0,10]
θ_b^i	The deteriorating rate of batches' setup time in $G_i, i = 1, 2, \cdots, n$	U[0,0.1]
θ_g	The deteriorating rate of groups' setup time	U[0,0.1]
c	The capacity of the batching machine	U[3,5]

In order to evaluate the performance of proposed VNS–ASHLO, it is compared with another four algorithms in the studied problem. In Table 4, the results of the average objective value (Avg.Obj) and the maximum objective value (Max.Obj) for the problem are listed. The convergence curves of all algorithms are shown in Fig. 2.

All cases run 10 times to avoid the contingency of the experiment. In order to ensure the fairness of the comparison experiment, the population sizes of ASHLO, PSO, and VNS–ASHLO are equal to the local search times of VNS and SA, both set to 5. All algorithms were implemented in Eclipse and run on a Lenovo computer running Window10 with a dual-core CPU Intel i3-3240@3.40 GHz and 4 GB RAM. All the algorithms are tested by 200 iterations in a reasonable time, and the program code runs in 9.5 sec for 10 times when $n = 45$. Thus, it is shown that the average running time of VNS–ASHLO does not exceed 1 second. From Table 4, we conclude that each algorithm can find the optimal solution of the problem within 200 iterations in most cases. It is easy to find that our proposed algorithm has better performance than other algorithms, since the average objective value obtained by VNS–ASHLO is better than those of other algorithms among all cases. In this experiment, the average result of the current optimal solutions for 1 to 200 iterations is used to plot the above convergence curves. From Fig. 2, it can be obtained that the VNS–ASHLO has better convergence rate and optimization capability than other algorithms. All algorithms can find reasonable solutions within 200 iterations in most cases, but VNS–ASHLO has better convergence rate than other algorithms. With the increasing number of groups, the results show that the optimal solution could not be found within 200 iterations by SA and VNS. Although ASHLO and PSO can obtain better solutions than SA and VNS within 200 iterations, the drawback of them is that they are not stable for all the cases. From Fig. 2, it is also obvious that VNS–ASHLO has better robustness than ASHLO and PSO.

Table 4 The results of the average objective value (Avg.Obj) and the minimum objective value (Max.Obj) for each algorithm

No.	n	SA		PSO		ASHLO		VNS		VNS–ASHLO	
		Ave.Obj	Max.Obj	Ave.Obj	Max.Obj	Ave.Obj	Max.Obj	Ave.Obj	Max.Obj	Ave.Obj	Max.Obj
1	10	1722.73	2709.13	1721.20	2709.13	1721.25	2709.13	1723.03	2709.13	1721.20	2709.13
2	15	4967.39	8323.29	4970.06	8323.29	4967.34	8323.29	4968.64	8323.29	4967.03	8323.29
3	20	33534.07	57219.39	33654.07	57219.39	33534.55	57219.39	33542.63	57219.39	33533.97	57219.39
4	25	78634.99	159976.43	79604.13	159976.08	78706.38	159968.40	78725.50	159980.26	78587.86	159968.40
5	30	246921.04	376997.20	247612.82	376997.20	246808.48	376997.20	247373.04	376997.20	246791.36	376997.20
6	35	1877355.06	3000278.64	1874938.04	3000283.34	1873534.12	3000278.64	1876454.94	3000278.96	1873514.89	3000278.64
7	40	5066428.44	6835307.01	5056925.93	6835407.14	5056879.40	6835307.01	5056990.32	6835308.89	5056839.30	6835306.95
8	45	11334349.56	16250500.41	11334928.72	16250500.41	11334427.31	16250500.41	11334966.90	16250500.41	11334315.07	16250500.41

