Applied Soft Computing Journal 84 (2019) 105701

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Two-stage hybrid flow shop scheduling on parallel batching machines considering a job-dependent deteriorating effect and non-identical job sizes

Siwen Liu [a], Jun Pei [a,c,*], Hao Cheng [a,*], Xinbao Liu [a,b,*], Panos M. Pardalos [c]

[a] School of Management, Hefei University of Technology, Hefei, Anhui, PR China
[b] Key Laboratory of Process Optimization and Intelligent Decision-making of Ministry of Education, Hefei, Anhui, PR China
[c] Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida, Gainesville, USA

ARTICLE INFO

Article history:
Received 8 October 2018
Received in revised form 28 June 2019
Accepted 2 August 2019
Available online 12 August 2019

Keywords:
Hybrid flow shop scheduling
Parallel batching
Job-dependent deteriorating effect
Heuristic
Hybrid EDA-DE algorithm

ABSTRACT

In this paper, we investigate a specialized two-stage hybrid flow shop scheduling problem with parallel batching machines considering a job-dependent deteriorating effect and non-identical job sizes simultaneously. A novel concept of three-dimensional wasted volume based on the job normal processing time, job size, and job deteriorating rate is first proposed. Some structural properties, as well as a heuristic algorithm, are developed to solve the single parallel batching machine scheduling problem. Since the two-stage hybrid flow shop scheduling problem is NP-hard, a hybrid EDA-DE algorithm combining estimation of distribution algorithm (EDA) and differential evolution (DE) algorithm is proposed to tackle the studied problem. In addition, the Taguchi method of design of experiments (DOE) is implemented to tune the parameters of the EDA-DE. Finally, a series of computational experiments are carried out to compare the performance of the proposed hybrid EDA-DE algorithm and some recent existing algorithms from the literature, and the comparative results validate the effectiveness and efficiency of the proposed algorithm.

1. Introduction

In the semiconductor manufacturing process, wafers with non-identical sizes are put into specific boards and processed in the parallel batching mode in order to enhance productivity. These wafers have to follow the same stage sequence and pass through all stages in the same sequence. Besides, any delay in processing a wafer is penalized and often implies additional time for accomplishing the wafer when maintenance wafers scheduling or assignments cleaning happens [1], and this phenomenon is known as "deteriorating effect" [2]. Since the deteriorating effect causes extra processing time and production cost for processing wafer, it becomes a critical issue to make scientific and reasonable schedule plans to maximize the productivity and utilization of equipment within shorter time in semiconductor manufacturing enterprises.

The production process of the wafer is a typical hybrid flow shop (HFS) on parallel batching machines problem (e.g., see [3–6]). The parallel batching machines can handle several jobs simultaneously as long as the machine capacity is not exceeded (e.g., see [7–9]). This generalization of the flow shop scheduling problem can help produce more products in less production cost and time and has been investigated by many scholars [10–12]. However, as a result of higher demand for fine production in manufacturing enterprises, the influence of the deteriorating effect on production schedule cannot be ignored.

Recent research has considered the scheduling problems under different deteriorating effect functions, and some algorithms were proposed to tackle these scheduling problems [13]. Effective heuristic algorithms can be developed to solve practical scheduling problems [14–16]. Given the time complexity of different scheduling problems, some researchers contribute to developing meta-heuristic algorithms to solve industry-size instances (e.g., see [17–21]).

Estimation of distribution algorithm (EDA), proposed by Mühlenbein and Paass [36], has been treated as a prominent alternative to traditional evolutionary algorithms. Different from regular genetic algorithms, EDA is designed based on the probability theory and it adopts a probabilistic model from selected individuals to produce new solutions at each generation. It guarantees that EDA performs well in different application scenarios by generating new population according to probability theory. For example, Wang et al. [33] designed an EDA to solve the distributed permutation flow shop scheduling problem. In the

* Corresponding authors at: School of Management, Hefei University of Technology, Hefei, Anhui, PR China.
E-mail addresses: feiyijun.ufl@gmail.com (J. Pei), chenghao2008@tom.com (H. Cheng), lxb@hfut.edu.cn (X. Liu).

https://doi.org/10.1016/j.asoc.2019.105701
1568-4946/© 2019 Elsevier B.V. All rights reserved.

Table 1
The comparisons between our previous work, similar work, and the current study.

Publication	DE/LE	Job processing	Algorithm	Features		
				HFS	Job size	Other
Pei et al. [22]	LE	Serial batching	Hybrid VNS-GSA	–	–	Linear setup time
Pei et al. [23]	LE	Serial batching	Hybrid GSA-TS	–	–	Resource-dependent
Pei et al. [24]	DE&LE	Serial batching	Heuristics	–	–	Time-dependent setup time
Pei et al. [25]	DE	Serial batching	Hybrid BA-VNS	–	–	Financial budget Resource constraint Multiple manufacturers
Lu et al. [26]	DE	General processing	Heuristics	–	–	Flow shop manufacturing cell scheduling
Lu et al. [27]	DE	Parallel batching	Hybrid ABC-TS	–	–	Maintenance activity
Liu et al. [28]	DE	Parallel batching	Hybrid VNS-HS	–	–	Supply chain scheduling
Kong et al. [29]	DE	Parallel batching	Hybrid SFLA-VNS	–	–	Non-linear processing time
Kheirandish et al. [30]	–	General processing	ABC	✓	–	Multilevel product structures Requirement operations
Rossi et al. [31]	–	Parallel batching	Heuristics	✓	–	–
Pan and Ruiz [32]	–	General processing	EDA	–	–	Lot-streaming flow shop Stochastic processing times
Wang, Choi, and Qin [33]	–	General processing	EDA	✓	–	Arbitrary release times
Zhou et al. [34]	–	Parallel batching	Modified PSO	–	✓	Due-window assignment
Yang, Yang, and Cheng [35]	DE	General processing	Heuristics	–	–	–
Current study	DE	Parallel batching	Hybrid EDA-DE	✓	✓	

In the above table, DE and LE denote the deteriorating effect and learning effect, respectively.

paper of Pan and Ruiz [32], they proposed a lot-streaming flow shop scheduling problem with setup times and applied EDA to solve their problem. Shen, Wang, and Wang [37] presented an effective bi-population EDA to solve the no-idle permutation flow shop scheduling problem with the total tardiness criterion.

Besides, the differential evolution (DE) is regarded as a traditional efficient evolutionary algorithm and it is well known for its simple and efficient scheme for global optimization over continuous spaces. The traditional DE employs a floating-point encoding method and uses the differentiation information among the population to search the global optimal solution. In the past two decades, DE has been hybridized with other intelligent algorithms and applied to numerous flow shop scheduling problems. For example, Han, Gong, and Sun [38] proposed a discrete Artificial Bee Colony (ABC) algorithm incorporating DE for the flow shop scheduling problem with blocking. Liu, Yin, and Gu [39] combined DE with the individual improving scheme and greedy-based local search to solve a permutation flow shop scheduling problem. Chen et al. [40] also considered a two-stage flow shop scheduling problem on batch processing machines, and they applied a hybrid discrete DE to minimize the makespan.

It is mentioned in Zhang and Li [41] that a local search procedure is effective in improving the solutions generated by the EDA. Different from the previous literature which adopted another neighborhood structures (see, for example, Tzeng, Chen, and Chen [42]), some mutation and crossover operators from DE are designed based on the problem characteristics to enhance the local exploitation around the best solution found by the EDA in this paper. We apply the newly proposed $DE/current-to-pbest$ strategy proposed in Zhang and Sanderson [43]. However, to the best of our knowledge, there is no research combines these two algorithms together to solve the two-stage hybrid flow shop scheduling problem on parallel-batching machines considering deteriorating effect and non-identical job sizes simultaneously.

An extensive study of different models and problems concerning the deteriorating effect and hybrid flow shop can be found in [2,44], and [45]. The parallel batching machines with non-identical job sizes scheduling problem is known to be NP-hard (see, e.g., [46–50]). However, there is a limited attempt in building a bridge between deteriorating effect, batch processing way, and non-identical job sizes. To help find the “right” balance among significant features as well as to optimize the makespan, we propose an effective heuristic algorithm to group the batches by considering above-mentioned features jointly. In the past research, though we have already investigated some similar scheduling with deteriorating effect problems involving scheduling features such as financial budget, manufacturing cell, maintenance activity, and resource constraint (see [22,23,27,28,51]), none of them considered non-identical job sizes and HFS. The deteriorating effect functions we considered in the past research are mostly job-independent and linear functions while the deteriorating effect function under study is a job-dependent non-linear function. Besides, we propose a heuristic algorithm based on the conception of three-dimensional wasted volume, and we apply hybrid EDA-DE to solve the HFS scheduling problem in this paper, which are significantly different from our previous research. The comparisons between our previous work, similar publications, and the current study are listed in Table 1.

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

(1) The coordinated two-stage hybrid flow shop scheduling on parallel batching machines considering the job-dependent deteriorating effect and non-identical job sizes is first investigated in this paper.

(2) For the single parallel batching problem, we first propose a three-dimensional wasted volume based on the job normal processing time, job size, and deteriorating rate. For this case, a heuristic algorithm is derived to group all the jobs into batches in order to minimize the makespan.

(3) Based on the derived structural properties and heuristic algorithm for the single parallel batching machine problem, we develop an effective hybrid EDA-DE which combines EDA and DE to solve the two-stage hybrid flow shop problem.

The remainder of this paper is organized as follows: Section 2 gives the description of the problem and list of notations. Some preliminary analysis is presented in Section 3 based on which a heuristic algorithm is proposed to tackle the single machine case. Section 4 describes the hybrid EDA-DE algorithm in details and illustrates the procedures. Computational results and comparisons are reported and discussed in Section 5. Finally, a brief conclusion is summarized in Section 6.

2. Notations and problem statement

The following notations listed in Table 2 are adopted to describe the problem.

A set $J = \{1, 2, \ldots, n\}$ of n non-identical jobs is processed on a two-stage hybrid flow shop production setting. There are

Table 2
The list of the notations.

Notations	Definition
n	Number of jobs
m_l	Number of machines at stage l, $l = 1, 2$
j	Index for jobs, $j = 1, 2, \ldots, n$
i	Index for machines, $i = 1, 2, \ldots, m_l$
l	Index for stages, $l = 1, 2$
b	Index for batches, $b = 1, 2, \ldots, n$
n_i	Number of jobs on machine i at each stage
r	Job position on a machine, $r = 1, 2, \ldots, n_i$
a_j	Job-dependent deteriorating effect of job J_j, $j = 1, 2, \ldots, n$
a_{B_b}	Deteriorating effect of batch B_b, $b = 1, 2, \ldots, n$
p_{jl}	Normal processing time of job J_j, $j = 1, 2, \ldots, n$ at stage l, $l = 1, 2$
p_{jlr}	Actual processing time of job J_j at stage l, position r, $j = 1, 2, \ldots, n$; $l = 1, 2$; $r = 1, 2, \ldots, n_i$
s_j	Size of job J_j, $j = 1, 2, \ldots, n$
C	Machine capacity

m_l identical parallel machines included at stage l. At least one of the two stages includes more than one machine, and both stages include identical parallel batching machines which can process multiple jobs simultaneously as long as the total size of a batch is no more than the machine capacity [27]. In this paper, we consider a job-dependent deteriorating effect in our scheduling model. Job J_j has a normal processing time p_{jl} and a job-dependent deterioration factor a_j, where $a_j > 0$. If job J_j is scheduled in the position r at stage l, then its actual processing time can be denoted as [35]

$$p_{jlr} = p_{jl} r^{a_j} \tag{2.1}$$

Considering the characteristic of parallel batching machine, once a batch is determined, the deteriorating effect rate of this batch B_b is defined as the largest deteriorating effect rate of the jobs belonging to this batch which means $a_{B_b} = \max_{J_j \in B_b} \{a_j\}$. Consistent with the measurement of system performance for classical flow shop scheduling problems, the objective of this paper is to minimize the makespan C_{max} of all the jobs. The following assumptions are made:

- Each job J_j can be processed by at most one machine for each stage.
- Once a batch is formed, jobs cannot be added to or removed from it.
- Batch reformation is not allowed on the second stage.
- All the machines are continuously available during the planning horizon.
- There is unlimited buffer capacity between the stages.

