Int J Adv Manuf Technol (2016) 86:3315–3331
DOI 10.1007/s00170-016-8408-8

ORIGINAL ARTICLE

Minimizing the makespan for a serial-batching scheduling problem with arbitrary machine breakdown and dynamic job arrival

Jun Pei[1,2,3] · **Xinbao Liu**[1,3] · **Wenjuan Fan**[1,4] · **Panos M. Pardalos**[2] · **Athanasios Migdalas**[5] · **Boris Goldengorin**[6] · **Shanlin Yang**[1,3]

Received: 13 November 2015 / Accepted: 19 January 2016 / Published online: 12 February 2016

Abstract Many dynamic events exist in real manufacturing systems, such as arbitrary machine breakdowns and dynamic job arrivals, which makes the scheduling problem even more complicated. In this paper, we address a serial-batching scheduling problem with the above dynamic events. Jobs need to be processed on the serial-batching machines of two manufacturers and then transported by vehicles to a customer for further processing. The objective of the scheduling problem is to minimize the makespan, and the problem is proved to be strongly NP-hard. Some structural properties and a lower bound of the problem are also proved or derived. On the basis of job arrival times, we divide the problem into two phases and propose different rules regarding these two phases. Based on these properties and rules, a heuristic algorithm is developed to solve the problem and its worst case performance is analyzed. The heuristic algorithm is tested on a large set of randomly generated problem instances, and the relative gaps between the found lower bound and the solutions of the proposed heuristic algorithm are reported. The experimental results illustrate the high efficiency and effectiveness of the proposed heuristic algorithm compared with other four classic approaches.

Keywords Serial-batching scheduling · Heuristic algorithm · Dynamic arrival · Machine breakdown · Setup time

✉ 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, USA

[3] Key Laboratory of Process Optimization and Intelligent Decision-making of Ministry of Education, Hefei, China

[4] Department of Computer Science, North Carolina State University, Raleigh, USA

[5] Division of Industrial Logistics, Department of Industrial Engineering, Lulea University of Technology, Sweden & Division of Transportation, Construction Management and Regional Planning, Department of Civil Engineering, Aristotle University of Thessaloniki, Thessaloniki, Greece

[6] Department of Industrial Engineering, Russ College of Engineering and Technology, Ohio University, Athens, Ohio, USA

Abbreviations

RFID	Radio frequency identification
IoT	Internet of Things
GA	Genetic algorithm
FFS	Flexible flow shop
VNS	Variable neighborhood search
DPMSPs	Dynamic parallel machine scheduling problems
RSA	Restricted simulated annealing
MCAC	Multi-contextual ant colony
TS	Tabu search
ICA	Imperialist competitive algorithm
DEA	Differential evolution algorithm
PSO	Particle swarm optimization
MTBF	Mean time of the intervals between the machine breakdowns
MTTR	Mean time of the intervals between repairing the machines
FIFO	First in first out dispatching rule
SPT	The shortest processing time dispatching rule
LPT	The largest processing time dispatching rule

1 Introduction

Nowadays, efficient scheduling cooperation among supply chain partners has become essential in such a competitive business environment. However, many dynamic events exist in real manufacturing systems, such as arbitrary machine breakdowns and dynamic job arrivals, which make the scheduling cooperation problem in supply chain even more complicated, and thus more effective and efficient approaches are needed to solve such problems. Technological advances such as radio frequency identification (RFID) and Internet of Things (IoT) make it possible for the supply chain partners to dynamically share information during production and promptly respond by adjusting the schedules to emerging situations. To this end, this study focuses on an ad hoc serial-batching scheduling problem in a two-stage supply chain, which arises from the real scenario in the aluminum production supply chain. The first stage is composed of two manufacturers (i.e., two extrusion factories) producing the jobs ordered by a customer, in which the situations of machine breakdown, dynamic job arrival, and setup time are considered simultaneously. In the second stage, the vehicles carry the jobs from the manufacturers to the customer.

The remainder of this paper is organized as follows: we start Section 2 with literature review. The problem description is presented in Section 3. In Section 4, we propose a mathematical model of this scheduling problem and prove that it is NP-hard. The structural properties and the lower bound are proved and derived in Section 5. In Section 6, some rules and a heuristic algorithm are proposed to solve the serial-batching scheduling problem and the worst case performance of the proposed algorithm is analyzed. In Section 7, computational experiments are presented to evaluate the effectiveness of the proposed algorithm. We conclude the paper with a summary and provide future research directions in Section 8.

2 Literature review

In this paper, the scheduling problem has the following features: machine breakdowns, dynamic job arrivals, setup times, supply chain scheduling, and serial-batching scheduling. In the last decades, a number of studies have focused on the scheduling problems with these above features, and in the following, we review these papers, respectively.

Machine breakdown is a common event during the production due to maintenance failure of the machines, which are studied extensively. Hasan et al. [1] focused on two scenarios of machine unavailability in the job-shop scheduling problem. They developed a genetic algorithm (GA) to solve the problem, combining an improved local search technique. Wang and Choi [2] investigated a flexible flow shop (FFS) scheduling problem in consideration of machine breakdown, and the objective is to minimize the makespan. A novel decomposition-based approach was proposed to decompose the problem into several cluster scheduling problems. Then, these problems were solved by different approaches. Benmansour et al. [3] studied a single machine scheduling problem, where the processing times of the jobs were exponentially distributed, and the common due date follows the Erlang distribution. The objective is to minimize the expected total weighted deviations of completion times. The optimal schedules were given when the machine was subject to arbitrary breakdowns. Lee and Kim [4] considered the scheduling problem on a single machine, where periodic maintenance is required. Their objective is to minimize the number of tardy jobs. A two-phase heuristic algorithm was presented to solve the problem. Computational experiments on randomly generated problem instances were carried out, and the results showed that the proposed heuristic algorithm had high solution quality. Mirabi et al. [5] addressed a two-stage hybrid flowshop scheduling problem under machine breakdown, where the probability of machine failure depends on the previous processed job. One optimal approach for job precedence was introduced to solve the problem, and some experiments were carried out to examine the performance of the approach. Xiong et al. [6] focused on a flexible job-shop scheduling problem with random machine breakdowns. They considered two objectives simultaneously, i.e., makespan and robustness. A multi-objective evolutionary algorithm was proposed to solve the problem. The experimental results showed that the proposed algorithm performed better for both small and large cases compared with existing measures.

The scheduling problems with dynamic job arrival have appeared in a wide range of practical production systems. Zandieha and Adibi [7] considered the situation of random job arrivals and machine breakdowns in a dynamic job scheduling problem. They introduced variable neighborhood search (VNS) to solve the problem and used an artificial neural network to update parameters of the VNS. Lee et al. [8] studied dynamic parallel machine scheduling problems (DPMSPs) with sequence-dependent setup times. A restricted simulated annealing (RSA) algorithm was designed to solve the problem, combining a restricted search strategy. The computational experiments demonstrated that the proposed algorithm was highly effective compared to existing algorithms. Chiang et al. [9] investigated a scheduling problem in the wafer fabrication facility, in which the jobs arrived at the machines at different time instants. A memetic algorithm with a new genome encoding scheme was used to find both batch formation and batch sequence. The computational efficiency of the proposed algorithm was demonstrated through the experiment with a large number of the instances. Yao et al. [10] addressed a single batch machine scheduling problem with incompatible job families and dynamic job arrivals, and their objective is to minimize the total completion time. Several dominance properties and two types of lower bounds were presented. Based on the characteristics of dynamic job arrivals, a decomposed branch and bound algorithm was proposed to solve the problem. Yao et al. [11] studied a two-stage hybrid flow shop in semiconductor manufacturing industry, where a discrete machine was followed by a batching machine. The computational complexity of the two-machine problem with dynamic job arrivals was analyzed. The heuristic algorithms were proposed for the problem, and the upper bounds on the worst case performance ratios of the heuristics were also established. Lu and Romanowski [12] focused on job shops with dynamic job arrivals. A new theory of context-dependent and multi-contextual scheduling functions was proposed, combining three multi-contextual ant colony (MCAC) scheduling methods. The experimental results indicated that the new theory can produce effective schedules.

In many real-life situations, the setup operations are often required. There have been a number of studies regarding constraints related to setup time [13,14]. Recently, Ciavotta et al. [15] investigated a bi-objective coordination scheduling problem in a two-stage industrial system, in which a setup is required when a new batch has different attribute from the previous batch. Three effective heuristic algorithms were proposed to solve the problem. Roshanaei et al. [16] addressed the problem of scheduling a job shop with sequence-dependent setup times, and the objective is to minimize makespan. An effective metaheuristic algorithm based on the VNS was designed to solve the problem. They conducted experiments based on Taillard's benchmark. The results showed the high performance of the proposed algorithm compared to other well-known heuristic algorithms. Shahvari et al. [17] addressed the flexible flow shop sequence-dependent group scheduling problem with the objective of minimizing the makespan. They developed six efficient metaheuristic algorithms based on tabu search (TS) to solve the problem. Varmazyar and Salmasi [18] studied the flow shop scheduling problems with sequence-dependent setup times, and they proposed several metaheuristic algorithms based on TS and the imperialist competitive algorithm (ICA) to solve the problem with the objective of minimizing the number of tardy jobs. Thurer et al. [19] investigated the influence of sequence-dependent setup times on the performance of a workload-controlled job shop. New setup-oriented dispatching rules were introduced to solve the problem. Nagano et al. [20] studied an m-machine no-wait flow shop problem, in which the setup time of a job is separated from its processing time. The objective of the problem is to minimize the total flowtime, and they designed a new hybrid metaheuristic Genetic Algorithm–Cluster Search to solve it. Moreover, the superiority of the proposed method was confirmed by the experimental tests. Chakaravarthy et al. [21] considered the problem of m-machine flow shop with lot streaming and setup time, and a differential evolution algorithm (DEA) and a particle swarm optimization (PSO) algorithm were proposed to find the best jobs sequence for minimizing the makespan.