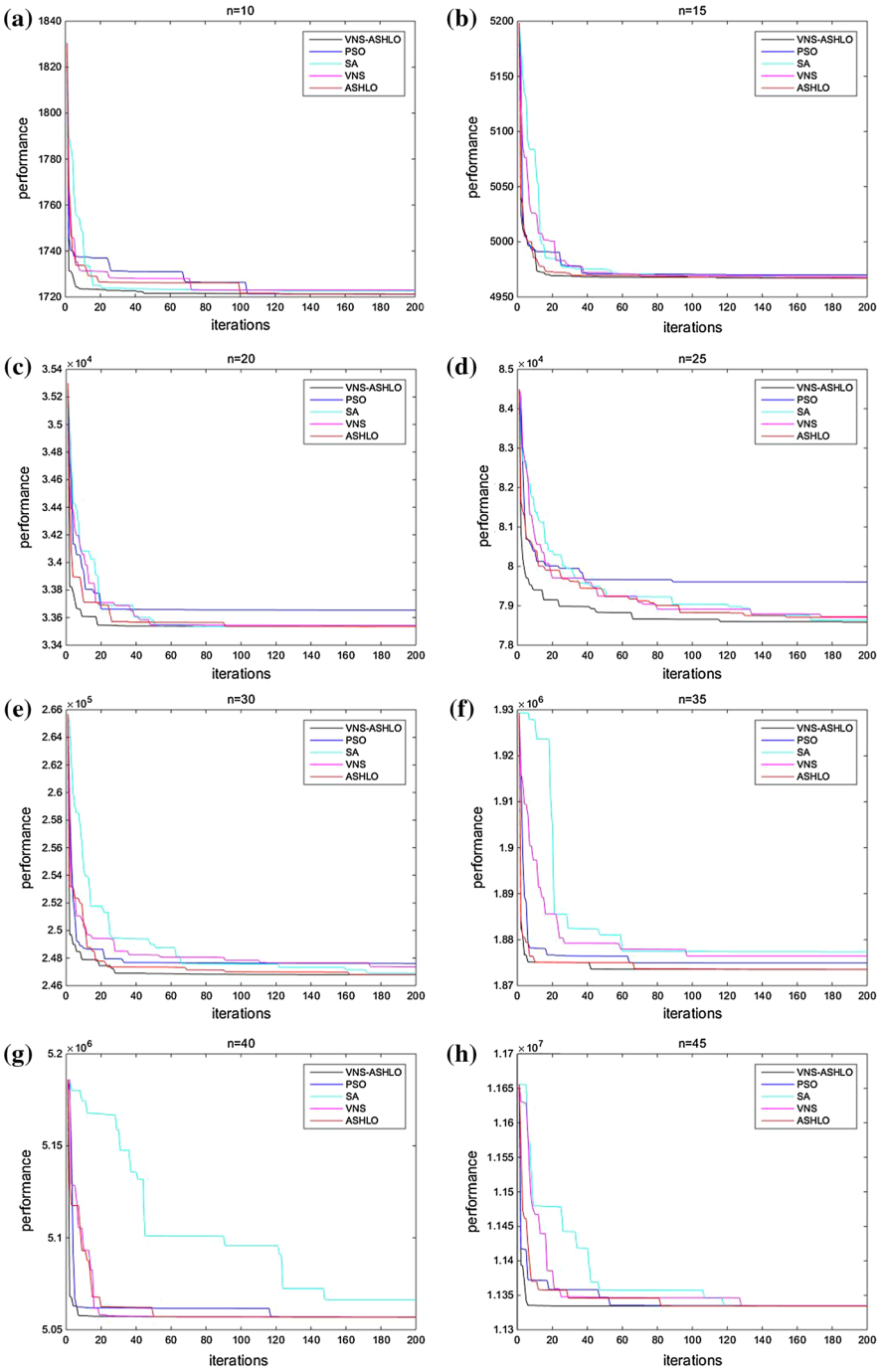

Fig. 2 Convergence curves for each algorithm when $n = 10, 15, 20, 25, 30, 35, 40, 45$. **a** Convergence curves for $n = 10$. **b** Convergence curves for $n = 15$. **c** Convergence curves for $n = 20$. **d** Convergence curves for $n = 25$. **e** Convergence curves for $n = 30$. **f** Convergence curves for $n = 35$. **g** Convergence curves for $n = 40$. **h** Convergence curves for $n = 45$

5 Conclusions

In this paper we study a single serial-batching machine scheduling problem to minimize the makespan, where the features of release times, group scheduling, the combined effects of deterioration and truncated job-dependent learning, and setup time are investigated simultaneously. For the special case that all groups have the same release times, the structural properties on jobs sequencing, jobs batching, and batches sequencing are studied, and an optimal batching rule and an algorithm are proposed for the special case. Based on the structural properties, the general case can be transformed into the resource allocation problem. Then, a hybrid VNS–ASHLO algorithm incorporating VNS and ASHLO algorithms is developed to solve the general case. The results of computational experiments on randomly generated instances show the effectiveness and efficiency of the proposed algorithm, compared with the algorithms of VNS, ASHLO, SA, and PSO.

Several promising directions can be further studied for future research. A possible research is to explore the different deterioration and learning effects according to different production situations. Moreover, other objective functions can be considered in the model, such as minimizing maximum lateness and minimizing the sum of the completion times, to accommodate more practical applications.

Acknowledgements This work is supported by the National Natural Science Foundation of China (Nos. 71501058, 71601065, 71231004, 71690235, 71690230, 71671055), 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), and Anhui Province Natural Science Foundation (No. 1608085QG167). Panos M. Pardalos is partially supported by the project of "Distinguished International Professor by the Chinese Ministry of Education" (MS2014HFGY026).