The problem formulation can be described using a triplet $\alpha|\beta|\gamma$ notation, which is derived from [52]. Applying this notation, the scheduling problem considered in this paper is denoted as follows.

$$FP2B(m_1, m_2)|p-batch, s_j, p_{jlr} = p_{jl} r^{a_j}|C_{max}$$

According to Uzsoy [53], the problem $P_m \parallel C_{max}$ is NP-hard. If there is only one stage in our problem, the problem $P_m \parallel C_{max}$ can be reduced to our problem in polynomial time. It is easily to derive that our problem $FP2B(m_1, m_2)|p-batch, s_j, p_{jlr} = p_{jl} r^{a_j}|C_{max}$ is also NP-hard.

In Fig. 1, we illustrate an example of our problem using a Gantt chart. As shown below, each job J_j with job size s_j has to be assigned to a machine at the first stage and then put into a batch. After the batches are completed at the first stage, these formed batches are then assigned to machines at the second stage. Both Stage 1 and Stage 2 contain two parallel batching machines M_{11}, M_{12} and M_{21}, M_{22}. Firstly, at Stage 1, $\{J_1, J_2, J_3, J_4\}$ are assigned to machine M_{11} while $\{J_5, J_6, J_7, J_8\}$ are assigned to machine M_{12}. Then, these jobs are grouped into four batches: $B_1 = \{J_1, J_2\}$, $B_2 = \{J_3, J_4\}$, $B_3 = \{J_5, J_6\}$, and $B_4 = \{J_7, J_8\}$. After processing at stage 1, B_1 and B_4 are then assigned to machine M_{21} at the second stage while B_2 and B_3 are then assigned to machine M_{22} at the second stage. The value of C_{max} is equal to the completion time of batch B_4 at the second stage.

3. Preliminary analysis

Considering the time complexity of our proposed problem is NP-hard, in this section, we first analyze the single machine circumstance. We find that the wasted volume (WV) in a batch directly affects the makespan of the schedule, and based on this, a heuristic algorithm is developed to group the batches. Besides, we list three classical heuristics for flow shop sequencing problems. The definition of the wasted volume and its relationship to the makespan is given below.

3.1. Batch formation heuristic

Based on the characteristics of the studied scheduling problem, each job has three attributions during the production process: basic processing time, size, and job-dependent deteriorating effect rate. We connect them with our objective from a three-dimensional view and propose a wasted volume method for the single machine circumstance.

Definition 1. Let P_x and a_{B_x} denote the largest normal processing time and largest deteriorating effect rate among a batch B_x. Then the wasted volume of the batch B_x can be calculated by the following formula (3.1):

$$WV_x = C \cdot P_x \cdot x^{a_{B_x}} - \sum_{J_j \in B_x} p_j \cdot s_j \cdot x^{a_j} \tag{3.1}$$

Fig. 2 illustrates an example of an arbitrary batch B_x in a feasible schedule containing two jobs J_1 and J_2, and the position of the batch B_x is x. The parameter WV_x is used to denote the wasted volume of the batch. In the three-dimensional reference system shown in Fig. 2(a), the coordinate axes represent the job-dependent deteriorating rate, the normal processing time P, and job size s. In Fig. 2(b), the first part of the wasted volume WV_1 is shown by a solid blue cube, and in Fig. 2(c) the second part of the wasted volume WV_2 is shown by another solid blue cube. We can easily see that $WV_x = WV_1 + WV_2$.

Lemma 1. *For the problem $1|p-batch, p_{jr} = p_j r^{a_j}, s_j|C_{max}$, let batch B_x be the current batch and P_x^A be the actual processing time of the current batch, then minimizing the wasted volume is equal to minimizing the actual processing time of the current batch.*

Proof. Based on the above formula (3.1), the total wasted volume of this batch can be denoted as follows:

$$WV_x = C \cdot P_x \cdot x^{a_{B_x}} - \sum_{J_j \in B_x} p_j \cdot s_j \cdot x^{a_j} = C \cdot P_x^A - \sum_{J_j \in B_x} p_j \cdot s_j \cdot x^{a_j}$$

Once the jobs assigned to each batch are determined, $\sum_{J_j \in B_x} p_j \cdot s_j \cdot x^{a_j}$ will be a constant. The above formula provides an approach to express the makespan using the total wasted volume. Observe that if the value of WV decreased, the value of P_x^A will be decreased accordingly. Therefore, Lemma 1 is proved. ■

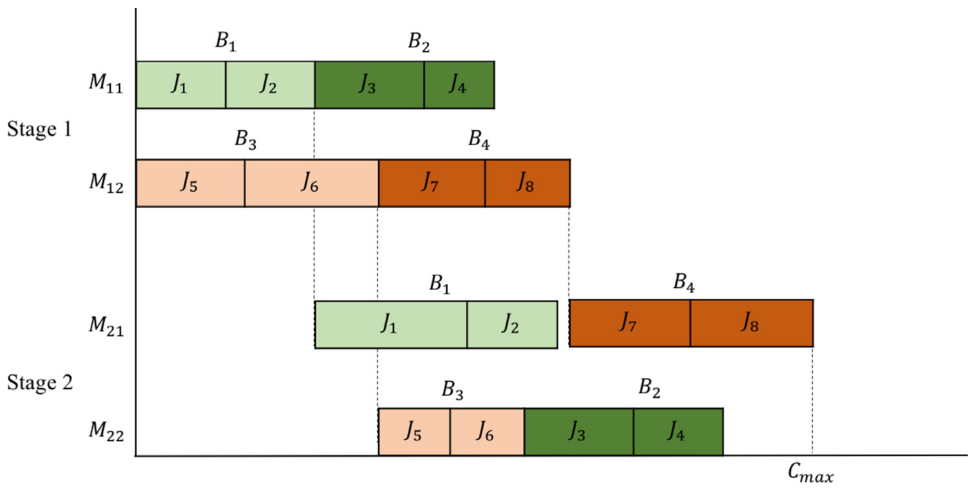
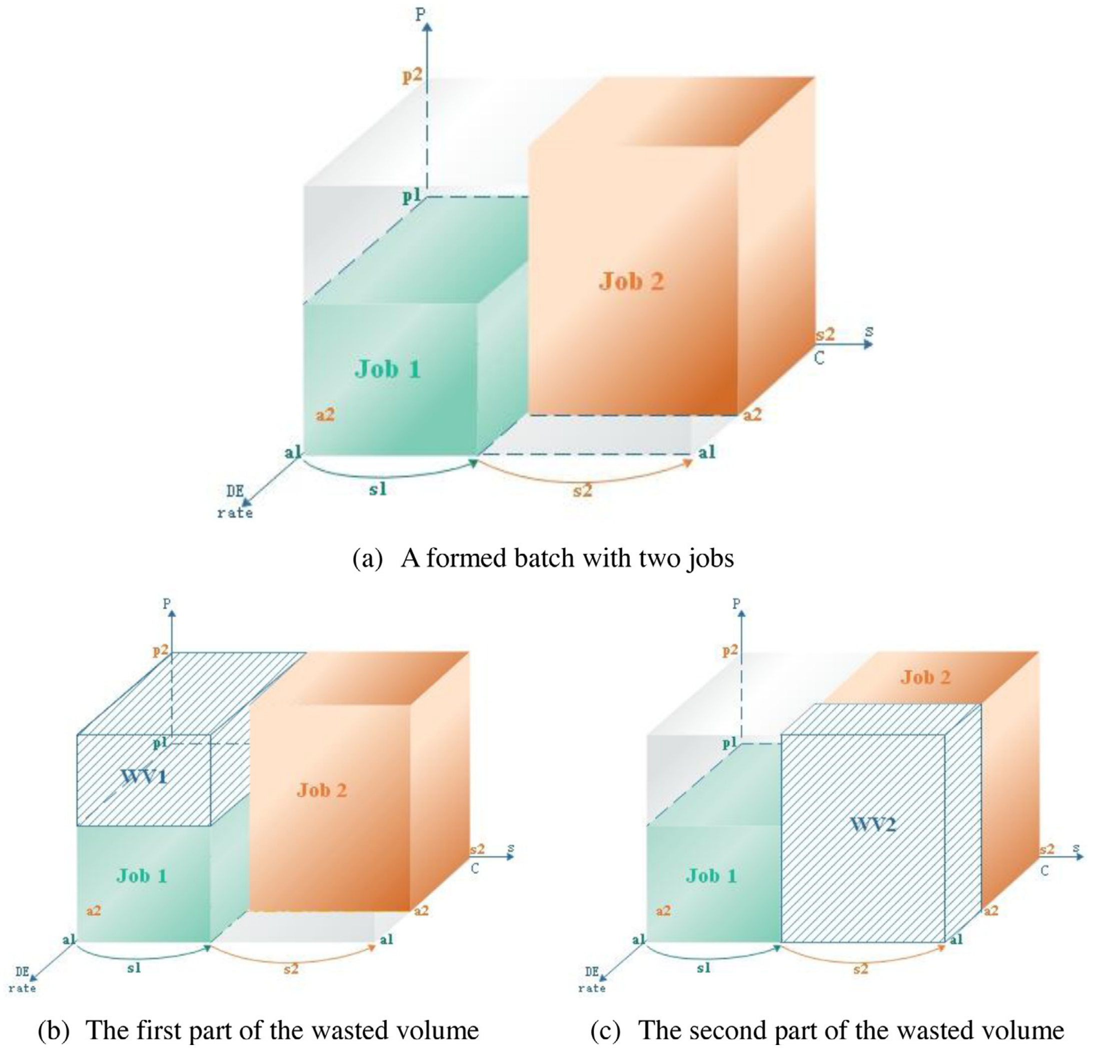


Fig. 1. Gantt chart of an example of our problem.



P
p2
p1
Job 1
Job 2
a1
a2
s1
s2
S
C
DE rate
WV1
WV2
(a) A formed batch with two jobs
(b) The first part of the wasted volume
(c) The second part of the wasted volume

Fig. 2. An example of the wasted volume.

Lemma 2. *Suppose batch B_x is the current batch, and the job set containing the unassigned jobs is U_x. If there exists a job J_y from U_x which can satisfy the following constraints:*

$$s_y \le C - s_{B_x} \tag{3.2}$$

$$C \cdot (max\{P_x, p_y\} \cdot x^{\max\{a_{B_x}, a_y\}} - P_x \cdot x^{a_x}) \ge s_y \cdot p_y \cdot x^{a_y} \tag{3.3}$$

then it will decrease the completion time of the current batch by adding this job J_y.

Proof. We apply the abovementioned concept of wasted volume to prove this lemma. Suppose job J_y is added into the batch B_x, and after that, the wasted volume of this batch B_x is:

$$WV'_x = C \cdot max\{P_x, p_y\} \cdot x^{max\{a_{B_x}, a_y\}} - \sum_{J_j \in B'_x} p_j \cdot s_j \cdot x^{a_j} \tag{3.4}$$

Combining the formulas (3.1) and (3.4), the difference between WV'_x and WV_x is calculated below:

$$\Delta WV_x = C \cdot max\left\{P_x, p_y\right\} \cdot x^{\max\left\{a_{B_x}, a_y\right\}} - C \cdot P_x \cdot x^{a_{B_x}} - p_y \cdot s_y \cdot x^{a_y}$$

Case 1. $p_y \geq P_x$, $a_y \geq a_{B_x}$

$$\begin{aligned}\Delta WV_x &= C \cdot p_y \cdot x^{a_y} - C \cdot P_x \cdot x^{a_{B_x}} - s_y \cdot p_y \cdot x^{a_y} \\ &= C \cdot (p_y \cdot x^{a_y} - P_x \cdot x^{a_{B_x}}) - s_y \cdot p_y \cdot x^{a_y}\end{aligned}$$

Case 2. $p_y \geq P_x$, $a_y < a_{B_x}$

$$\begin{aligned}\Delta WV_x &= C \cdot p_y \cdot x^{a_{B_x}} - C \cdot P_x \cdot x^{a_x} - s_y \cdot p_y \cdot x^{a_y} \\ &= C \cdot (p_y \cdot x^{a_x} - P_x \cdot x^{a_x}) - s_y \cdot p_y \cdot x^{a_y}\end{aligned}$$

Case 3. $p_y < P_x$, $a_y \geq a_{B_x}$

$$\begin{aligned}\Delta WV_x &= C \cdot P_x \cdot x^{a_y} - C \cdot P_x \cdot x^{a_{B_x}} - s_y \cdot p_y \cdot x^{a_y} \\ &= C(P_x \cdot x^{a_y} - P_x \cdot x^{a_{B_x}}) - s_y \cdot p_y \cdot x^{a_y}\end{aligned}$$

Case 4. $p_y < P_x$, $a_y < a_{B_x}$

$$\Delta WV_x = C \cdot P_x \cdot x^{a_{B_x}} - C \cdot P_x \cdot x^{a_{B_x}} - p_y \cdot s_y \cdot x^{a_y} = -p_y \cdot s_y \cdot x^{a_y}$$

In this case, the value of ΔWV_x is always less than zero, so we omit it. Combining the other three cases, the above Lemma 2 is proved. ■

Lemma 3. *Let $f(x) = (r+1)^x - r^x$, $r = 1, 2, \ldots, n-1$, $x > 0$. Then, $f(x)$ is a non-decreasing function.*

The proof is simple, and we omit it.