In order to enhance competitiveness of supply chain, the cooperation among suppliers, manufacturers, distributers, and customers becomes more and more important. Under this consideration, Hall and Potts [22] first proposed the concept of supply chain scheduling in 2003. After that, supply chain scheduling problems have received much attention of researchers in the last decade. Chandra [23] addressed the joint problem of warehouse procurement decisions and delivery to retailers for multiple products, and a heuristic algorithm was designed for the problem. Su et al. [24] considered a two-stage supply chain scheduling problem with minimization of the makespan, in which the jobs are processed by two parallel machines and delivered to a customer. A heuristic algorithm was proposed to solve it, and its worst case ratio was proved to be 63/40. Delavar et al. [25] focused on a coordinated supply chain scheduling problem of production and air transportation with the objective of optimizing customer service at minimum total cost. They designed two genetic algorithm approaches to solve it. Bard and Nananukul [26] studied an integrated production and inventory routing problem in a supply chain. The objective is to minimize the sum of production, inventory, and delivery costs across the various stages of the system. A hybrid methodology was presented for the problem, combining exact and heuristic procedures within a branch-and-price framework, and a novel column generation heuristic algorithm and a rounding heuristic algorithm were also developed to improve algorithmic efficiency. Steinruecke [27] addressed an aluminum supply chain scheduling problem considering both production and transportation. They presented a novel type of mixed-integer decision-making model for the problem, and relax-and-fix heuristic algorithms were designed to solve it. You and Hsieh [28] investigated a single stage assembly problem with transportation allocation. They established the problem as a mixed-integer programming model and proposed a hybrid heuristic algorithm to solve the problem. Cakici et al. [29] studied a supply chain scheduling problem in an integrated production and distribution environment with minimization of the total weighted tardiness and total distribution costs. A number of weighted linear combinations of the objectives were used to aggregate both objectives into a single objective. Based on a genetic algorithm, they developed different heuristic algorithms to solve the problem.

The related research on the job processing way of serial-batching scheduling has been conducted by many researchers [30,31]. Serial-batching scheduling is an important characteristic of this paper. In [32], we investigated a two-stage coordinated scheduling problem, where jobs are first processed on multiple manufacturers' serial-batching machines and then transported by vehicles to a customer. Subsequently, we studied the scheduling problems of a single serial-batching machine with deteriorating job processing times and independent setup time [33,34]. However, in these studies, the important feature of machine breakdown was not taken into consideration.

Although there are an increasing number of publications addressing the scheduling problem with the mentioned features as well as the problem of coordinated scheduling and transportation [35–37], they typically address specific situations which do not really match real-world manufacturing conditions. Obviously, this does not match the status of the real manufacturing conditions. To the best of our knowledge, our paper is the first attempt to full study the problem under consideration. The main contributions of this paper can be summarized as follows:

(1) New research is pursued by taking into account the features of serial batching, arbitrary machine breakdowns, and dynamic job arrivals.
(2) This scheduling problem is proved to be NP-hard.
(3) Some rules and a heuristic algorithm are proposed to solve the serial-batching scheduling problem and the worst case performance of the proposed algorithm is analyzed.

3 Problem description

For supply chain production, most real manufacturing systems usually operate in highly dynamic environment, where several arbitrary disruptions can break the execution of the pre-planned schedules. In this paper, the scheduling of products and vehicles between two manufacturers and a customer in a two-stage supply chain is investigated. The layout of the scheduling problem is shown in Fig. 1. Suppose that there is a set of jobs to be processed in two parallel manufacturers and then transported to a customer. The two manufacturers are actually extrusion factories, and both include one serial-batching machine, i.e., an extrusion machine. The processing time and the size of the jobs, denoted by p_i and s_i ($i = 1,2,\ldots,n$), are independent random variables following arbitrary probability distributions, while the maximum size is given. The jobs partitioned in the same batch are processed and transported together.

The problem involves two stages, i.e., the production stage on the two manufacturers' machines and the transportation stage from the manufacturers to the customer. In the first stage, all jobs arrive at manufacturers dynamically. These jobs are first partitioned into batches and then processed on the two manufacturers' serial-batching machines. Before a new batch is processed, the setup time s is required. Both serial-batching machines are supposed to have the same finite capacity of size c, and the total size of the jobs in a batch cannot exceed c. In the serial-batching production, the jobs in a batch are processed one after another, so that the processing time P_k of batch k is defined as the sum of the processing time of all the jobs in the batch k [38], i.e., $P_k = \sum_{J_i \in b_k} p_i$. The earliest available time for a batch to be processed depends on the largest arrival time of all jobs in that batch. Machine breakdown may occur during the scheduling period, and the information of the breakdown will be transmitted to the center of production management immediately by RFID based on IoT. In the second stage, all batches are transported by vehicles from two manufactures to the customer once their processing is completed. The assumption that the capacity of the vehicles is the same as that of the manufacturers' machines is made, and that any batch from the manufacturers can be carried by vehicles in one shipment. The vehicles' one-way trip time from the two manufacturers to the customer is supposed to be a constant T. There are enough vehicles to transport the batches to the customer as soon as they are completed on the manufacturers' machines.

Following Holthaus [39], Dominic et al. [40], and Zandieh and Adibi [41], the manufacturers' machines are subject to arbitrary breakdowns, which are described in terms of the *MTBF*, *MTTR*, and *Ag*. If a machine breaks down at the instant time t, then it is assumed that the time required to repair the machine at the instant time t is known. Both of the intervals between every two breakdown occurrences and repair times follow exponential distribution, with *MTBF* and *MTTR* denoting the mean time of the intervals between the machine breakdowns and repairing the machines, and $Ag = MTTR/(MTTR + MTBF)$ denoting the breakdown level, i.e., the percentage of time the machine have breakdowns. For instance, if $Ag = 0.1$ and $MTTR = 10$ time units, then $MTBF = 90$ time units. Therefore, a machine has failure after operation for an average of 90 time units, and it is required for an average of 10 time units to repair. During the whole processing of all jobs, if *Ag* is small, then the number of machine breakdowns is relatively small and the intervals between two continuous breakdowns of the same machine are relatively long.

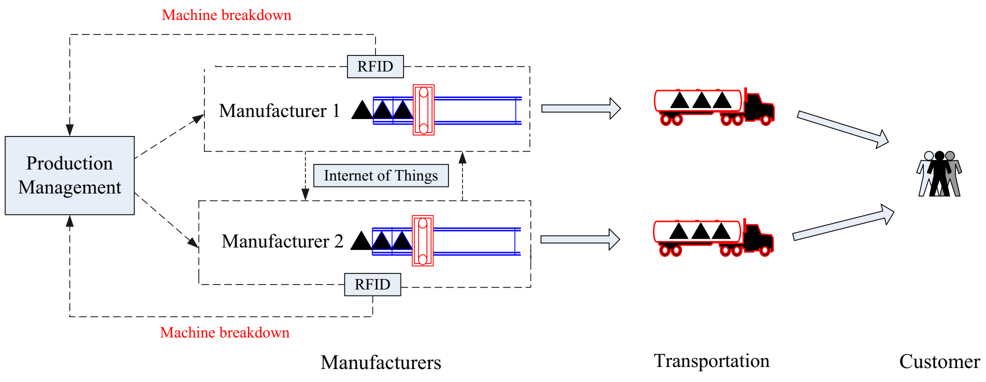


Fig. 1 The layout of the serial-batching scheduling problem in a two-stage supply chain

The aim of the scheduling process is to make jobs batching and sequencing decisions to minimize the makespan of all jobs. For simplicity, the problem is denoted as ψ. The following assumptions are made for the problem formulation:

- All the facilities (machines, vehicles, and RFID) are all available at time zero.
- The size of any job is smaller than the capacity of each machine.
- No preemption is allowed, i.e., once a batch is initiated, no job in the batch can be released until the whole batch is completely processed.
- If the machine breaks down during the process of a batch, then the task of processing the rest unprocessed jobs in the batch should be resumed on the same machine after it is repaired.

4 Complexity analysis

4.1 Model formalization

The notations used for the problem formulation are defined and the model is given as follows.