References

1. Wang, J.B., Jiang, Y., Wang, G.: Single-machine scheduling with past-sequence-dependent setup times and effects of deterioration and learning. Int. J. Adv. Manuf. Technol. **41**(11–12), 1221–1226 (2009)
2. Wang, J.B., Gao, W.J., Wang, L.Y., Wang, D.: Single machine group scheduling with general linear deterioration to minimize the makespan. Int. J. Adv. Manuf. Technol. **43**, 146–150 (2009)
3. Cheng, T.C.E., Lee, W.-C., Wu, C.-C.: Scheduling problems with deteriorating jobs and learning effects including proportional setup times. Comput. Ind. Eng. **58**(2), 326–331 (2010)
4. Kuo, W.-H.: Single-machine group scheduling with time-dependent learning effect and position-based setup time learning effect. Ann. Oper. Res. **196**, 349–359 (2012)
5. Wang, J.B., Huang, X., Wu, Y.B., Ji, P.: Group scheduling with independent setup times, ready times, and deteriorating job processing times. Int. J. Adv. Manuf. Technol. **60**, 643–649 (2012)
6. Yang, S.-J.: Unrelated parallel-machine scheduling with deterioration effects and deteriorating multi-maintenance activities for minimizing the total completion time. Appl. Math. Model. **37**, 2995–3005 (2013)
7. Yang, S.-W., Wan, L., Yin, N.: Research on single machine SLK/DIF due window assignment problem with learning effect and deteriorating jobs. Appl. Math. Model. **39**, 4593–4598 (2015)
8. Wu, C.-C., Wu, W.-H., Hsu, P.-H., Lai, K.: A two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function. Appl. Math. Model. **36**, 5001–5014 (2012)
9. He, H., Liu, M., Wang, J.B.: Resource constrained scheduling with general truncated job-dependent learning effect. J. Comb. Optim. (2015). doi:10.1007/s10878-015-9984-5
10. Wu, Y.-B., Wang, J.-J.: Single-machine scheduling with truncated sum-of-processing-times-based learning effect including proportional delivery times. Neural. Comput. Appl. **27**, 937–943 (2016)
11. Niu, Y.-P., Wang, J., Yin, N.: Scheduling problems with effects of deterioration and truncated job-dependent learning. J. Appl. Math. Comput. **47**, 315–325 (2015)
12. Wang, J.B., Wang, X.-Y., Sun, L.-H., Sun, L.-Y.: Scheduling jobs with truncated exponential learning functions. Optim. Lett. **7**, 1857–1873 (2013)

13. Wu, C.-C., Yin, Y., Cheng, S.-R.: Single-machine and two-machine flowshop scheduling problems with truncated position-based learning functions. J. Oper. Res. Soc. **64**, 147–156 (2013)
14. Pei, J., Liu, X., Pardalos, P.M., Migdalas, A., Yang, S.: Serial-batching scheduling with time-dependent setup time and effects of deterioration and learning on a single-machine. J. Glob. Optim. **67**(1), 251–262 (2017)
15. Pei, J., Pardalos, P.M., Liu, X., Fan, W., Yang, S.: Serial batching scheduling of deteriorating jobs in a two-stage supply chain to minimize the makespan. Eur. J. Oper. Res. **244**(1), 13–25 (2015)
16. Pei, J., Liu, X., Pardalos, P.M., Fan, W., Yang, S.: Scheduling deteriorating jobs on a single serial-batching machine with multiple job types and sequence-dependent setup times. Ann. Oper. Res. **249**, 175–195 (2017)
17. Pei, J., Liu, X., Pardalos, P.M., Li, K., Fan, W., Migdalas, A.: Single-machine serial-batching scheduling with a machine availability constraint, position-dependent processing time, and time-dependent set-up time. Optim. Lett. (2016). doi:10.1007/s11590-016-1074-9
18. Xuan, H., Tang, L.X.: Scheduling a hybrid flowshop with batch production at the last stage. Comput. Oper. Res. **34**(9), 2718–2733 (2007)
19. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation indeterministic sequencing and scheduling: a survey. Ann. Discret. Math. **5**, 287–326 (1979)
20. Hansen, P., Mladenović, N.: Variable neighborhood search. Comput. Oper. Res. **24**(11), 1097–1100 (1997)
21. Hansen, P., Mladenović, N., Pérez, J.A.M.: Variable neighbourhood search: methods and applications. 4OR **175**(4), 367–407 (2008)
22. Wang, L., Ni, H., Yang, R., Pardalos, P.M., Du, X., Fei, M.: An adaptive simplified human learning optimization algorithm. Inf. Sci. **320**, 126–139 (2015)
23. Xia, H., Li, X., Gao, L.: A hybrid genetic algorithm with variable neighborhood search for dynamic integrated process planning and scheduling. Comput. Ind. Eng. **102**, 99–112 (2016)
24. Borges, P., Eid, T., Bergseng, E.: Applying simulated annealing using different methods for the neighborhood search in forest planning problems. Eur. J. Oper. Res. **233**(3), 700–710 (2014)
25. Liang, X., Li, W., Zhang, Y., Zhou, M.: An adaptive particle swarm optimization method based on clustering. Soft Comput. **19**(2), 431–448 (2015)