Lemma 4. *For the problem $1|p-batch, p_{jr} = p_j r^{a_j}, s_j|C_{max}$, if there are two consecutive batches B_u and B_v with their deteriorating rate $a_{B_u} \geq a_{B_v}$ and their normal processing time $P_u \geq P_v$ in an optimal schedule, then the batch B_u should be sequenced in an earlier position.*

Proof. Suppose that there are two consecutive formed batches B_u and B_v, their normal processing times and deteriorating effect rates are P_u, P_v, a_u, and a_v, respectively. The processing order in an optimal schedule π is (W, B_u, B_v), the current position of batch B_u is r. Swap the batches B_u and B_v, the sequence turns into (W, B_v, B_u), and the new schedule is denoted as π'. The completion time of batch B_v in schedule π is

$$C(B_v(\pi)) = t_0 + P_u \cdot r^{a_{B_u}} + P_v \cdot (r+1)^{a_{B_v}}$$

Then the completion time of batch B_u in schedule π' is

$$C\left(B_u\left(\pi'\right)\right) = t_0 + P_v \cdot r^{a_{B_v}} + P_u \cdot (r+1)^{a_{B_u}}$$

Therefore,

$$\begin{aligned}C\left(B_u\left(\pi'\right)\right) - C(B_v(\pi)) &= P_v \cdot r^{a_{B_v}} + P_u \cdot (r+1)^{a_{B_u}} \\ &\quad - \left(P_u \cdot r^{a_{B_u}} + P_v \cdot (r+1)^{a_{B_v}}\right) \\ &= P_u \cdot \left((r+1)^{a_{B_u}} - r^{a_{B_u}}\right) \\ &\quad - P_v \cdot ((r+1)^{a_{B_v}} - r^{a_{B_v}})\end{aligned}$$

If $a_u \geq a_v$, then $(r+1)^{a_{B_u}} - r^{a_{B_u}} \geq (r+1)^{a_{B_v}} - r^{a_{B_v}}$. To ensure the priority of the original schedule, the processing times of these two batches need to satisfy that $P_u \geq P_v$. Thus, the proof is completed. ■

Based on the above Lemmas 1, 2, and 4, we develop an effective heuristic algorithm in the following to group the batches when the job sequence is determined.

Heuristic 1

Step 1.	Choose the first job J_j from a given sequence and put it into the first batch B_1. Calculate the completion time of the current batch.
Step 2.	Select job J_j which satisfies the constraint (3.2) and (3.3) from the unassigned job set. If the batch size does not exceed the machine capacity after adding job J_j, put the job J_j into the current batch.
Step 3.	Repeat Step 2 until the current batch cannot hold any job. Then turn to the next batch.
Step 4.	Repeat above Step 2 and Step 3 until all the batches are formed.
Step 5.	Choose the first formed batch n_1, if $a_1 \leq a_2$ and $P_1 < P_2$ exists, then swap batch B_1 and B_2.
Step 6.	Turn to next formed batch until all the formed batches are examined.

3.2. Job sequence heuristics

In this subsection, three classical and typical heuristic algorithms in scheduling problems are introduced to give a reasonable job sequence. These heuristics include Johnson's rule [54], LNPT (Longest normal processing time), and SNPT (Shortest normal processing time). The basic procedures of abovementioned heuristics are as follows [11]:

1. Johnson's rule: A sequence generated by Johnson's rule can be stated as follows: First, partition the jobs into two sets with Set I containing all jobs with $p_{1j} < p_{2j}$ and Set II containing all jobs with $p_{1j} > p_{2j}$. The jobs with $p_{1j} = p_{2j}$ can be put into either set. The jobs in Set I go first and are sequenced in the increasing order of p_{1j} (SPT) while the jobs in Set II are sequenced in the decreasing order of p_{2j} (LPT).
2. LNPT: In the LNPT heuristic, the jobs are sequenced based on their sum of normal processing time of each job on all machines in the descending order.
3. SNPT: In the SNPT heuristic, the jobs are sequenced based on their sum of normal processing time of each job on all machines in the ascending order.

In the computational experiments section, we will design a series of preliminary experiments to test the performance of the above three job sequence heuristics combined with our proposed Heuristic 1.

3.3. Lower bound

To verify the effectiveness of the proposed hybrid EDA-DE in the next section, a lower bound is proposed for the problem under study. Inspired by Ozturk et al. [55], we develop a lower bound algorithm based on the job-split method. Considering the jobs with different size may not fill a batch, the job-split method allows a job to be split into smaller size job with corresponding normal processing time. The remaining part of this job will be accommodated into the next batch.

The procedure to calculate the lower bound starts by sorting the jobs in the non-decreasing order of their normal processing time and form the batches according to the Full-batch-shortest-processing-time (FBSPT). The relaxed number of batches is denoted as $n_b = \left\lceil \frac{\sum_{j=1}^{n} s_j}{C} \right\rceil$. In order to relax the deteriorating effect, we assume that all the jobs have the equal deteriorating effect ratio $a = \min_{j\in J} a_j$.

After the jobs are all grouped into batches, they will be assigned to different machines at the first stage consecutively. Considering that the batch-reformation is not allowed in this paper, once the batches finish processing at the first stage, the batch completion can be seen as the release dates for the second stage. The detailed procedure is presented in the following LB algorithm.

Two-stage LB algorithm	
Step 1.	Sort jobs in non-decreasing order of job normal processing time p_j such that $L_1 : p_{11} \le p_{21} \le \cdots \le p_{n1}$.
Step 2.	Calculate the minimum number of batches needed: $n_b = \left\lceil \frac{\sum_{j=1}^{n} s_j}{C} \right\rceil$
Step 3.	While $n_b > 0$, open a new batch B_b
Step 3.1.	Put the first job of L_1 into the batch B_b. Update L_1. If there is no space or no elements in L_1, close the batch and set $n_b = n_b - 1$.
Step 3.1.1.	If the job does not completely fit the batch, split the job. Put the first part of the job to fill the batch entirely, then close the batch and set $n_b = n_b - 1$. Update the size of the split job.
Step 4.	Sort batches in non-decreasing order of batch completion time P^A_{1b} such that $L_2 : P^A_{1b} \le P^A_{2b} \le \cdots \le P^A_{1n_b}$, then arrange the batches at the second stage accordingly.

4. Hybrid EDA-DE algorithm

To solve the problem under study, we employ a meta-heuristic algorithm to solve industry-size problems. In our proposed flow shop scheduling model, jobs need to be assigned to different batches, and batches need to be assigned to different machines at each stage. Considering this situation, a population-based meta-heuristic algorithm called estimation of distribution algorithm (EDA) is applied for its remarkable performance in traditional flow shop scheduling problems (see, e.g., [56–58]). Instead of using the conventional crossover and mutation operations of genetic algorithms, EDA produces new offspring in an implicit way based on a probabilistic model. The general steps of EDA are explained in the following:

(1) Initialization: randomly generate an initial population.
(2) Probabilistic model construction: the individuals used to generate the probabilistic model are chosen from the best individuals.
(3) Offspring production: generate new offspring according to the probabilistic model.
(4) Local search: conduct the local search on offspring individuals to enhance the local exploitation.
(5) Replacement: replace part/all of the individuals in the current population with the new offspring.
(6) Termination check: repeat from step 2 until the termination criterion is met.

Moreover, in order to improve the overall performance of EDA and prevent local optimality, we hybridize it with another meta-heuristic algorithm: differential evolution (DE). DE is one of the most powerful stochastic optimization algorithms. It starts with a population of randomly initialized individuals. Then, a mutation operator and a crossover operator are used respectively to produce new candidates at each generation. A selection operator is applied to decide whether the offspring or the parent keeps in the next population.

To accelerate the search procedure and improve the solution quality of our proposed algorithm, we combine these two classical algorithms EDA and DE together to solve our proposed flow shop problem.

In this section, the solution representation and population initialization are explained first, then the details of selection operator, probabilistic model, generation of new populations, and procedure of local search are presented. The framework of the proposed hybrid EDA-DE algorithm is illustrated in the last.

4.1. Solution representation and population initialization

For the hybrid flow shop problem, the widely applied encoding method is job permutation-based scheme, in which each integer denotes a job number. Based on our previous analysis, we adopt a machine-based encoding method in which a sequence of number is given to determine the machine assignment for all the jobs on the first stage. The following example in Fig. 3 illustrates the encoding procedure for an eight-job, three-machine on first stage instance with a given permutation. To transform the given permutation, we order the decimals in descending order. The largest number 9 and the second largest number 8 are regarded as flags to separate these 8 jobs. As a result of the separation, $\{J_1, J_2\}$ are assigned to machine 1, $\{J_3, J_4, J_5, J_6\}$ are assigned to machine 2 while $\{J_7, J_8\}$ are assigned to machine 3. In order to guarantee the diversity of the initial population, all the individuals are randomly generated which is also applied by Jarboui, Eddaly, and Siarry [58].

4.2. Selection operator and probability model

In some classical evolutionary meta-heuristic algorithms like GA, selection operators are always implemented through roulette and tournament. In this paper, the best *SP_Size* individuals are selected from the current population to generate the probabilistic model. Given the performance of the EDA is largely influenced by the probabilistic model, the best choice of the model is crucial for designing an effective EDA. Obviously, a good model cannot only enhance the algorithm's efficiency but also improve the effectiveness of optimization problems. In previous flow shop scheduling literature which also applied EDA, for example, Jarboui et al. [58] presented a probabilistic model based on both the order of the jobs in the sequence and similar blocks of jobs exist in the selected individuals. Different from the traditional probabilistic model establishment, in this paper, we adopt a novel probabilistic model construction strategy considering our encoding method which is described as follows:

Let element $p_{ij}(l)$ of the probability matrix P represent the probability of flag appears in position i in the selected individual j at generation l.

$$p_{ij}(l) = \begin{cases} \dfrac{1}{m_1 - 1} & \textit{if the current position } i \textit{ appears flag} \\ 0 & \textit{else} \end{cases}$$

At each generation of the EDA, the new individuals are generated through sampling the solution space according to the probability matrix P. After traverse the chosen *SP_Size* individuals, each position i will have a probability p_i. Then, new individuals are generated according to the probability matrix P. We present the pseudo code of the constructing procedure in the following part.

In the procedure, a random number function which returns a decimal between 0 and 1 is used to decide the value for the current position. To generate the next offspring, the above procedure will repeat *pop_size* times to generate offspring from the probabilistic model.