Parameters:

n Total number of the jobs

i Index of the jobs, $i = 1,2,\ldots,n$

j Index of the manufacturers, $j = 1,2$

p_i Processing time of job i on the manufacturers' machines

s_i Size of the job i

r_i Arrival time of the job i

c The capacity of the batching machines and corresponding vehicles

h Total number of the batches, $\left\lceil \sum_{i=1}^{n} s_i/c \right\rceil \leq h \leq n$

h_j Total number of the batches processed on the machine of the manufacturer j, $0 \leq h_j \leq h$, $j = 1,2$

k, f Index of the batches, $k = 1,2,\ldots,h$, $f = 1,2,\ldots,h$

s Setup time on the manufacturers' machines

n_k The number of jobs in the k-th batch, $k = 1,2,\ldots,h$

m_j Total number of the machine breakdowns in the manufacturer j, $j = 1,2$

T The vehicles' one-way trip time between the manufacturers and the customer

Sets:

J Set of all jobs, $J = \{J_1, J_2,\ldots, J_n\}$

b_k Set of all jobs in the k-th batch, $k = 1,2,\ldots,h$

Decision variables:

xik 1, if the job i is assigned to the k-th batch; 0, otherwise;

$ykfj$ 1, if the k-th batch is processed before the f-th batch on the machine of the manufacturer j,

$j = 1,2$; 0, otherwise;

zkj 1, if the k-th batch is processed by the machine of the manufacturer j, $j = 1,2$; 0, otherwise;

$vkjl$ 1, if the k-th batch is processed by the machine of the manufacturer j and the l-th breakdown on the machine of the manufacturer j happens during the k-th batch's process,

$j = 1,2$, $l = 1,2,\ldots, mj$; 0, otherwise;

gk 1, if the k-th batch is not empty; 0, otherwise;

P_k Total processing time of the jobs in the k-th batch

d_{jl} The time that the l-th breakdown begins on the machine of the manufacturer j, $j = 1,2$, $l = 1,2,\ldots, m_j$

e_{jl} The time that the l-th breakdown ends on the machine of the manufacturer j, $j = 1,2$, $l = 1,2,\ldots, m_j$

S_{1kj} Starting time of the k-th batch processed on the machine of the manufacturer j during the first stage, $j = 1,2$

C_{1kj} Completion time of the k-th batch processed on the machine of the manufacturer j during the first stage, $j = 1,2$

C_{2kj} Arrival time of the k-th batch at the customer from the manufacturer j during the second stage, $j = 1,2$

C^j Maximum completion time of all jobs on the machine of the manufacturer j, $j = 1,2$

$C_{\max}$ Maximum completion time of all jobs during the second stage

Mixed-integer programming model

$$\text{Minimze}\, C_{\max} \tag{1}$$

Subject to

$$\sum_{k=1}^{h} x_{ik} = 1,\ i = 1,2,\cdots,n \tag{2}$$

$$\sum_{i=1}^{n} s_i \cdot x_{ik} \leq c,\ k = 1,2,\cdots,h \tag{3}$$

$$\sum_{j=1}^{2} z_{kj} = 1, k = 1,2,\ldots,h \tag{4}$$

$$h_j = \sum_{k=1}^{h} z_{kj}, j = 1,2 \tag{5}$$

$$h = \sum_{j=1}^{2} hj \tag{6}$$

$$S_{1kj} \geq \max_{J_i \in b_k} \{r_i\} + g_k s,\ k = 1,2,\cdots,h,\ j = 1,2 \tag{7}$$

$$C_{1kj} \geq S_{1kj} + \sum\nolimits_{i=1}^{n} x_{ik} \cdot p_i,\ k = 1,2,\cdots,h,\ j = 1,2 \tag{8}$$

$$v_{kjl}\left(d_{jl} - S_{1kj}\right)\left(C_{1kj} - e_{jl}\right) \geq 0,\ k = 1,2,\cdots,h,\ j = 1,2,\ l = 1,2,\cdots,m_j \tag{9}$$

$$C_{2kj} = C_{1kj} + g_k T,\ k = 1,2,\cdots,h,\ j = 1,2 \tag{10}$$

$$g_f C_{1kj} - C_{1fj} + P_f + g_f s - \left(1 - y_{kfj}\right) M \leq 0,$$
$$k = 1,2,\cdots,h,\ f = 1,2,\cdots,h,\ j = 1,2,\ k \neq f \tag{11}$$

$$C^j \geq s \cdot h_j + \sum\nolimits_{k=1}^{h} z_{kj} \cdot P_k + \sum\nolimits_{l=1}^{m_j} \left(e_{jl} - d_{jl} \right), \; j = 1, 2 \tag{12}$$

$$C_{\max} \geq C^j + T, \; j = 1, 2 \tag{13}$$

$$x_{ik}, y_{kfj}, z_{kj}, v_{kjl} \in \{0, 1\}, \; \forall i, k, f, j, l \tag{14}$$

The objective function (1) is to minimize the makespan. Constraint set (2) assures that each job is only assigned to one batch. Constraint set (3) requires that the total physical size of all jobs assigned to a batch cannot exceed the capacity of the batching machines and corresponding vehicles. Constraint set (4) guarantees that one batch should be processed by only one manufacturer. The number of batches processed on two manufacturers' machines is described by constraint set (5). Constraint set (6) enforces that the total number of the batches processed on two manufacturers' machines is equal to the total number of the batches. Constraint set (7) specifies that the operation of a batch can be started until its last job arrives, and each batch requires setup time before being processed on a manufacturer's machine. Constraint set (8) restricts the completion time for each batch is no smaller than the sum of its starting time and processing time. Constraint set (9) indicates that the jobs cannot be processed during the machine breakdown. The arrival time of all batches at the customer is described by constraint set (10). Constraint set (11) guarantees that there is no overlapping situation between any two different batches. Constraint set (12) describes the maximum completion time of the manufacturers' machines. Constraint set (13) indicates the property of the maximum completion time. The ranges of the variables are defined by constraint sets (14).

4.2 Complexity analysis

In the following, we present the strong NP-hardness proof of the problem ψ without the condition of machine breakdowns.

Theorem 1. *The problem ψ without the condition of machine breakdowns is strongly NP-hard.*

Proof. An instance is constructed to perform the reduction by the following 3-PARTITION problem, which is known to be strongly NP-hard [42].

3-PARTITION: Given an integer a and a set A of $3h$ positive integers $\{a_1, a_2, \ldots, a_{3h}\}$, $a/4 < a_i < a/2$, $1 \leq i \leq 3h$, such as $\sum_{i=1}^{3h} x_i = ha$, does there exist a partition $A_1, A_2, \ldots, A_h$ of the set A such that $|A_k| = 3$ and $\sum_{a_i \in A_k} a_i = a$, $1 \leq k \leq h$?

In order to prove the theorem, we construct the instance of the problem ψ as follows.

Number of the jobs and capacity of the batching machines and vehicles: $n = 3h$ and $c = 3$

Size of the jobs and processing time of the jobs on the manufacturers' machines: $s_i = 1$ and $p_i = a_i$, $i = 1, 2, \ldots, 3h$

One time trip time and setup time: $T = 2a$ and $t = a$

Arrival time of the jobs: $r_j = 0$ for $i = 1, 2, \cdots, 6$ and $r_j = \left(\frac{i}{3} - 2\right) a$ for $i = 7, 8, \cdots 3h$

$h = 2g$, and g is a positive integer

Threshold value: $y = (h + 2)a$

We claim that there is a solution to 3-PARTITION problem if and only if there exists an optimal schedule for the instance of the problem ψ with makespan no greater than $(h + 2)a$.

($\Rightarrow$) The set $A = (A_1, A_2, \ldots, A_h)$ can be constructed to be a partition in this 3-PARTITION problem, and Fig. 2 shows the schedule. It is easy to check that the makespan is $(h + 2)a = y$.

($\Leftarrow$) Conversely, it is supposed that there exists an optimal schedule with makespan no greater than y. Since the smallest number of possible batches is $n * s_i/c = h$, it is obtained that the sum of the processing time and setup time for all batches on the manufacturers' machines is $h * (a + a) = 2ha$. The possible earliest completion time of processing the jobs on two manufacturers' machines is $2ha/2 = ha$. Next, the possible earliest departure time of the last batch in two manufacturers is ha, and $ha + 2a = y$. Therefore, for $y = (h + 2)a$, (a) there is no idle time for the setup on two manufacturers' machines in the interval $[2ia, (2i + 1)a]$ $(i = 0, 1, \ldots, h/2 - 1)$ and (b) there is no idle time for processing the jobs on two manufacturers' machines in the interval $[(2i + 1)a, (2i + 2)a]$ $(i = 0, 1, \ldots, h/2 - 1)$.

The proof can be done by contradiction. Suppose that there is a batch A_k, where $\sum_{J_i \in A_k} p_i \neq a$. There are two situations to consider:

Case 1. $\sum_{J_i \in A_k} p_i > a$. From Fig. 2, it is easy to infer that it creates idle time during the setup on a manufacturer's serial-batching machines for the batch A_k, and this conflicts with (a).

Case 2. $\sum_{J_i \in A_k} p_i < a$. From Fig. 2, we can also see that that idle time can be created during the jobs processing on two manufacturers' serial-batching machines, which conflicts with (b).

Therefore, there exists a solution for 3-PARTITION problem.

Combining the "if" part and the "only if" part, we have proved the proposed theorem.

When machine breakdown may occur, we have the following corollary from Theorem 1.

Corollary 1. *The problem ψ is strongly NP-hard.*

5 The structural properties and lower bound for the problem ψ

5.1 The structural properties

At first, we present two following properties.

Lemma 1 *For all schedules, the solution remains unchanged when any two jobs in a batch are swapped.*

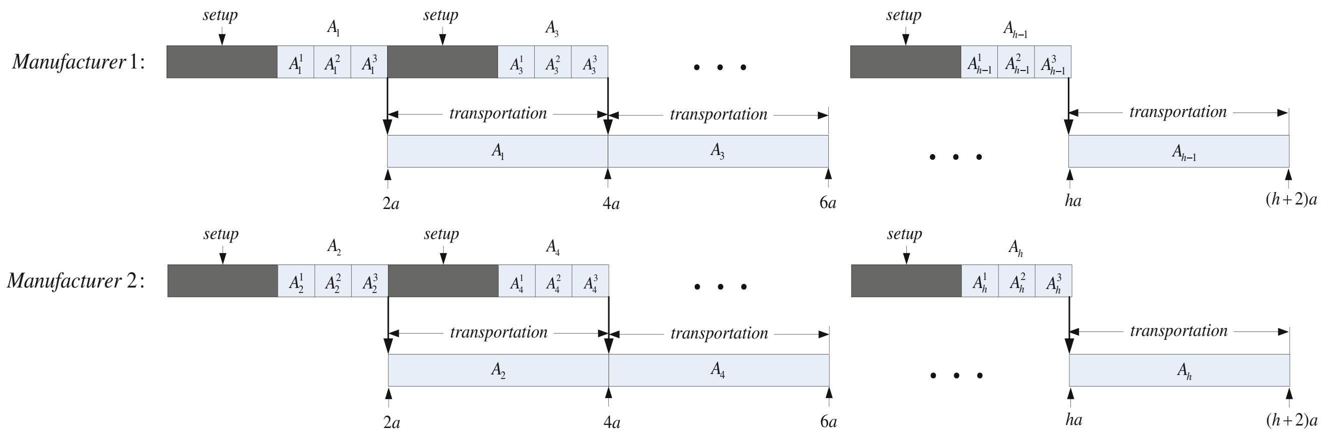


Fig. 2 The optimal schedule in Theorem 1

Lemma 2 *There exists an optimal schedule such that all jobs in each batch are processed in non-decreasing order of their arrival times on the machines of both manufacturers m_1 and m_2*

We omit the proof of the above two lemmas as it is simple. Then, the property about the setup time is given as follows:

Lemma 3 *In a schedule of the problem ψ, each batch is processed in non-decreasing order of ready time, which is equal to the arrival time of the last job in the batch. There exists a batch $b_k = \{J_i, \ldots, J_{i+x}, J_{i+x+1}, \ldots, J_{i+n_k-1}\}(x = 0, 1, \ldots, n_k-2)$ processed in the manufacturer m_j $(j = 1, 2)$, where $n_k \geq 2$ and $r_i \geq C_{1(k-1)j}$. If $r_{i+n_k-1} > r_{i+x} + s$, then the solution can be improved.*

Proof. The completion time of processing the batch b_k is $C_{1kj} = r_{i+n_k-1} + s + (p_i + \ldots + p_{i+n_k-1})$. Then, we can divide the batch b_k into two batches b_k^1 and b_k^2, i.e., $b_k^1 = \{J_i, \ldots, J_{i+x}\}$ and $b_k^2 = \{J_{i+x+1}, \ldots, J_{i+n_k-1}\}$. The starting and completion time of processing the batch b_k^1 are denoted as S_{1kj}^1 and C_{1kj}^1, respectively. Then, $C_{1kj}^1 = S_{1kj}^1 + s + (p_i +, \ldots, + p_{i+x}) = r_{i+x} + s + (p_i +, \ldots, + p_{i+x})$. Similarly, the starting and completion time of processing the batch b_k^2 are denoted as S_{1kj}^2 and C_{1kj}^2, respectively. There are three cases as follows:

(a) $C_{1kj}^1 \leq r_{i+n_k-1}$. Then, $C_{1kj}^2 = r_{i+n_k-1} + s + \left(p_{i+x+1} + \ldots + p_{i+n_k-1}\right)$. It is easy to see that $C_{1kj}^2 < C_{1kj}$.

(b) $r_{i+x} + s < r_{i+n_k-1} < C_{1kj}^1$. We can get that $C_{1kj}^2 = C_{1kj}^1 + s + \left(p_{i+x+1} + \ldots + p_{i+n_k-1}\right) = r_{i+x} + s + \left(p_i +, \ldots, + p_{i+x}\right) + s + \left(p_{i+x+1} + \ldots + p_{i+n_k-1}\right)$. It can be deduced that $C_{1kj}^2 - C_{1kj} = r_{i+x} + s - r_{i+n_k-1} < 0$. Thus, $C_{1kj}^2 < C_{1kj}$.

(c) $r_{i+n_k-1} \leq r_{i+x} + s$. $C_{1kj}^2 = C_{1kj}^1 + s + \left(p_{i+x+1} + \ldots + p_{i+n_k-1}\right) = r_{i+x} + s + (p_i +, \ldots, + p_{i+x}) + t + \left(p_{i+x+1} + \ldots + p_{i+n_k-1}\right)$. Similarly, we can infer that $C_{1kj}^2 \geq C_{1kj}$.

Based on cases (a), (b), and (c), if $r_{i+n_k-1} > r_{x+i} + s$, then the solution can be improved after dividing the original batch into two batches. Thus, the proof is completed.

Lemma 4 *The maximum completion time of all jobs on the machine of the manufacturer j is*

$$C^j \geq \sum_{k=1}^{h_j} P_k + s \cdot h_j + \min_{i=1,\ldots,n} \{r_i\} + \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right) (j = 1, 2)$$

Lemma 5 The sum of maximum completion time of all jobs on the machines of both manufacturers is

$$\sum_{j=1}^{2} C^j \geq \sum_{i=1}^{n} p_i + s \cdot h + 2 \min_{i=1,\ldots,n} \{r_i\} + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right) (j = 1, 2)$$

Proof. Based on Lemma 4, we obtain that $\sum_{j=1}^{2} C^j \geq \sum_{j=1}^{2} \sum_{k=1}^{h_j} P_k + \sum_{j=1}^{2} s \cdot h_j + \sum_{j=1}^{2} \min_{i=1,\ldots,n} \{r_i\} + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right)$. $\sum_{i=1}^{n} p_i + s \cdot h + 2 \min_{i=1,\ldots,n} \{r_i\} + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right)$. Then, the proof is completed.

5.2 Lower bound

Theorem 2. *The lower bound for the problem ψ is*

$$LB = \frac{\sum_{i=1}^{n} p_i + s \cdot \left\lceil \sum_{i=1}^{n} s_i / c \right\rceil + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right)}{2} + \min_{i=1,\ldots,n} \{r_i\} + T.$$

Proof. We assume that there exists a solution value $C'_{\max}$, and $C'_{\max} < LB$, i.e., $C'_{\max} <$

$$LB = \frac{\sum_{i=1}^{n} p_i + s \cdot \left\lceil \sum_{i=1}^{n} s_i / c \right\rceil + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right)}{2} + \min_{i=1,\ldots,n} \{r_i\} + T.$$

Then, $\max_{j=1,2} \left\{C^j + T\right\} < \frac{\sum_{i=1}^{n} p_i + s \cdot \left\lceil \sum_{i=1}^{n} s_i / c \right\rceil + \sum_{j=1}^{2} \sum_{l=1}^{m_j} \left(e_{jl} - d_{jl}\right)}{2} + \min_{i=1,\ldots,n} \{r_i\} + T.$

Next, $\max_{j=1,2}\left\{C^j\right\}+T<\frac{\sum_{i=1}^{n}p_i+s\cdot\left\lceil\sum_{i=1}^{n}s_i/c\right\rceil+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)}{2}+\min_{i=1,\ldots,n}\{r_i\}+T$.

It is obtained that $\max_{j=1,2}\left\{C^j\right\}<\frac{\sum_{i=1}^{n}p_i+s\cdot\left\lceil\sum_{i=1}^{n}s_i/c\right\rceil+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)}{2}+\min_{i=1,\ldots,n}\{r_i\}$.

Furthermore, $\sum_{j=1}^{2}C^j\leq 2\times\max_{j=1,2}\left\{C^j\right\}<2\times\left(\frac{\sum_{i=1}^{n}p_i+s\cdot\left\lceil\sum_{i=1}^{n}s_i/c\right\rceil+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)}{2}+\min_{i=1,\ldots,n}\{r_i\}\right)$.

Thus, $\sum_{j=1}^{2}C^j<\sum_{i=1}^{n}p_i+s\cdot\left\lceil\sum_{i=1}^{n}s_i/c\right\rceil+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)+2\min_{i=1,\ldots,n}\{r_i\}$.