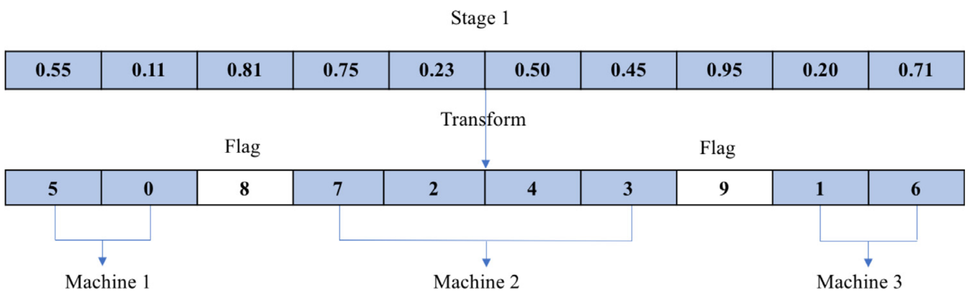


Fig. 3. An example of the encoding method.

```
Begin
For $i = 1$ to $n$ do
    If $rand() > p_i$ then
        Set the current position as a flag
    Else
        Set the current position as $rand()$
    End if
End for
End
```

In order to maintain the diversity of the population and avoid local optimum, the updating of the EDA probability matrix is required. Based on our previous analysis, the following Eq. (4.1) is used to generate the new probability matrix:

$$p_{ij}(l+1) = (1-\alpha)\,p_{ij}(l) + \frac{\alpha}{i \times SP\_Size}\sum_{k=1}^{SP\_Size} I_i^k \tag{4.1}$$

where $\alpha \in (0, 1)$ is the learning rate of P, and I_i^k is the following indicator function of the kth individual in the superior sub-population. The second term on the right of the equation can obtain learning information from the superior sub-population, so the above updating procedure is regarded as a type of increased learning.

$$I_i^k = \begin{cases} 1 & \text{if flag appears before or in position } i \\ 0 & \text{else} \end{cases} \tag{4.2}$$

4.3. Mutation and crossover operations

It is mentioned in Zhang and Li [41] that a local search procedure is effective in improving the solutions generated by the EDA. Different from the previous literature which adopted another neighborhood structures (see, for example, Tzeng, Chen, and Chen [42]), some mutation and crossover operators from DE are designed based on the problem characteristics to enhance the local exploitation around the best solution found by the EDA in this paper. We apply the newly proposed $DE/current-to-pbest$ strategy proposed in Zhang and Sanderson [43] where a new individual is generated by Eq. (4.3)

$$v_i^t = x_i^t + F\cdot(x_p^{best,t} - x_i^t) + F\cdot(x_{r_1}^t - x_{r_2}^t) \tag{4.3}$$

where $x_p^{best,t}$ is uniformly selected as one of the top 100p% individuals in the current population with $p \in (0, 1]$, and $x_{r_1}^t$, $x_{r_2}^t$ are randomly chosen from the current population.

The crossover is applied to increase the diversity of the perturbed parameter vectors. To this end, the trial vector:

$$u_i^{t+1} = (u_{1i}^{t+1}, u_{2i}^{t+1}, \ldots, u_{hi}^{t+1}) \tag{4.4}$$

is formed, where

$$u_{ji}^{t+1} = \begin{cases} v_{ji}^{t+1} & if(randb(j) \le CR) \\ x_{ji}^{t+1} & if(randb(j) > CR) \end{cases} \tag{4.5}$$

Table 3
Combinations of different parameter values.

Parameters	Factor level			
	1	2	3	4
P_Size	40	80	120	160
SP_Size	0.1	0.2	0.3	0.4
α	0.1	0.2	0.3	0.4
F	0.2	0.4	0.6	0.8
CR	0.3	0.5	0.7	0.9

Table 4
Orthogonal array and ARV values.

Experiment number	Factor					ARV
	P_Size	*SP_Size*	α	*F*	*CR*	
1	1	1	1	1	1	30.780
2	1	2	2	2	2	43.662
3	1	3	3	3	3	67.571
4	1	4	4	4	4	44.044
5	2	1	2	3	4	41.560
6	2	2	1	4	3	41.869
7	2	3	4	1	2	37.168
8	2	4	3	2	1	42.825
9	3	1	3	4	2	58.614
10	3	2	4	3	1	27.779
11	3	3	1	2	4	46.143
12	3	4	2	1	3	30.067
13	4	1	4	2	3	40.473
14	4	2	3	1	4	29.045
15	4	3	2	4	1	31.106
16	4	4	1	3	2	26.484

4.4. Framework of the proposed EDA-DE

In Fig. 4, the framework of our proposed hybrid EDA-DE algorithm is shown to solve the problem $FP2B(m_1, m_2)|p-batch, s_j, p_{jr} = p_j r^{a_j}|C_{max}$.

5. Computational results and comparisons

To test the performance of the proposed hybrid EDA-DE under different problem scales, a series of computational experiments are carried out in this section. Given the particularity of the proposed two-stage hybrid flow shop scheduling model, there is no available benchmark for this type of problem, so we adopt datasets generated based on the production scenarios. To evaluate the performance of the proposed EDA-DE algorithm from different views, first, the EDA-DE algorithm is integrated with different job sequence heuristic methods to choose a relative better combination. Then, the EDA-DE algorithm is compared with DE (Storn and Price [59]) and EDA (Larrañaga and Lozano [60]). Also, another three meta-heuristic algorithms: PSO (Liu, Wang, and Jin [61]), VNS (Li, Pan, and Wang [62]), and CS (Yang and Deb [63]) are realized to more visually observe the superior aspects of the EDA-DE. Note that all the algorithms are coded in Java, and run on

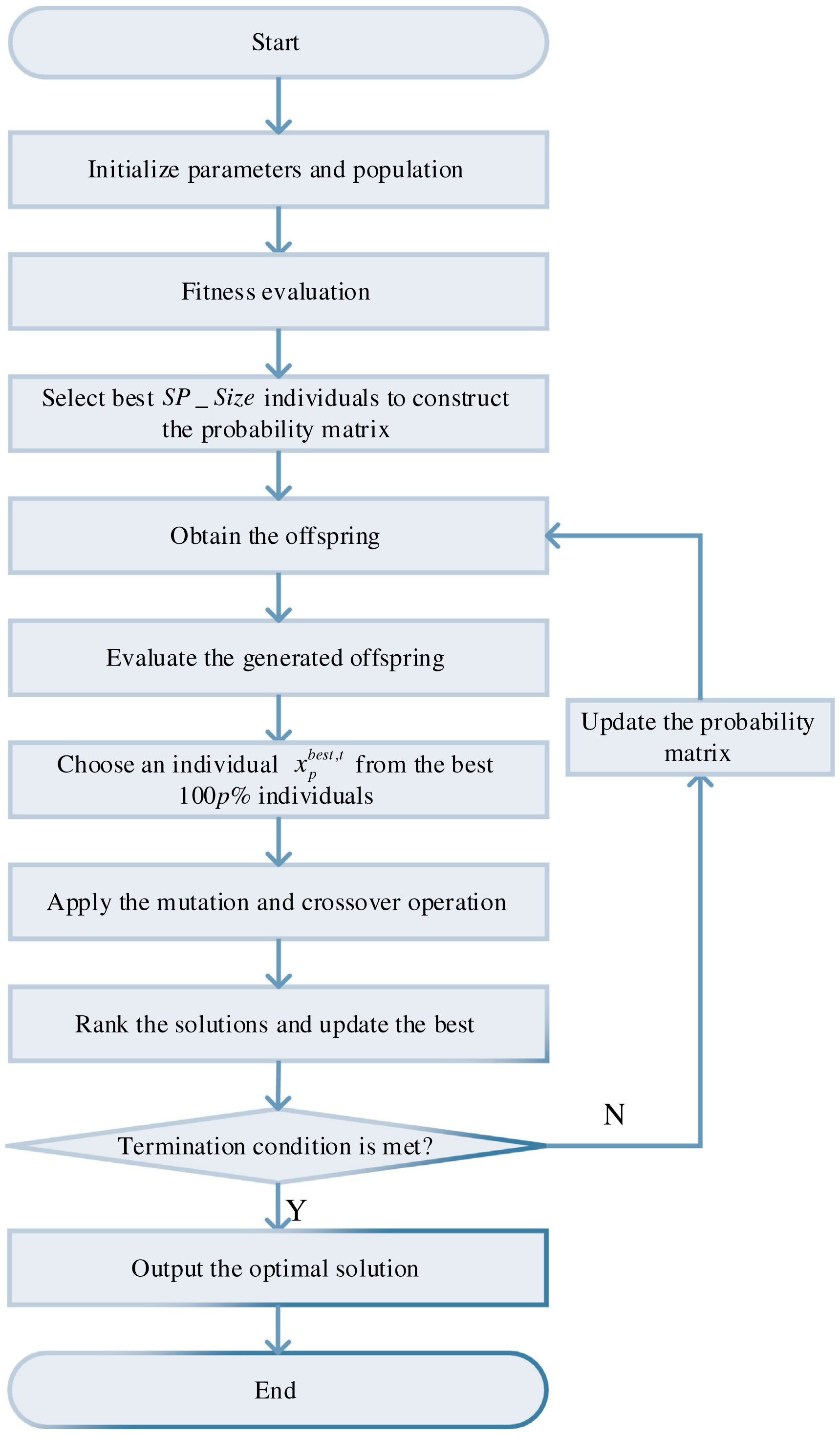

Start
Initialize parameters and population
Fitness evaluation
Select best SP_Size individuals to construct the probability matrix
Obtain the offspring
Evaluate the generated offspring
Choose an individual $x_p^{best,t}$ from the best $100p\%$ individuals
Apply the mutation and crossover operation
Rank the solutions and update the best
Termination condition is met?
N
Y
Update the probability matrix
Output the optimal solution
End

Fig. 4. The flowchart of the proposed EDA-DE.

Intel(R) Core(TM) i5-6200U CPU @2.30 GHz under Windows 10 environment.

Relative Percentage Deviation (RPD) values presented in Naderi and Ruiz [64] will be applied to evaluate the performance of the EDA-DE. RPD is calculated by following Eq. (5.1):

$$\mathrm{RPD} = \frac{alg - opt}{opt} \times 100 \tag{5.1}$$

where *alg* and *opt* are the method solution and best-known solution in a single instance, respectively. A smaller RPD value indicates that the current algorithm has closer results compared to the optimal solutions.

5.1. Experimental design

The proposed EDA-DE contains several key parameters: *P_Size* (the population size), *SP_Size* (the chosen population size), α (the learning rate for probability matrix P), F (the mutation factor), and *CR* (the crossover factor). To investigate the influence of these parameters on the performance of the EDA-DE, we implement the

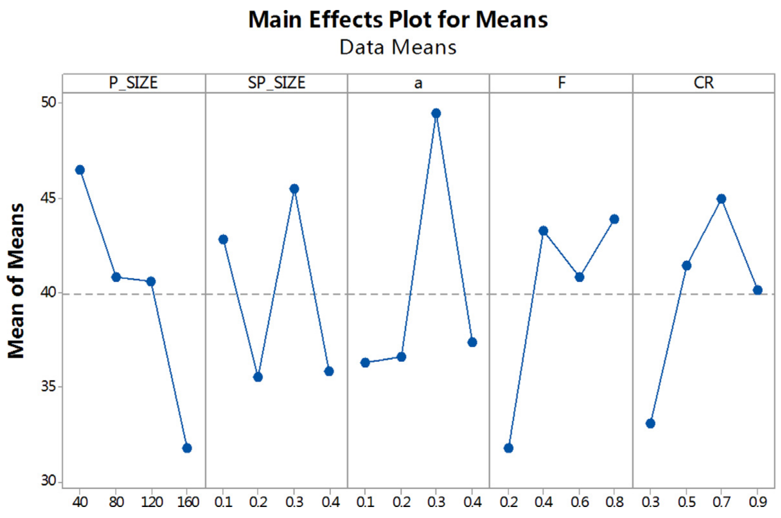


Fig. 5. Factor level trend of the EDA-DE.

Table 5
Response value and rank of each parameter.

Level	*P_Size*	*SP_Size*	α	*F*	*CR*
1	46.51	42.86	36.32	31.77	33.12
2	40.86	35.59	36.60	43.25	41.48
3	40.65	45.50	49.51	40.85	45.00
4	31.78	35.86	37.37	43.91	40.20
Delta	14.74	9.91	13.19	12.14	11.87
Rank	1	5	2	3	4

Taguchi method of design of experiment (DOE) by using a small-sized instance, i.e., job number is 30, machine number at each stage is 3 and 2. Based on the previous literature about EDA and DE, the factors and their values are listed in Table 3. According to the number of parameters and factor level listed, we choose the orthogonal array $L_{16}(4^4)$ to implement the DOE. Each instance will run 10 times independently and the average values calculated by EDA-DE will be taken as the final ARV (average response variable). Different combinations of these factors and the corresponding ARV values are listed in Table 4.