Because $h\geq\left\lceil\sum_{i=1}^{n}s_i/c\right\rceil$, it can be deduced that $\sum_{j=1}^{2}C^j<\sum_{i=1}^{n}p_i+s\cdot h+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)+2\min_{i=1,\ldots,n}\{r_i\}$, which obviously contradicts with Lemma 5. Thus, the proof is completed.

6 A new heuristic algorithm *H*

In this section, some scheduling rules are first proposed, and based on these rules, a heuristic algorithm H is designed for solving this problem. Then, the flow chart of the proposed algorithm is presented, and the worst case performance of this algorithm in a special situation is also analyzed.

The notations of the proposed algorithm are defined as Table 1.

Before the heuristic algorithm is developed, we propose some useful rules for designing it, which are based on the situation whether the earliest available time of two machines to process a new batch is more than the maximum arrival time of all jobs or not.

Situation 1 $r_{\max}>ET$

We first give the rule of sequencing the jobs in this situation.

Rule 1. The jobs are sequenced in non-decreasing order of arrival time. If the arrival time of any two jobs is equal, then they are sequenced in non-increasing order of processing time.

During the period when no machines break down, considering a batch processed on the machine of the manufacturer j, if $r_i<E^j$ and $s_i+ts_j\leq c$, then we add the job J_i into this batch. Otherwise, based on Lemma 3, if $s_i+ts_j>c$ or $r_i-r_{i-1}\geq s$, then the job J_i is not added into this batch. We have the following rule.

Rule 2. Considering a batch to be processed on the machine of the manufacturer j, if the condition that $s_i+ts_j>c$ or the conditions that $r_i>E^j$ and $r_i-r_{i-1}\geq s$ are satisfied, then the job J_i is not added into this batch. Otherwise, we add the job J_i into this batch.

Considering the situation when the machine breakdown happens, if both machines break down simultaneously during a certain period, then the additional idle time should be inserted in this period and based on this idea, we add the jobs into the batch as many as possible when the updated available time of this batch to be processed is no more than the available time of the machine after breakdown. The corresponding rule is as follows:

Rule 3. Considering the situation when both machines breakdown and a batch to be processed on the machine of the manufacturer j, if $s_i+ts_j\leq c$ and $r_i<E^j$, then the job J_i is added into this batch.

Subsequently, we consider the situation when one machine breaks down and the other machine is available during a certain period; if the available time for the machine after breakdown is no more than that of the other machine without breakdown, then the jobs are added into the temporary batch to be processed on the machine with breakdown as many as possible when $r_i<E^j$ and $s_i+ts_j\leq c$. We have the following rule.

Rule 4. Considering the situation when one machine breaks down and the other machine is available, if $ET=E^j$, $r_i<E^j$, and $s_i+ts_j\leq c$, where the machine of the manufacturer j breaks down, then the job J_i is added into this batch.

Situation 2 $r_{max}\leq ET$.

This situation is equivalent to the situation that all jobs are available. The rule of sequencing the jobs in this situation is given as follows:

Rule 5. The jobs are sequenced in non-decreasing order of processing time. If the processing time of any two jobs is equal, then they are sequenced in non-increasing order of job size.

If both machines are available, then we add as many jobs as possible into each batch. The corresponding rule is as follows:

Rule 6. Considering a batch to be processed on the machine of the manufacturer j, if $s_i+ts_j\leq c$, then the job J_i is added into the batch.

If either machine breaks down, then we first update the available time of the machine to process the batch, and afterwards, Rule 6 is used.

Based on these rules, we develop a two-phase heuristic algorithm for the studied problem, and the details of this algorithm are as follows:

Phase 1 Scheduling jobs when $r_{max} > ET$.

In the following, we first present the procedure of initialization, and then we present the procedure of arranging the jobs into the batches when both machines are available, either machine breaks down, or both machines break down, respectively.

Procedure 1. Initialization

```
1  Initialize the corresponding parameters of machines and jobs;
2  Sequence the jobs according to Rule 1;
3  Initialize the first empty batch $b_1$, and place the first job $J_1$ into $b_1$. Arrange $b_1$ on the machine
   of the manufacturer 1. Update the statuses of $b_1$ and the machine of the manufacturer 1:
   $P^1 = p_1, ts_1 = s_1, r^1 = r_1, S^1 = s + r_1$, and $E^1 = S^1 + P^1$ .
```

Procedure 2. Scheduling a job when both machines are available in phase 1

```
1  **If** $E^1 < E^2$ **then**
2      Judge whether this job is placed into the current batch on the machine of the manufacturer
       1 according to Rule 2;
3      **If** this job is not placed into the batch **then**
4          Judge whether this job is placed into the current batch on the machine of the
           manufacturer 2 according to Rule 2;
5          **If** this job is not placed into the batch **then**
6              Create a new batch on the machine of the manufacturer 1 and place this job into this
               new empty batch;
7          **End**
8      **End**
9  **Else**
10     Judge whether this job is placed into the current batch on the machine of the manufacturer
       2 according to Rule 2;
11     **If** this job is not placed into the batch **then**
12         Judge whether this job is placed into the current batch on the machine of the
           manufacturer 1 according to Rule 2;
13         **If** this job is not placed **then**
14             Create a new batch on the machine of the manufacturer 2 and place this job into this
               new empty batch;
16         **End**
17     **End**
18 **End**
19 Update all statuses of current jobs, batches, and machines.
```

Procedure 3. Scheduling a job when both machines break down in phase 1

```
1  If $E^1 < E^2$ then
2      Judge whether this job is placed into the current batch on the machine of the manufacturer 1 according to Rule 3;
3      If this job is not placed into the batch then
4          Create a new batch on the machine of the manufacturer 1 and place this job into this new empty batch;
5      End
6  Else
7      Judge whether this job is placed into the current batch on the machine of the manufacturer 2 according to Rule 3;
8      If this job is not placed into the batch then
9          Create a new batch on the machine of the manufacturer 2 and place this job into this new empty batch;
10     End
11 End
12 Update all statuses of current jobs, batches, and machines.
```

Procedure 4. Scheduling a job when either machine breaks down in phase 1

```
1  If $E^1 < E^2$ and machine 1 breaks down then
2      Judge whether this job is placed into the current batch on the machine of the manufacturer 1 according to Rule 4;
3      If this job is not placed into the batch then
4          Create a new batch on the machine of the manufacturer 1 and place this job into this new empty batch;
5      End
6  Else if $E^1 > E^2$ and the machine of the manufacturer 2 breaks down then
7      Judge whether this job is placed into the current batch on the machine of the manufacturer 2 according to Rule 4;
8      If this job is not placed into the batch then
9          Create a new batch on the machine of the manufacturer 2 and place this job into this new empty batch;
10     End
11 Else
12     Do the judgment as Procedure 2;
13 End
14 Update all statuses of current jobs, batches, and machines.
```

Phase 2 Scheduling jobs when $r_{max} \leq ET$

In this phase, the procedures of updating jobs' sequence and arranging the jobs into the batches are presented, respectively.

Procedure 5. Updating jobs' sequence in phase 2

```
1  Update all states of current jobs, batches, and machines.
2  Sequence the jobs according to Rule 5;
```

Procedure 6. Scheduling a job in phase 2

```
1  If $E^1 < E^2$ then
2    Judge whether this job is placed into the current batch on the machine of the manufacturer 1 according to Rule 6;
3    If this job is not placed into the batch then
4      Judge whether this job is placed into the current batch on the machine of the manufacturer 2 according to Rule 6;
5      If this job is not placed into the batch then
6        Create a new batch on the machine of the manufacturer 1 and place this job into this new empty batch;
7      End
8    End
9  Else
10   Judge whether this job is placed into the current batch on the machine of the manufacturer 2 according to Rule 6;
11   If this job is not placed into the batch then
12     Judge whether this job is placed into the current batch on the machine of the manufacturer 1 according to Rule 6;
13     If this job is not placed into the batch then
14       Create a new batch on the machine of the manufacturer 2 and place this job into this new empty batch;
16     End
17   End
18 End
19 Update all states of current jobs, batches, and machines.
```

Then, based on these procedures, the overall heuristic algorithm is proposed as follows:

Heuristic algorithm *H*

```
1   For $i = 1$ to $n$ do
2     If $r_{max} > ET$ then
3       Execute procedure 1;
4       If both machines are available then
5         Execute procedure 2;
6       Else if both machines break down then
7         Execute procedure 3;
8       Else
9         Execute procedure 4;
10      End
11    End
12  Else
13    Execute procedure 5;
14    Execute procedure 6;
15  End
16  Update all states of current jobs, batches, and machines.
17  End for
```

The whole flow chart of this algorithm is shown in Fig. 3.

Theorem 3. *The proposed heuristic algorithm H can solve the studied problem in $O(n^2 logn)$ time, and its worst case performance ratio is*

$$3+\frac{ns}{\sum_{i=1}^{n}P_i+\left[\sum_{i=1}^{n}Si/c\right]s+\sum_{j=1}^{2}\sum_{l=1}^{m_j}\left(e_{jl}-d_{jl}\right)}$$

when there is no situation of machine breakdown since the time . $r_{\max}$

Proof. In this proposed algorithm, from step 2 to step 11, the time complexity is $O(n\log n)$. The time complexity of steps 13 and 14 is also $O(n\log n)$, the total execution time of step 1 is n, and the time complexity of other steps is $O(1)$. Thus, the time complexity of this algorithm is $O(n^2 logn)$.