According to the orthogonal table presented in Table 4, we illustrate the trend of each factor level in Fig. 5. Then, we figure out the response value of each parameter to analyze its significance rank. The results are listed in Table 5.

From Table 5, it can be seen that the *P_Size* is the most significant one among the five parameters, that is, the *P_Size* of the EDA-DE is crucial. A large value of *P_Size* makes the algorithm sample the solution space more entirely. Besides, the learning rate α ranks second, which implies that the number of superior sub-population to update the probability model is also important. A large value of α could lead to premature convergence. In addition, the significant rank of the value of *F* is the third. However, it can be seen from Fig. 5 that it makes no improvement when the value of *CR* is too large. According to the above analysis, an appropriate parameter combination is suggested to be $P\_Size = 160$, $SP\_Size = 0.4$, $\alpha = 0.1$, $F = 0.6$, and $CR = 0.5$.

5.2. Comparison of Johnson, LNPT, SNPT

In order to select a heuristic algorithm which can combine the EDA-DE best to sequence all the jobs at the first stage, we design comparison experiments between Johnson, LNPT, and SNPT using small-scaled instances. From three different aspects: problem size, number of jobs, and number of machines, each instance is run 20 times, and the average makespan, best value, and RPD are recorded. Each job processing time is between [1,2], and the job size is between [1,2]. The experiment results from the above mentioned three aspects are shown in Tables 6, 7, and 8, respectively. As it can be seen, best average values for AVG, Best,

Table 6
Comparison between different heuristic based on problem size.

Problem size	Johnson			LNPT			SNPT		
	AVG	Best	RPD	AVG	Best	RPD	AVG	Best	RPD
30	**12.971**	12.795	**0.014**	17.214	14.819	0.162	13.350	**12.582**	0.061
40	**11.672**	**10.241**	0.140	14.401	12.245	0.176	12.007	10.970	**0.095**
50	18.601	17.772	0.047	18.832	16.619	0.133	**15.441**	**14.702**	**0.050**
60	**10.344**	**9.535**	0.085	13.790	13.134	0.050	10.738	10.442	**0.028**
AVG	13.397	12.586	0.072	16.059	14.204	0.130	**12.884**	**12.174**	**0.059**

Table 7
Comparison between different heuristic based on the number of jobs.

Number of jobs	Johnson			LNPT			SNPT		
	AVG	Best	RPD	AVG	Best	RPD	AVG	Best	RPD
20	11.447	11.228	0.020	10.946	**8.793**	0.245	**8.898**	**8.793**	**0.012**
30	19.728	19.647	**0.004**	18.615	15.104	0.232	**15.157**	**12.749**	0.189
40	23.459	21.508	**0.091**	25.784	20.960	0.230	**22.686**	**19.817**	0.145
50	**20.951**	**19.202**	**0.091**	29.450	25.192	0.169	24.303	21.669	0.122
AVG	18.896	17.896	**0.052**	21.199	17.512	0.219	**17.761**	**15.757**	0.117

Table 8
Comparison between different heuristic based on the number of machines.

Number of machines	Johnson			LNPT			SNPT		
	AVG	Best	RPD	AVG	Best	RPD	AVG	Best	RPD
(2×2)	15.429	15.429	**0.000**	19.757	16.228	0.217	**14.862**	**12.126**	0.226
(2×3)	14.925	14.925	**0.000**	15.190	13.776	0.103	**12.312**	**10.981**	0.121
(3×2)	11.355	9.760	**0.163**	15.590	12.831	0.215	**11.257**	**8.557**	0.316
(3×3)	**11.070**	**9.836**	0.125	14.133	13.315	0.061	12.320	11.307	**0.090**
AVG	13.195	12.488	**0.072**	16.168	14.038	0.149	**12.688**	**10.742**	0.188

Table 9
Parameters setting.

Parameters	Description	Value
n	Number of total jobs	30, 40, 50, 60, 70, 80, 90, 100
m_l	Number of machines at stage l	2, 3
p_{jl}	Normal processing time of job J_j at stage l	$U[1, 10]$
a_j	Job-dependent deteriorating effect rate of job J_j	$U[0, 1]$
s_j	Job size of job J_j	1, 2, 3
C	Machine capacity	6

Table 10
Comparative results for the six algorithms.

Problem	EDA-DE			EDA			DE			CS			PSO			VNS		
	Best	Avr	RPD	Best	Avr	RPD	Best	Avr	RPD	Best	Avr	RPD	Best	Avr	RPD	Best	Avr	RPD
30, 2, 2	**13.393**	**13.652**	**0.019**	**13.393**	14.93	0.103	**13.393**	14.79	0.104	17.985	18.602	0.034	17.308	18.565	0.073	13.393	14.491	0.082
30, 2, 3	**13.228**	**13.790**	**0.042**	**13.228**	14.852	0.109	13.727	14.671	0.069	15.672	16.959	0.082	13.782	16.796	0.219	13.381	15.515	0.159
30, 3, 2	**8.854**	**9.556**	**0.079**	**8.854**	10.007	0.130	9.55	11.056	0.158	10.778	12.571	0.166	11.492	12.982	0.130	9.418	12.844	0.364
30, 3, 3	**7.300**	**8.201**	**0.171**	7.385	9.439	0.218	7.385	8.887	0.294	9.077	11.018	0.228	8.501	10.742	0.361	7.385	9.057	0.296
40, 2, 2	**12.889**	**13.95**	0.082	12.991	14.237	0.096	14.199	15.069	0.061	14.199	16.48	0.161	17.013	17.013	**0.000**	13.779	15.060	0.093
40, 2, 3	**13.596**	**14.307**	**0.052**	13.664	14.600	0.069	13.711	15.909	0.160	15.394	21.294	0.383	19.626	23.512	0.198	13.664	15.183	0.111
40, 3, 2	11.218	**12.329**	**0.099**	11.249	12.442	0.106	11.080	13.398	0.209	14.366	17.062	0.188	14.330	17.397	0.214	**10.704**	15.646	0.462
40, 3, 3	**10.523**	**11.471**	**0.090**	11.156	13.040	0.144	10.754	12.745	0.185	12.070	14.838	0.229	12.718	14.810	0.164	11.618	14.500	0.248
50, 2, 2	**15.805**	**17.676**	**0.118**	16.576	18.942	0.125	16.475	19.070	0.158	18.367	21.587	0.175	20.440	23.209	0.135	17.132	22.904	0.337
50, 2, 3	**18.117**	**20.723**	0.144	18.543	21.743	0.147	**18.117**	23.172	0.279	22.674	26.610	0.174	27.267	27.468	**0.008**	19.698	26.262	0.333
50, 3, 2	15.017	**16.548**	0.102	14.766	16.687	0.130	16.506	19.244	0.166	19.374	24.274	0.253	23.370	25.549	**0.093**	**14.158**	16.965	0.198
50, 3, 3	**13.790**	**14.635**	0.061	14.036	14.792	0.054	13.948	15.658	0.123	15.444	18.003	0.166	18.105	18.238	0.007	18.245	18.245	**0.000**
60, 2, 2	**23.908**	**26.159**	**0.094**	23.908	26.501	0.108	26.147	30.235	0.156	27.175	37.389	0.376	34.570	39.883	0.154	27.236	36.455	0.338
60, 2, 3	**18.780**	**21.096**	0.123	19.947	22.858	0.127	21.605	24.226	0.121	22.844	31.119	0.362	25.595	37.426	0.462	19.947	22.016	**0.104**
60, 3, 2	21.192	**23.569**	0.112	**21.078**	23.579	0.119	21.427	26.326	0.229	27.839	28.634	**0.029**	23.806	28.501	0.197	22.527	28.501	0.262
60, 3, 3	15.210	**16.658**	**0.095**	**13.006**	16.824	0.294	16.395	19.295	0.177	16.395	23.441	0.430	23.612	26.119	0.106	15.210	21.530	0.417
70, 2, 2	27.173	**30.208**	**0.112**	27.729	31.377	0.132	30.890	35.114	0.137	29.134	38.398	0.318	34.967	39.858	0.140	**25.535**	36.452	0.428
70, 2, 3	23.791	**25.851**	**0.087**	**23.790**	26.041	0.095	25.039	26.639	0.184	30.763	33.731	0.096	32.290	35.579	0.102	**23.790**	30.132	0.267
70, 3, 2	20.096	**21.914**	**0.090**	**19.800**	22.969	0.138	22.054	24.362	0.105	24.791	28.983	0.169	26.744	29.769	0.113	19.804	28.081	0.418
70, 3, 3	**18.410**	**22.436**	0.219	20.482	23.211	0.133	21.580	25.586	0.186	23.754	32.475	0.367	38.931	41.532	**0.067**	19.236	25.198	0.310
80, 2, 2	26.441	**28.398**	**0.074**	26.714	29.707	0.101	27.392	32.795	0.197	27.800	41.352	0.487	36.382	45.080	0.239	**26.429**	30.676	0.161
80, 2, 3	**32.004**	**37.735**	0.179	33.857	38.217	**0.129**	33.857	42.813	0.265	41.799	56.206	0.345	54.076	69.569	0.287	33.431	39.135	0.171
80, 3, 2	**22.040**	**25.676**	0.165	**22.040**	25.704	0.166	23.761	30.005	0.263	23.023	40.544	0.761	31.024	35.198	**0.135**	27.357	42.133	0.539
80, 3, 3	**21.746**	**24.680**	**0.135**	**21.746**	27.043	0.196	24.694	29.379	0.190	27.283	34.473	0.264	28.182	35.144	0.247	22.947	31.660	0.380
90, 2, 2	**28.580**	**31.603**	0.106	30.535	33.280	**0.090**	34.729	38.821	0.118	34.921	43.223	0.238	40.791	45.577	0.117	29.494	38.496	0.305
90, 2, 3	**28.281**	**31.504**	0.114	29.448	32.744	0.110	30.274	36.319	0.200	35.285	43.397	0.230	38.251	47.390	0.239	30.759	33.582	**0.092**
90, 3, 2	25.021	**26.895**	**0.075**	**24.507**	28.013	0.125	25.814	30.971	0.200	30.225	36.447	0.206	31.205	38.234	0.225	25.501	32.805	0.286
90, 3, 3	**22.044**	**28.229**	0.281	24.322	30.345	0.248	29.324	33.418	0.140	31.172	42.405	0.360	45.896	49.307	**0.074**	24.116	29.298	0.215
100, 2, 2	29.032	**31.278**	**0.077**	29.636	32.635	0.101	32.414	37.086	0.144	37.176	44.461	0.196	33.110	45.415	0.372	**28.576**	39.334	0.376
100, 2, 3	**30.206**	**34.793**	**0.152**	30.615	38.017	0.242	35.48	40.908	0.153	39.212	50.646	0.292	45.232	55.394	0.225	30.671	38.42	0.253
100, 3, 2	**26.083**	**28.092**	**0.077**	26.631	31.255	0.174	30.369	36.118	0.189	34.653	42.59	0.229	35.315	41.703	0.181	30.129	40.272	0.337
100, 3, 3	28.973	**31.978**	**0.104**	28.973	33.38	0.152	31.84	37.993	0.193	40.466	48.283	0.193	40.442	44.881	0.110	**27.264**	34.751	0.275
Average	**20.086**	**22.362**	**0.110**	20.456	23.419	0.138	21.998	25.690	0.172	24.722	31.172	0.256	28.262	33.058	0.169	21.017	26.737	0.269

and RPD are obtained by SNPT seven times among nine times. Though Johnson obtains some better results in AVG, Best, and RPD, in terms of overall performance, SNPT searches better AVG results in AVG, Best than Johnson. Therefore, SNPT is chosen to sequence all the jobs at the first stage for our proposed problem.

5.3. The performance analysis of hybrid EDA-DE

In our preliminary experiment, several typical values for each parameter are tried by simply fixing others, and we select the best combinations for our calibration experiment based on the recommending values and preliminary experimental results. The source data of random test instances are generated in the following way. The normal processing time of each job is distributed uniformly over [1,10], and job sizes are randomly chosen from 1 to 3. The deteriorating effect rate for each job also follows a uniform distribution between [0, 1]. For each category of the problem, the instance is run 20 times independently and the average values are reported. Once the iteration for each algorithm reaches 250, the searching process is terminated. The parameters of the test problems are shown in Table 9.

We tested our proposed hybrid EDA-DE algorithm against other five algorithms, they are EDA, DE, CS, PSO, and VNS. The common parameter settings are the same for the six algorithms. Table 10 presents the comparative best and average results and RPD values for the tested 32 instances, which are all obtained through 20 independent runs. Besides, the t tests with 5% significance level are conducted for the proposed hybrid EDA-DE

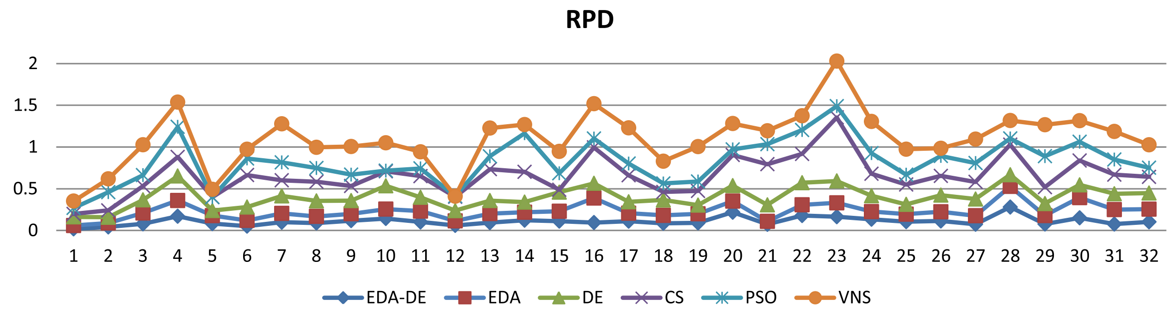


Fig. 6. RPD values in each instance for all algorithm.

Table 11
Comparison between the lower bound and the optimal results and the runtime.

Problem	EDADE		EDA		DE		CS		PSO		VNS	
	Gap	Time (s)	Gap	Time (s)	Gap	Time (s)	Gap	Time (s)	Gap	Time (s)	Gap	Time (s)
30, 2, 2	0.027	12.156	0.264	13.427	0.088	3.430	0.471	2.801	0.471	0.553	0.033	0.926
40, 2, 2	0.014	18.293	0.204	23.567	0.125	5.441	0.213	4.575	0.213	0.905	0.135	1.521
50, 2, 2	0.091	26.818	0.168	37.761	0.149	8.351	0.168	7.551	0.168	1.398	0.159	2.557
60, 2, 2	0.111	39.165	0.249	55.416	0.271	11.614	0.666	10.705	0.666	2.021	0.169	2.764
70, 2, 2	0.116	88.306	0.231	90.305	0.373	24.502	0.531	23.805	0.531	4.705	0.338	6.658
80, 2, 2	0.073	105.194	0.258	144.142	0.193	29.212	0.321	26.828	0.325	5.221	0.155	7.724
90, 2, 2	0.118	139.196	0.447	125.247	0.242	36.553	0.558	34.239	0.558	6.739	0.153	8.489
100, 2, 2	0.090	182.752	0.253	165.982	0.204	49.512	0.623	47.119	0.623	9.139	0.167	11.681
Average	0.080	76.485	0.259	81.981	0.206	21.077	0.444	19.704	0.444	3.835	0.164	5.290

algorithm against EDA, DE, CS, PSO, and VNS. The p and h values calculated by t tests are presented in Table 11. In the t tests analysis, an h value of 1 or -1 means the proposed hybrid EDA-DE algorithm is statistically better or worse than the compared algorithm, while an h value of 0 implies that there is no significant difference between the two algorithms. We also carry out the boxplots experiments to evaluate the stability and quality of the final results obtained by different algorithms.

In Table 10, column 1 represents the job number and machine number at each stage of the current instance. The job number ranges from 30 to 100 while the machine number varies from 2 to 3. Columns 2, 3, 4, 5, 6, and 7 show the best and average makespan as well as the RPD value of each algorithm. The EDA-DE can always get the lowest average values for all instances and the lowest best values in most of the instance. Besides, the EDA-DE, EDA, DE, and VNS obtain the best results in some instances, respectively. For example, in Problem 30, 2, 2, 30, 2, 3, 30, 3, 2, 60, 3, 2, 60, 3, 3, 70, 2, 3, 70, 3, 2, 80, 3, 2, 80, 3, 3, and 90, 3, 2, EDA algorithm obtains the lowest best results among these six algorithms. DE and VNS algorithm also obtain the lowest best results twice and seven times, respectively. Our proposed hybrid EDA-DE algorithm gets the lowest best results 21 times and acquires the lowest average makespan in all instances.

The performances of all involved algorithms are also reported in Table 10. As can be seen, the hybrid EDA-DE algorithm obtains better results than any other algorithms in 20 instances. The RPD values for different algorithms in all instances are shown in Fig. 6.

In order to test the accuracy and efficiency of the hybrid EDA-DE, we design a series of experiments as follows. The gap between the algorithm solution C_{max}^{Sol} and the lower bound C_{max}^{LB} proposed in Section 3 and the CPU time are used as performance measures. The gap is computed as Eq. (5.2)

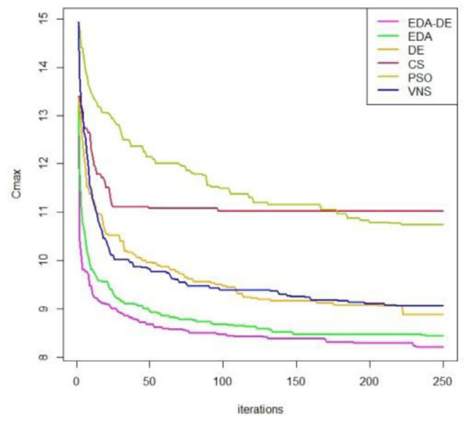
$$Gap = \frac{C_{max}^{Sol} - C_{max}^{LB}}{C_{max}^{LB}} \tag{5.2}$$

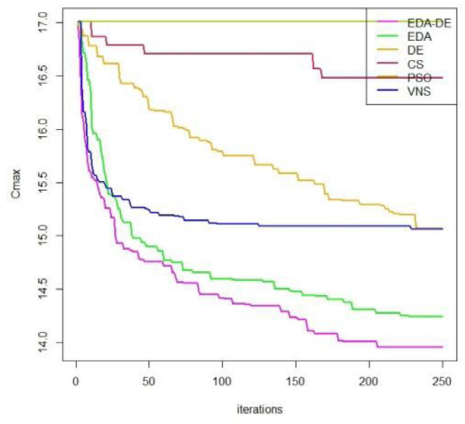
Besides, the runtime (s) is the average results obtained by 20 times independent runs within 160 iterations. The parameters are the same as Table 9 and the problem size are from 30 jobs to 100 jobs while the machine numbers are all the same. From Table 11, it can be seen that the EDA-DE reported an average gap at 0.08 least gap values in all the instances. The runtime is increased as the problem size increases and the longest time to run the hybrid EDA-DE is around 3 min.

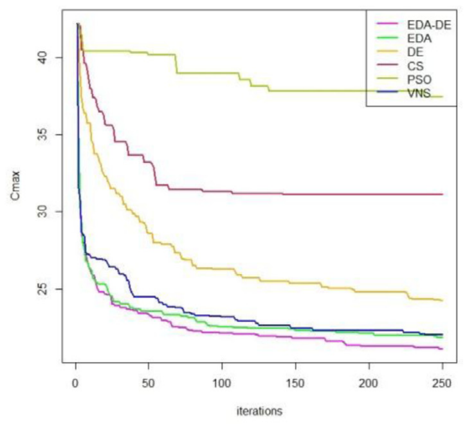
An analysis of variance (ANOVA) is performed to compare 20 final results generated by the six algorithms. Table 12 shows the results of t tests for all 32 instances. Comparing with EDA, EDA-DE yields significantly better results for 30 out of 32 instances, and it is competitive for the remaining instances where there is no statistical difference between the two algorithms. When compared to DE and CS, EDA-DE obtains significantly better results in all instances, which proves the improvement of our proposed hybrid algorithm. Comparing with CS and VNS, out of 32 instances, we can see from Table 12 that EDA-DE can respectively yield 31 and 29 significantly better results.

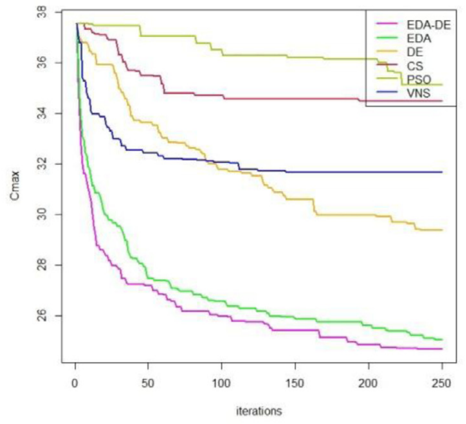
Furthermore, to verify the convergence speed and quality of the compared algorithms, and visually observe the convergence efficiency of the six algorithms, we choose six typical instances to present the convergence curves of different algorithms. A series of convergence curves are shown in Fig. 7. In all instances, the hybrid EDA-DE algorithm has the fastest convergence speed, and it is capable of finding a better solution than all other methods. For example, in the sub-figure (f), six algorithms are applied to solve the problem involving $n = 100$, $m_1 = 2$, and $m_2 = 3$. We observe that EDA-DE is the winner in terms of the overall solution quality and convergence speed. EDA also convergences sharply, but its result is much worse than ours. Considering VNS is a kind of local search algorithm, we generate *P_Size* initial individuals in VNS as same as other five algorithms (EDA-DE, EDA, DE, CS, and PSO) to avoid its premature convergence. Moreover, VNS searches a relative better solution than CS and PSO considering three different neighborhood structures are adopted in the computational experiments. In nearly all cases, fast convergence speed and best solutions are realized with our proposed EDA-DE algorithm.

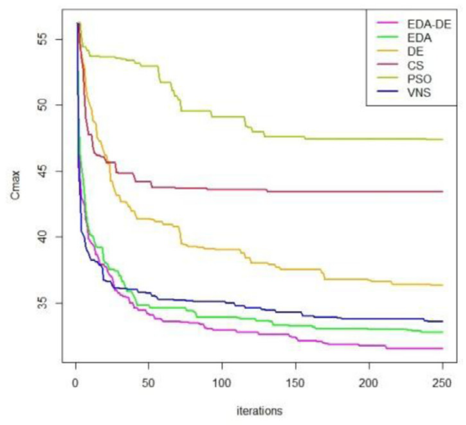
In Fig. 8, we present the boxplots for 8 instances with different job numbers and machine numbers. In all instances, it is clear that our proposed hybrid EDA-DE algorithm acquires the lowest values and shows a relatively stable condition. From the analysis of the above experiments, we can verify the superiority of the


(a) Convergence curve for instance $(30, 3, 3)$


(b) Convergence curve for instance $(40, 2, 2)$


(c) Convergence curve for instance $(60, 2, 3)$


(d) Convergence curve for instance $(80, 3, 3)$


(e) Convergence curve for instance $(90, 2, 3)$

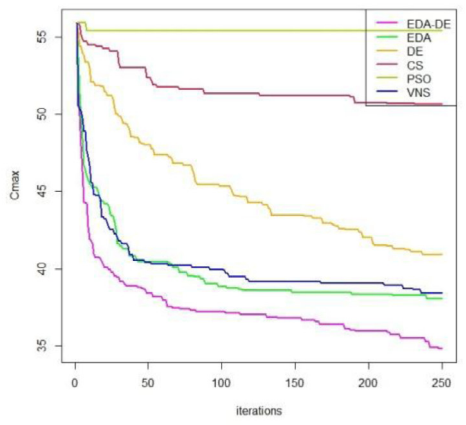
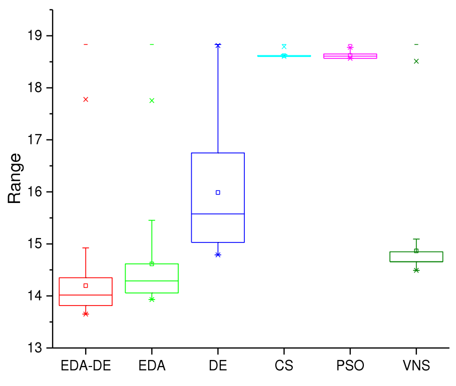

(f) Convergence curve for instance $(100, 2, 3)$

Fig. 7. The convergence curve for six problem categories.