For this problem, let $C_{\max}(H)$ and $C^*_{\max}$ denote the makespan generated by heuristic algorithm H and the optimal makespan. For the leftover jobs since the time $r_{\max}$, let T_1 and T_2 denote the sum of processing times of all jobs assigned to the machines of the manufacturers 1 and 2 based on heuristic algorithm H, let n^1 and n^2 denote the number of corresponding jobs scheduled on the machines of the manufacturers 1 and 2, let $C_{r_{\max}}$ and $C^*_{r_{\max}}$ denote the makespan generated by heuristic algorithm H and the optimal makespan, and let C^*_1 and C^*_2 denote the optimal makespan on the machines of the manufacturers 1 and 2. Then, we have

$$C_{r_{\max}}\le T_1+n^1s+T_2+n^2s,$$

and

$$T_1+T_2+\left\lceil\sum_{i=1}^{n_1+n_2}s_i/c\right\rceil s<C^*_1+C^*_2\le 2C^*_{r_{\max.}}$$

This implies that

$$\begin{aligned}C_{r_{\max}}+T_1+T_2+\left\lceil\sum_{i=1}^{n_1+n_2}s_i/c\right\rceil s\\ <T_1+n^1s+T_2+n^2s+2C^*_{r_{\max}}\\ <T_1+T_2+ns+2C^*_{r_{\max.}}\end{aligned}$$

Table 1 Notations of the proposed heuristic algorithm

Notation	Description
P^j	The processing time of the temporary batch processed on the machine of the manufacturer $j(j = 1,2)$
ts_j	The total size of the jobs in the temporary batch processed on the machine of the manufacturer $j(j = 1,2)$
r^j	The arrival time of the last job in the temporary batch processed on the machine of the manufacturer $j(j = 1,2)$
S^j	The earliest starting time of the temporary batch processed on the machine of the manufacturer $j(j = 1,2)$
E^j	The earliest completion time of processing the jobs on the machine of the manufacturer $j(j = 1,2)$
ET	The earliest available time of two machines to process a new batch
$r_{\max}$	The maximum arrival time of all jobs

So, we obtain

$$C_{r_{\max}} < 2C^*_{r_{\max}} + ns,$$

We have

$$C_{\max}(H) = r_{\max} + C_{r_{\max}} + T,$$

$$C^*_{r_{\max}} < C^*_{\max},$$

and

$$r_{\max} + T < C^*_{\max.}$$

Then,

$$C_{\max}(H) = r_{\max} + C_{r_{\max}} + T < C^*_{\max} + 2C^*_{\max} + ns.$$

Thus,

$$\frac{C_{\max}(H)}{C^*_{\max}} = 3 + \frac{ns}{C^*_{\max}} < 3 + \frac{ns}{LB}.$$

Based on the derived lower bound in Section 5.2, it can be derived that

$$\frac{C_{\max}(H)}{C^*_{\max}} < 3 + \frac{ns}{\sum_{i=1}^{n} p_i + \left\lceil \sum_{i=1}^{n} s_i/c \right\rceil s + \sum_{j=1}^{2} \sum_{l=1}^{m_j} (e_{jl} - d_{jl})}.$$

The proof is completed.

7 Computational experiments

In order to evaluate the performance of the proposed heuristic algorithm for the serial-batching scheduling problem, computational experiments were conducted, compared with some common dispatching rules [39–41] combining with batch first-fit heuristic algorithm [43,44]. These dispatching rules are as follows:

(1) First in first out dispatching rule (FIFO).
(2) The shortest processing time dispatching rule (SPT).
(3) The largest processing time dispatching rule (LPT).

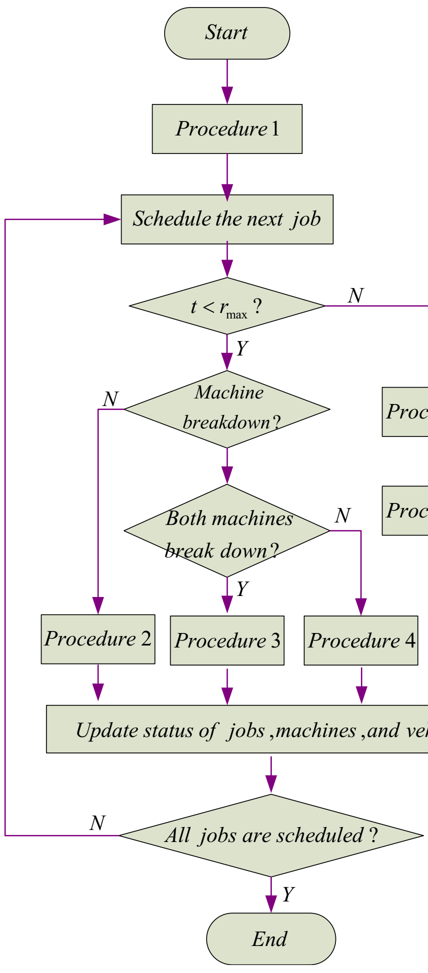


Fig. 3 Flow chart of the proposed heuristic algorithm *H*

Table 2 Parameters setting

Parameter	Description	Value
n	Number of the jobs	50, 100, 200, 400, 600, 800, 1000
c	Capacity of the batching machines and the vehicles	$U[20, 30]$
s_i	Job size	$U[2, 15]$
p_i	Job processing time on the manufacturers' serial-batching machines	$U[2, 22]$
r_i	Job arrival time	$U[1, 0.2{*}n\overline{p}]$
s	Setup time on the manufacturers' serial-batching machines	$U[5, 8]$
T	Transportation time between the manufacturers and customer	$U[20, 40]$
Ag	Machine breakdown level	0.05, 0.1
$MTTR$	Mean time to repair the machine	12, 36

All approaches were coded in PowerBuilder 9.0 language, and their code was run on a Pentium(R)-4 and 300 MHz PC with 2GB of RAM

In addition, the proposed heuristic algorithm was also compared with the original schedule, which is resumed to execute the rest task with no adjustment after the machine is repaired.

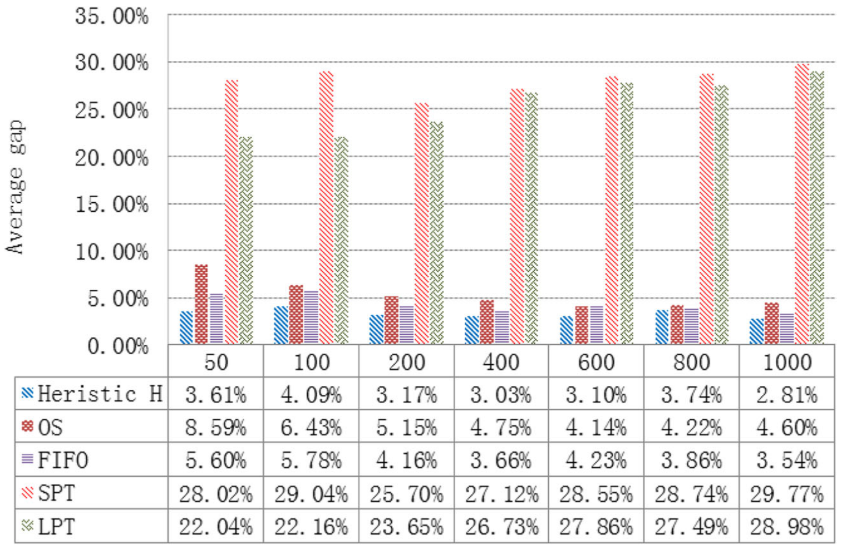
The test problems were randomly generated based on the real aluminum production. Job processing time on the manufacturers' serial-batching machines is generated from the discrete uniform distribution $U[2, 22]$; thus, the mean processing time is $\overline{p} = 12$. Following Holthaus [39], Dominic et al. [40], and Zandieh and Adibi [41], the breakdown level Ag and the mean time to repair $MTTR$ are set to $\{0.05,\ 0.1\}$ and $\{\overline{p},\ 3\overline{p}\}$ for evaluating the effect of the breakdown level. Following Tang and Liu [45], the job arrival time is generated from the discrete uniform distribution $U[1,\ 0.2{*}n\overline{p}]$. The experimental parameters are set as Table 2.

In order to evaluate the performance of the proposed heuristic algorithm H, its solutions were compared with those of the above three dispatching rules and original schedule, and original schedule is denoted as OS for simplicity. The comparison was made by measuring the relative gap between the makespan reported by each approach and the lower bound derived in Section 5.2. The relative gap with LB was measured on the test problem instances, and it is defined as $gap = (C_{\max} - LB) \times 100\ \%/LB$.

We designed a factorial experiment to determine the impact of two factors on the performance of each approach. One factor is machine breakdown level (i.e., Ag) that was tested for $Ag = 0.05$ and $Ag = 0.1$, and the other is mean time to repair the machine (i.e., $MTTR$) that was tested for $MTTR = 12$ and $MTTR = 36$. The combination (Ag, $MTTR$) was replicated ten times for each treatment.