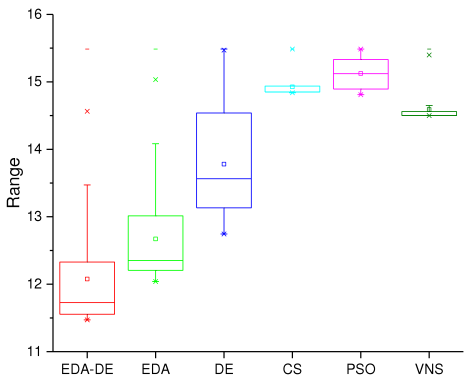
proposed hybrid EDA-DE in terms of values of makespan, RPD, convergence situation, and stability. To sum up, our EDA-DE algorithm is stable and robust in convergence speed and the solution quality to solve the present scheduling problem.

6. Conclusions

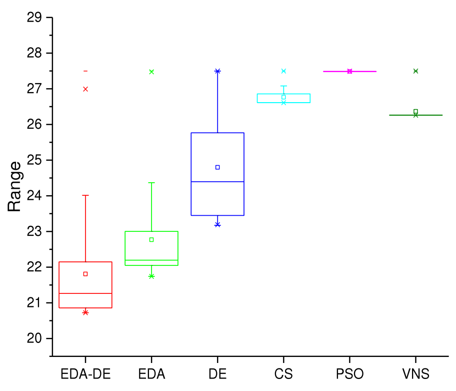
This paper has considered the problem of scheduling parallel batching machines in a two-stage flow shop with job-dependent deteriorating effect and non-identical job sizes such that the makespan is minimized. To the best of our knowledge, this paper is the first in the literature to deal with the problem under study. A three-dimensional wasted volume based on job normal processing time, job size, and job deteriorating rate is first proposed and some structural properties are derived for the single machine case. Since the investigated problem is NP-hard, we proposed a hybrid EDA-DE algorithm to solve our problem in a reasonable time. Besides, computational experiments are conducted and the



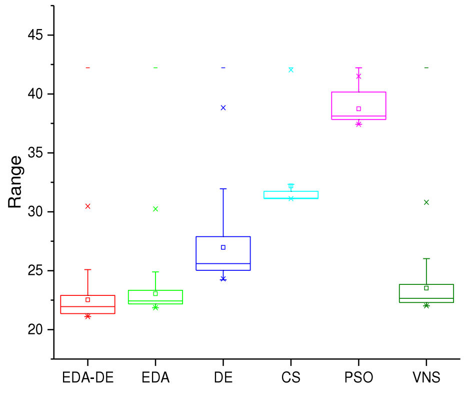
(g) Boxplot for instance (30, 2, 2)



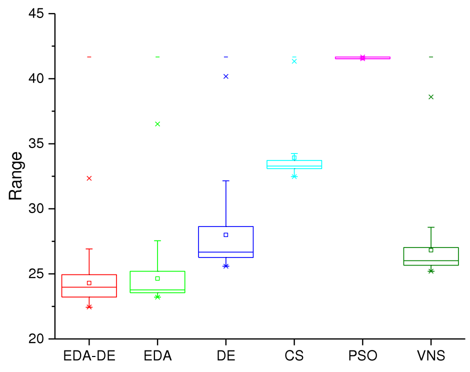
(h) Boxplot for instance (40, 3, 3)



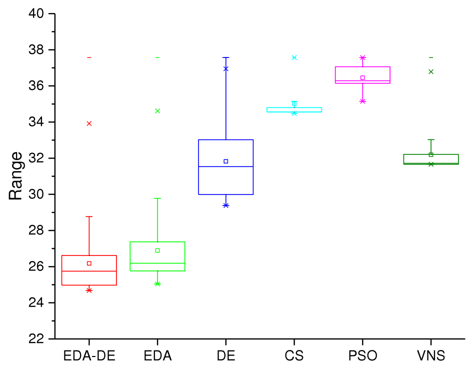
(i) Boxplot for instance (50, 2, 3)



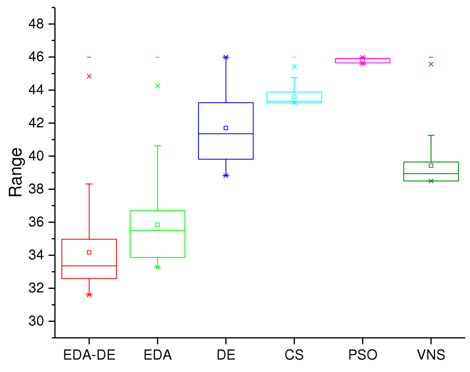
(j) Boxplot for instance (60, 2, 3)



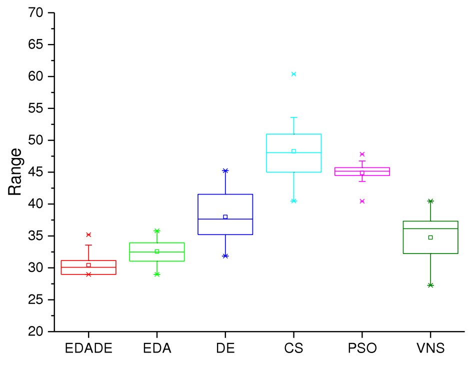
(k) Boxplot for instance (70, 3, 3)



(l) Boxplot for instance (80, 3, 3)



(m) Boxplot for instance (90, 2, 2)



(n) Boxplot for instance (100, 3, 3)

Fig. 8. Boxplot for different instances.

Table 12
The *t* tests for the EDA-DE with the compared algorithms.

Problem	(EDA, EDA-DE)		(DE, EDA-DE)		(CS, EDA-DE)		(PSO, EDA-DE)		(VNS, EDA-DE)	
	p	h	p	h	p	h	p	h	p	h
30, 2, 2	**0.16091**	**0**	0.000461	1	7.44988E−27	1	2.35415E−25	1	0.018621755	1
30, 2, 3	0.001818038	1	1.13526E−06	1	8.37155E−29	1	2.48077E−19	1	4.76829E−06	1
30, 3, 2	9.27491E−05	1	9.80144E−10	1	1.16984E−18	1	2.98971E−25	1	8.86669E−16	1
30, 3, 3	0.048672809	1	5.52863E−05	1	2.02355E−12	1	1.08307E−12	1	**0.519706155**	**0**
40, 2, 2	**0.053919578**	**0**	2.09658E−08	1	7.05623E−13	1	8.57126E−28	1	0.001535842	1
40, 2, 3	0.00965838	1	0.000244352	1	1.06887E−09	1	0.001411827	1	1.38289E−24	1
40, 3, 2	0.017706547	1	0.002497208	1	1.8379E−12	1	**0.311711744**	**0**	0.018923195	1
40, 3, 3	0.000210463	1	5.65955E−06	1	6.84458E−16	1	8.64762E−19	1	2.79435E−12	1
50, 2, 2	0.008356235	1	2.26052E−06	1	2.54539E−13	1	5.21134E−23	1	1.58178E−07	1
50, 2, 3	0.009017206	1	0.000244705	1	1.76E−15	1	7.91336E−25	1	6.93206E−11	1
50, 3, 2	0.005319361	1	5.95073E−10	1	4.76767E−13	1	2.55598E−26	1	0.006978471	1
50, 3, 3	0.000656718	1	3.59246E−08	1	1.17141E−21	1	2.68265E−29	1	2.37813E−29	1
60, 2, 2	0.002981126	1	9.39706E−08	1	4.79168E−13	1	2.51467E−27	1	5.81312E−11	1
60, 2, 3	0.012601008	1	7.85556E−09	1	2.36771E−11	1	1.39846E−15	1	0.01719307	1
60, 3, 2	0.008815764	1	7.35513E−07	1	9.14641E−21	1	7.39898E−16	1	2.04571E−14	1
60, 3, 3	0.029330736	1	1.81729E−08	1	8.73247E−14	1	2.52217E−34	1	6.29098E−07	1
70, 2, 2	0.009699375	1	5.13427E−11	1	8.42274E−14	1	3.3949E−22	1	7.17344E−08	1
70, 2, 3	0.001880174	1	3.88163E−09	1	1.48568E−18	1	5.08135E−28	1	4.85839E−05	1
70, 3, 2	0.016379776	1	1.27245E−08	1	1.46895E−20	1	7.86407E−27	1	6.47745E−09	1
70, 3, 3	0.005298522	1	1.69366E−06	1	8.84415E−12	1	3.48839E−34	1	0.036400127	1
80, 2, 2	0.009592368	1	9.99676E−07	1	5.8963E−14	1	1.10008E−25	1	0.00092539	1
80, 2, 3	0.008738724	1	1.91624E−05	1	5.24219E−10	1	2.44305E−19	1	**0.131228806**	**0**
80, 3, 2	0.026333231	1	4.60581E−07	1	7.73443E−10	1	8.92407E−20	1	7.03505E−12	1
80, 3, 3	0.005341447	1	1.04479E−09	1	1.47561E−14	1	8.45758E−16	1	3.09798E−06	1
90, 2, 2	0.000248334	1	3.25701E−14	1	4.9272E−16	1	7.1092E−29	1	3.95117E−07	1
90, 2, 3	0.0068486	1	9.73059E−08	1	1.07863E−12	1	3.8774E−13	1	0.004325934	1
90, 3, 2	0.005349712	1	1.69078E−07	1	6.00904E−16	1	1.2692E−18	1	5.01962E−05	1
90, 3, 3	0.006794538	1	9.29812E−08	1	2.59121E−13	1	4.41743E−29	1	**0.243864959**	**0**
100, 2, 2	0.000781994	1	6.96673E−09	1	4.91885E−20	1	3.32537E−21	1	8.06809E−07	1
100, 2, 3	0.000813532	1	3.12789E−08	1	2.75466E−16	1	1.9066E−27	1	0.020264935	1
100, 3, 2	0.009838594	1	3.56798E−08	1	1.00833E−15	1	3.32823E−16	1	8.23506E−09	1
100, 3, 3	0.00057526	1	5.84834E−10	1	1.98765E−18	1	9.17712E−25	1	5.10421E−05	1

results show that our proposed hybrid algorithm outperforms EDA, DE, CS, PSO, and VNS. Particularly, the hybrid EDA-DE algorithm is robust for finding better solutions in both small scale and large scale experiments.

For future research, we can apply different meta-heuristics to solve flow shop scheduling with different features, such as no-wait permutation flow shop, distributed permutation flow shop. Also, multitasking scheduling models and deteriorating effect can be considered simultaneously, for which heuristics or exact methods can be developed. Other practical objectives such as the maximum lateness, total weighted completion time, and the minimum scheduling cost, multi-objective scheduling problems are also worth being investigated.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2019.105701.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Nos. 71871080, 71601065, 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), and Base of Introducing Talents of Discipline to Universities for Optimization and Decision-making in the Manufacturing Process of Complex Product (111 project).

References

[1] J.B. Wang, Z.Q. Xia, Flow shop scheduling with deteriorating jobs under dominating machines, Omega 34 (4) (2006) 327–336.
[2] W.C. Lee, C.C. Wu, C.C. Wen, Y.H. Chung, A two-machine flowshop makespan scheduling problem with deteriorating jobs, Comput. Ind. Eng. 54 (4) (2008) 737–749.
[3] L.N. Xing, Y.W. Chen, P. Wang, Q.S. Zhao, J. Xiong, A knowledge-based ant colony optimization for flexible job shop scheduling problems, Appl. Soft Comput. 10 (3) (2010) 888–896.
[4] J. Xiong, R. Leus, Z. Yang, H.A. Abbass, Evolutionary multi-objective resource allocation and scheduling in the Chinese navigation satellite system project, European J. Oper. Res. 251 (2) (2016) 662–675.
[5] H. Luo, G.Q. Huang, Y. Feng Zhang, Q. Yun Dai, Hybrid flowshop scheduling with batch-discrete processors and machine maintenance in time windows, Int. J. Prod. Res. 49 (6) (2011) 1575–1603.
[6] D. Lei, T. Wang, An effective neighborhood search algorithm for scheduling a flow shop of batch processing machines, Comput. Ind. Eng. 61 (3) (2011) 739–743.
[7] T.C. Huang, B.M. Lin, Batch scheduling in differentiation flow shops for makespan minimisation, Int. J. Prod. Res. 51 (17) (2013) 5073–5082.
[8] D. Li, X. Meng, Q. Liang, J. Zhao, A heuristic-search genetic algorithm for multi-stage hybrid flow shop scheduling with single processing machines and batch processing machines, J. Intell. Manuf. 26 (5) (2015) 873–890.
[9] H. Gong, L. Tang, C.W. Duin, A two-stage flow shop scheduling problem on a batching machine and a discrete machine with blocking and shared setup times, Comput. Oper. Res. 37 (5) (2010) 960–969.
[10] M. Rabiee, R.S. Rad, M. Mazinani, R. Shafaei, An intelligent hybrid meta-heuristic for solving a case of no-wait two-stage flexible flow shop scheduling problem with unrelated parallel machines, Int. J. Adv. Manuf. Technol. 71 (5–8) (2014) 1229–1245.
[11] S. Wang, M. Liu, C. Chu, A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling, Int. J. Prod. Res. 53 (4) (2015) 1143–1167.
[12] X. Feng, F. Zheng, Y. Xu, Robust scheduling of a two-stage hybrid flow shop with uncertain interval processing times, Int. J. Prod. Res. 54 (12) (2016) 3706–3717.
[13] C.T. Ng, J.B. Wang, T.E. Cheng, L.L. Liu, A branch-and-bound algorithm for solving a two-machine flow shop problem with deteriorating jobs, Comput. Oper. Res. 37 (1) (2010) 83–90.

[14] J.B. Wang, C.D. Ng, T.E. Cheng, L.L. Liu, Minimizing total completion time in a two-machine flow shop with deteriorating jobs, Appl. Math. Comput. 180 (1) (2006) 185–193.

[15] J. Pei, X. Liu, B. Liao, P.M. Pardalos, M. Kong, Single-machine scheduling with learning effect and resource-dependent processing times in the serial-batching production, Appl. Math. Model. (2017) http://dx.doi.org/10.1016/j.apm.2017.07.028.

[16] C.C. Wu, W.C. Lee, Two-machine flowshop scheduling to minimize mean flow time under linear deterioration, Int. J. Prod. Econ. 103 (2) (2006) 572–584.

[17] Y.R. Shiau, M.S. Tsai, W.C. Lee, T.E. Cheng, Two-agent two-machine flowshop scheduling with learning effects to minimize the total completion time, Comput. Ind. Eng. 87 (2015) 580–589.

[18] Y. Tan, J.E. Carrillo, Strategic analysis of the agency model for digital goods, Prod. Oper. Manage. 26 (4) (2017) 724–741.

[19] X. Geng, Y. Tan, L. Wei, How add-on pricing interacts with distribution contracts, Prod. Oper. Manage. 27 (4) (2018) 605–623.

[20] Y. Marinakis, M. Marinaki, A. Migdalas, An adaptive bumble bees mating optimization algorithm, Appl. Soft Comput. 55 (2017) 13–30.

[21] O. Shahvari, R. Logendran, A comparison of two stage-based hybrid algorithms for a batch scheduling problem in hybrid flow shop with learning effect, Int. J. Prod. Econ. 195 (2018) 227–248.

[22] J. Pei, B. Cheng, X. Liu, P.M. Pardalos, M. Kong, Single-machine and parallel-machine serial-batching scheduling problems with position-based learning effect and linear setup time, Ann. Oper. Res. (2017) http://dx.doi.org/10.1007/s10479-017-2481-8.

[23] J. Pei, X. Liu, B. Liao, P.M. Pardalos, M. Kong, Single-machine scheduling with learning effect and resource-dependent processing times in the serial-batching production, Appl. Math. Model. 58 (2018) 245–253.

[24] J. Pei, X. Liu, P.M. Pardalos, A. Migdalas, S. Yang, Serial-batching scheduling with time-dependent setup time and effects of deterioration and learning on a single-machine, J. Global Optim. 67 (1–2) (2017) 251–262.

[25] J. Pei, X. Liu, W. Fan, P.M. Pardalos, S. Lu, A hybrid BA-VNS algorithm for coordinated serial-batching scheduling with deteriorating jobs, financial budget, and resource constraint in multiple manufacturers, Omega (2017) http://dx.doi.org/10.1016/j.omega.2017.12.003.

[26] S. Lu, X. Liu, J. Pei, P.M. Pardalos, Permutation flowshop manufacturing cell scheduling problems with deteriorating jobs and sequence dependent setup times under dominant machines, Optim. Lett. (2018) http://dx.doi.org/10.1007/s11590-018-1322-2.

[27] S. Lu, X. Liu, J. Pei, M.T. Thai, P.M. Pardalos, A hybrid ABC-TS algorithm for the unrelated parallel-batching machines scheduling problem with deteriorating jobs and maintenance activity, Appl. Soft Comput. 66 (2018) 168–182.

[28] X. Liu, S. Lu, J. Pei, P.M. Pardalos, A hybrid VNS-HS algorithm for a supply chain scheduling problem with deteriorating jobs, Int. J. Prod. Res. (2017) 1–18.

[29] M. Kong, X. Liu, J. Pei, P.M. Pardalos, N. Mladenovic, Parallel-batching scheduling with nonlinear processing times on a single and unrelated parallel machines, J. Global Optim. (2018) http://dx.doi.org/10.1007/s10898-018-0705-3.

[30] O. Kheirandish, R. Tavakkoli-Moghaddam, M. Karimi-Nasab, An artificial bee colony algorithm for a two-stage hybrid flowshop scheduling problem with multilevel product structures and requirement operations, Int. J. Comput. Integr. Manuf. 28 (5) (2015) 437–450.

[31] A. Rossi, A. Puppato, M. Lanzetta, Heuristics for scheduling a two-stage hybrid flow shop with parallel batching machines: application at a hospital sterilisation plant, Int. J. Prod. Res. 51 (8) (2013) 2363–2376.

[32] Q.K. Pan, R. Ruiz, An estimation of distribution algorithm for lot-streaming flow shop problems with setup times, Omega 40 (2) (2012) 166–180.

[33] K. Wang, S.H. Choi, H. Qin, An estimation of distribution algorithm for hybrid flow shop scheduling under stochastic processing times, Int. J. Prod. Res. 52 (24) (2014) 7360–7376.

[34] H. Zhou, J. Pang, P.K. Chen, F.D. Chou, A modified particle swarm optimization algorithm for a batch-processing machine scheduling problem with arbitrary release times and non-identical job sizes, Comput. Ind. Eng. 123 (2018) 67–81.

[35] S.J. Yang, D.L. Yang, T.E. Cheng, Single-machine due-window assignment and scheduling with job-dependent aging effects and deteriorating maintenance, Comput. Oper. Res. 37 (8) (2010) 1510–1514.

[36] H. Mühlenbein, G. Paass, From recombination of genes to the estimation of distributions I. Binary parameters, in: International Conference on Parallel Problem Solving from Nature, Springer, Berlin, Heidelberg, 1996, pp. 178–187.

[37] J.N. Shen, L. Wang, S.Y. Wang, A bi-population EDA for solving the no-idle permutation flow-shop scheduling problem with the total tardiness criterion, Knowl.-Based Syst. 74 (2015) 167–175.

[38] Y.Y. Han, D. Gong, X. Sun, A discrete artificial bee colony algorithm incorporating differential evolution for the flow-shop scheduling problem with blocking, Eng. Optim. 47 (7) (2015) 927–946.

[39] Y. Liu, M. Yin, W. Gu, An effective differential evolution algorithm for permutation flow shop scheduling problem, Appl. Math. Comput. 248 (2014) 143–159.

[40] H. Chen, S. Zhou, X. Li, R. Xu, A hybrid differential evolution algorithm for a two-stage flow shop on batch processing machines with arbitrary release times and blocking, Int. J. Prod. Res. 52 (19) (2014) 5714–5734.

[41] Y. Zhang, X. Li, Estimation of distribution algorithm for permutation flow shops with total flowtime minimization, Comput. Ind. Eng. 60 (4) (2011) 706–718.

[42] Y.R. Tzeng, C.L. Chen, C.L. Chen, A hybrid EDA with ACS for solving permutation flow shop scheduling, Int. J. Adv. Manuf. Technol. 60 (9–12) (2012) 1139–1147.

[43] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.

[44] S.H. Yang, J.B. Wang, Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration, Appl. Math. Comput. 217 (9) (2011) 4819–4826.

[45] J.B. Wang, P. Ji, T.C.E. Cheng, D. Wang, Minimizing makespan in a two-machine flow shop with effects of deterioration and learning, Optim. Lett. 6 (7) (2012) 1393–1409.

[46] Z.H. Jia, C. Wang, J.Y.T. Leung, An ACO algorithm for makespan minimization in parallel batch machines with non-identical job sizes and incompatible job families, Appl. Soft Comput. 38 (2016) 395–404.

[47] Z.H. Jia, Y.L. Zhang, J.Y.T. Leung, K. Li, Bi-criteria ant colony optimization algorithm for minimizing makespan and energy consumption on parallel batch machines, Appl. Soft Comput. 55 (2017) 226–237.

[48] M. Ji, T.E. Cheng, Batch scheduling of simple linear deteriorating jobs on a single machine to minimize makespan, European J. Oper. Res. 202 (1) (2010) 90–98.

[49] J. Behnamian, S.F. Ghomi, F. Jolai, O. Amirtaheri, Minimizing makespan on a three-machine flowshop batch scheduling problem with transportation using genetic algorithm, Appl. Soft Comput. 12 (2) (2012) 768–777.

[50] E. Cakici, S.J. Mason, J.W. Fowler, H.N. Geismar, Batch scheduling on parallel machines with dynamic job arrivals and incompatible job families, Int. J. Prod. Res. 51 (8) (2013) 2462–2477.

[51] J. Pei, P.M. Pardalos, X. Liu, W. Fan, S. Yang, Serial batching scheduling of deteriorating jobs in a two-stage supply chain to minimize the makespan, European J. Oper. Res. 244 (1) (2015) 13–25.

[52] R. Ruiz, J.A. Vázquez-Rodríguez, The hybrid flow shop scheduling problem, European J. Oper. Res. 205 (1) (2010) 1–18.

[53] R. Uzsoy, Scheduling a single batch processing machine with non-identical job sizes, Int. J. Prod. Res. 32 (7) (1994) 1615–1635.

[54] M.R. Garey, D.S. Johnson, R. Sethi, The complexity of flowshop and jobshop scheduling, Math. Oper. Res. 1 (2) (1976) 117–129.

[55] O. Ozturk, M.L. Espinouse, M.D. Mascolo, A. Gouin, Makespan minimisation on parallel batch processing machines with non-identical job sizes and release dates, Int. J. Prod. Res. 50 (20) (2012) 6022–6035.

[56] S.Y. Wang, L. Wang, M. Liu, Y. Xu, An effective estimation of distribution algorithm for solving the distributed permutation flow-shop scheduling problem, Int. J. Prod. Econ. 145 (1) (2013) 387–396.

[57] S. Yang, J. Wang, L. Shi, Y. Tan, F. Qiao, Engineering management for high-end equipment intelligent manufacturing, Front. Eng. Manage. 5 (4) (2018) 420–450.

[58] B. Jarboui, M. Eddaly, P. Siarry, An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems, Comput. Oper. Res. 36 (9) (2009) 2638–2646.

[59] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.

[60] P. Larrañaga, J.A. Lozano (Eds.), Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, Vol. 2, Springer Science & Business Media, 2001.

[61] B. Liu, L. Wang, Y.H. Jin, An effective PSO-based memetic algorithm for flow shop scheduling, IEEE Trans. Syst. Man Cybern. B 37 (1) (2007) 18–27.

[62] J.Q. Li, Q.K. Pan, F.T. Wang, A hybrid variable neighborhood search for solving the hybrid flow shop scheduling problem, Appl. Soft Comput. 24 (2014) 63–77.

[63] X.S. Yang, S. Deb, Cuckoo search via Lévy flights, in: Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, IEEE, 2009, pp. 210–214.

[64] B. Naderi, R. Ruiz, The distributed permutation flowshop scheduling problem, Comput. Oper. Res. 37 (4) (2010) 754–768.