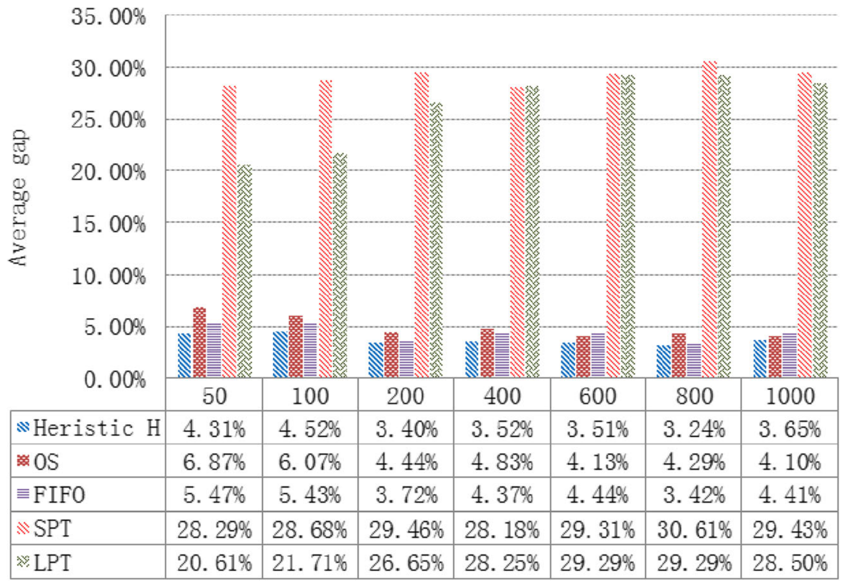
(1)Analysis of results for $Ag = 0.05$

Figure 4 reports the problem size's effect on the average gap of the problems when $Ag = 0.05$ and $MTTR = 12$. Heuristic algorithm H performed better than the other four approaches in this case, and its average gaps range from approximately 3.10 to 4.09 %. For all instance, the approach



	50	100	200	400	600	800	1000
▨ Heristic H	3.61%	4.09%	3.17%	3.03%	3.10%	3.74%	2.81%
▨ OS	8.59%	6.43%	5.15%	4.75%	4.14%	4.22%	4.60%
▤ FIFO	5.60%	5.78%	4.16%	3.66%	4.23%	3.86%	3.54%
▨ SPT	28.02%	29.04%	25.70%	27.12%	28.55%	28.74%	29.77%
▨ LPT	22.04%	22.16%	23.65%	26.73%	27.86%	27.49%	28.98%

Fig. 4 Computational results for $Ag = 0.05$ and $MTTR = 12$



	50	100	200	400	600	800	1000
▨ Heristic H	4. 31%	4. 52%	3. 40%	3. 52%	3. 51%	3. 24%	3. 65%
▩ OS	6. 87%	6. 07%	4. 44%	4. 83%	4. 13%	4. 29%	4. 10%
≡ FIFO	5. 47%	5. 43%	3. 72%	4. 37%	4. 44%	3. 42%	4. 41%
▨ SPT	28. 29%	28. 68%	29. 46%	28. 18%	29. 31%	30. 61%	29. 43%
▩ LPT	20. 61%	21. 71%	26. 65%	28. 25%	29. 29%	29. 29%	28. 50%

Fig. 5 Computational results for *Ag* = 0.05 and *MTTR* = 36

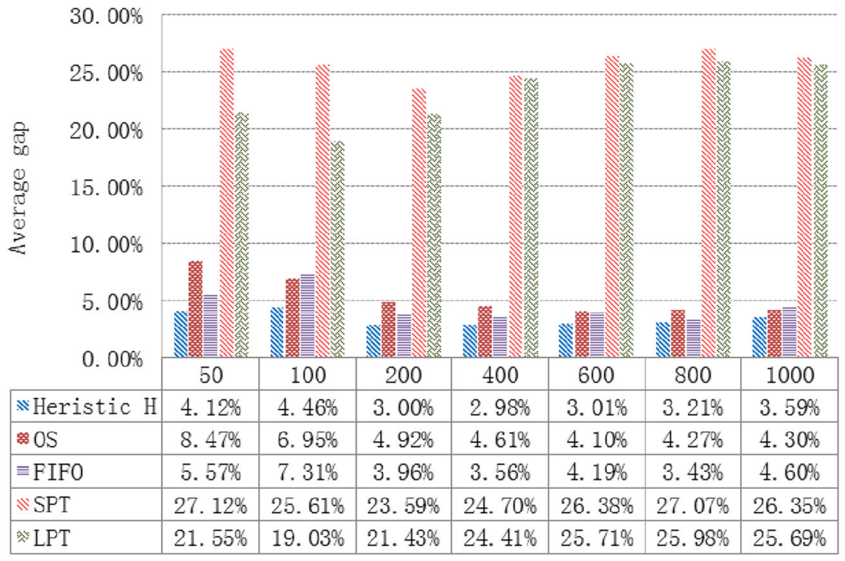
FIFO performed better than the approaches OS, SPT, and LPT except when *n* = 600, and the approach OS performed better than the approaches FIFO, SPT, and LPT when *n* = 600. In addition, the approach OS performed better than the approaches SPT and LPT for all instances.

In Fig. 5, we compare similar effect for the problems when *Ag* = 0.05 and *MTTR* = 36. The average gaps of heuristic algorithm *H* range from approximately 3.24 to 4.31 %. Heuristic algorithm *H* can also obtain better solutions than the other four approaches. The approach FIFO performed better than the approaches OS, SPT, and LPT when *n* = 50, 100, 200, 400, 800, and the approach OS performed better than the approaches FIFO, SPT, and LPT when *n* = 600, 1000.

(1)Analysis of results for *Ag* = 0.1

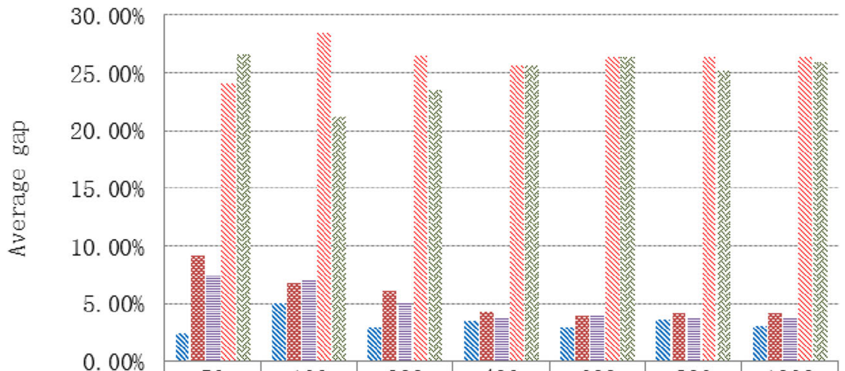
Figure 6 shows the performance comparison among these five approaches in terms of average gap when *Ag* = 0.1 and *MTTR* = 12. Compared with heuristic algorithm *H*, the approaches OS, FIFO, SPT, and LPT performed poorly for all instances, and the average gaps of heuristic algorithm *H* range from approximately 2.98 to 4.46 %. The approach FIFO performed better than the approaches OS, SPT, and LPT when *n* = 50, 200, 400, 800, and the approach OS performed better than the approaches FIFO, SPT, and LPT for other instances.

Figure 7 compares the different approaches on the average gap of the problems when *Ag* = 0.1 and *MTTR* = 36. It can be seen that heuristic algorithm *H* outperformed the other four approaches in this case, and its average gaps range from approximately 2.44 to 5.03 %. The approach



	50	100	200	400	600	800	1000
▨ Heristic H	4. 12%	4. 46%	3. 00%	2. 98%	3. 01%	3. 21%	3. 59%
▩ OS	8. 47%	6. 95%	4. 92%	4. 61%	4. 10%	4. 27%	4. 30%
≡ FIFO	5. 57%	7. 31%	3. 96%	3. 56%	4. 19%	3. 43%	4. 60%
▨ SPT	27. 12%	25. 61%	23. 59%	24. 70%	26. 38%	27. 07%	26. 35%
▩ LPT	21. 55%	19. 03%	21. 43%	24. 41%	25. 71%	25. 98%	25. 69%

Fig. 6 Computational results for *Ag* = 0.1 and *MTTR* = 12



	50	100	200	400	600	800	1000
▧ Heristic H	2. 44%	5. 03%	2. 93%	3. 54%	3. 01%	3. 65%	3. 07%
▩ OS	9. 18%	6. 75%	6. 10%	4. 33%	4. 00%	4. 17%	4. 19%
▤ FIFO	7. 55%	7. 12%	5. 13%	3. 89%	4. 08%	3. 82%	3. 87%
▨ SPT	24. 04%	28. 47%	26. 54%	25. 60%	26. 40%	26. 38%	26. 36%
▩ LPT	26. 60%	21. 23%	23. 55%	25. 66%	26. 38%	25. 17%	25. 93%

Fig. 7 Computational results for $Ag = 0.1$ and $MTTR = 36$

OS performed better than the approaches FIFO, SPT, and LPT when $n = 100$, 600, and the approach FIFO performed better than the approaches OS, SPT, and LPT when $n = 50$, 200, 400, 800, 1000.

It can be concluded from Figs. 4–7 that heuristic H outperformed the other four approaches for all instances, and its average gaps are below 5.1 %. Also, it is worthwhile to note that heuristic algorithm H can obtain solutions within 6 s for each problem instance, even for the instance up to 1000 jobs. This indicates that heuristic algorithm H can obtain feasible solutions within a reasonable CPU time.

8 Conclusions and future work

In this study, we investigate a serial-batching scheduling problem in which a set of jobs is processed on two manufacturers' serial-batching machines and then delivered in batches to a customer, considering machine breakdown, dynamic job arrival, and setup time. For the objective of minimizing the makespan, we prove that the problem is strongly NP-hard and provide its lower bound. The structural properties of the problem are carefully investigated through a number of proven propositions. Different rules are designed for two phases of the problem, respectively. Furthermore, based on these properties and rules, we have developed an effective heuristic algorithm to solve the problem and analyzed its worst case performance in a special situation. To evaluate the effectiveness of the proposed heuristic algorithm H, we conducted experiments with various randomly generated test problems. The experimental results suggest that the proposed heuristic algorithm H is more efficient than the other four approaches and it can handle the large-size problems in a reasonable time.

Our future research will continue following three directions: Firstly, other dynamic manufacturing conditions can be considered to extend the research and applications. Secondly, we need to combine other objective functions, such as minimizing the sum of completion time and minimizing maximum lateness. Last but not least, intelligent algorithms can be developed to solve the problems more efficiently and effectively.

Acknowledgments This work is supported by the National Natural Science Foundation of China (Nos. 71231004, 71501058, 71131002, 71571058, 71521001, 71471052, 71531008), the Humanities and Social Sciences Foundation of the Chinese Ministry of Education (No. 15YJC630097), Anhui Province Natural Science Foundation (No. 1608085QG167), and the Fundamental Research Funds for the Central Universities (Nos. JZ2015HGBZ0116, JZ2015HGBZ0117, JZ2015HGQC0187). Panos M. Pardalos is partially supported by the project of "Distinguished International Professor by the Chinese Ministry of Education" (MS2014HFGY026).

References

1. Hasan SMK, Sarker R, Essam D (2011) Genetic algorithm for job-shop scheduling with machine unavailability and breakdowns. Int J Prod Res 49(16):4999–5015
2. Wang K, Choi SH (2012) A decomposition-based approach to flexible flow shop scheduling under machine breakdown. Int J Prod Res 50(1):215–234
3. Benmansour R, Allaoui H, Artiba A (2012) Stochastic single machine scheduling with random common due date. Int J Prod Res 50(13):3560–3571
4. Lee JY, Kim YD (2012) Minimizing the number of tardy jobs in a single-machine scheduling problem with periodic maintenance. Comput Oper Res 39(9):2196–2205
5. Mirabi M, Fatemi Ghomi SMT, Jolai F (2013) A two-stage hybrid flowshop scheduling problem in machine breakdown condition. J Intell Manuf 24(1):193–199

6. Xiong J, Xing LN, Chen YW (2013) Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. Int J Prod Econ 141(1):112–126
7. Zandieha M, Adibi MA (2010) Dynamic job shop scheduling using variable neighbourhood search. Int J Prod Res 48(8):2449–2458
8. Lee ZJ, Lin SW, Ying KC (2010) Scheduling jobs on dynamic parallel machines with sequence-dependent setup times. Int J Adv Manuf Technol 47(5–8):773–781
9. Chiang TC, Cheng HC, Fu LC (2010) A memetic algorithm for minimizing total weighted tardiness on parallel batch machines with incompatible job families and dynamic job arrival. Comput Oper Res 37(12):2257–2269
10. Yao S, Jiang Z, Li N (2012) A branch and bound algorithm for minimizing total completion time on a single batch machine with incompatible job families and dynamic arrivals. Comput Oper Res 39(5):939–951
11. Yao FS, Zhao M, Zhang H (2012) Two-stage hybrid flow shop scheduling with dynamic job arrivals. Comput Oper Res 39(7): 1701–1712
12. Lu MS, Romanowski R (2012) Multi-contextual ant colony optimization of intermediate dynamic job shop problems. Int J Adv Manuf Technol 60(5–8):667–681
13. Anglani A, Grieco A, Guerriero E, Musmanno R (2005) Robust scheduling of parallel machines with sequence-dependent set-up costs. Eur J Oper Res 161(3):704–720
14. Quadt D, Kuhn H (2005) Conceptual framework for lot-sizing and scheduling of flexible flow lines. Int J Prod Res 43(11):2291–2308
15. Ciavotta M, Detti P, Meloni C, Pranzo M (2008) A bi-objective coordination setup problem in a two-stage production system. Eur J Oper Res 189(3):734–745
16. Roshanaei V, Naderi B, Jolai F, Khalili M (2009) A variable neighborhood search for job shop scheduling with set-up times to minimize makespan. Futur Gener Comput Syst 25(6):654–661
17. Shahvari O, Salmasi N, Logendran R, Abbasi B (2012) An efficient tabu search algorithm for flexible flow shop sequence-dependent group scheduling problems. Int J Prod Res 50(15):4237–4254
18. Varmazyar M, Salmasi N (2012) Sequence-dependent flow shop scheduling problem minimising the number of tardy jobs. Int J Prod Res 50(20):5843–5858
19. Thurer M, Silva C, Stevenson M, Land M (2012) Improving the applicability of workload control (WLC): the influence of sequence-dependent set-up times on workload controlled job shops. Int J Prod Res 50(22):6419–6430
20. Nagano MS, Silva AA, Lorena LAN (2012) A new evolutionary clustering search for a no-wait flow shop problem with set-up times. Eng Appl Artif Intell 25(6):1114–1120
21. Chakaravarthy GV, Marimuthu S, Sait AN (2013) Performance evaluation of proposed Differential Evolution and Particle Swarm Optimization algorithms for scheduling m-machine flow shops with lot streaming. J Intell Manuf 24(1):175–191
22. Hall NG, Potts CN (2003) Supply chain scheduling: batching and delivery. Oper Res 51(4):566–584
23. Geunes J., and Pardalos P.M. (2003). Supply chain optimization. Kluwer Academic Publishers
24. Su CS, Panb JCH, Hsua TS (2009) A new heuristic algorithm for the machine scheduling problem with job delivery coordination. Theor Comput Sci 410(27–29):2581–2591
25. Delavar MR, Hajiaghaei-Keshteli M, Molla-Alizadeh-Zavardehi S (2010) Genetic algorithms for coordinated scheduling of production and air transportation. Expert Syst Appl 37(12):8255–8266
26. Bard JF, Nananukul N (2010) A branch-and-price algorithm for an integrated production and inventory routing problem. Comput Oper Res 37(12):2202–2217
27. Steinruecke M (2011) An approach to integrate production-transportation planning and scheduling in an aluminium supply chain network. Int J Prod Res 49(21):6559–6583
28. You PS, Hsieh YC (2012) A heuristic approach to a single stage assembly problem with transportation allocation. Appl Math Comput 218(22):11100–11111
29. Cakici E, Mason SJ, Kurz ME (2012) Multi-objective analysis of integrated an supply chain scheduling problem. Int J Prod Res 50(10):2624–2638
30. Ng CT, Cheng TCE, Yuan JJ, Liu ZH (2003) On the single machine serial batching scheduling problem to minimize total completion time with precedence constraints, release dates and identical processing times. Oper Res Lett 31(4):323–326
31. Yuan JJ, Lin YX, Cheng TCE, Ng CT (2007) Single machine serial-batching scheduling problem with a common batch size to minimize total weighted completion time. Int J Prod Econ 105(2):402–406
32. Pei J, Liu X, Pardalos PM, Fan W, Yang S, Wang L (2014) Application of an effective modified gravitational search algorithm for the coordinated scheduling problem in a two-stage supply chain. Int J Adv Manuf Technol 70(1–4):335–348
33. Pei J, Liu X, Pardalos PM, Fan W, Yang S (2015) Single machine serial-batching scheduling with independent setup time and deteriorating job processing times. Optim Lett 9(1):91–104
34. Pei J, Pardalos PM, Liu X, Fan W, Yang S (2015) Serial batching scheduling of deteriorating jobs in a two-stage supply chain to minimize the makespan. Eur J Oper Res 244(1):13–25
35. Tang L, Guan J, Hu G (2010) Steelmaking and refining coordinated scheduling problem with waiting time and transportation consideration. Comput Ind Eng 58(8):239–248
36. Gong H, Tang L (2011) Two-machine flowshop scheduling with intermediate transportation under job physical space consideration. Comput Oper Res 38(9):1267–1274
37. Tang L, Gong H, Liu J, Li F (2014) Bicriteria scheduling on a single batching machine with job transportation and deterioration considerations. Nav Res Logist 61(4):269–285
38. Xuan H, Tang L (2007) Scheduling a hybrid flowshop with batch production at the last stage. Comput Oper Res 34(9):2718–2733
39. Holthaus O (1999) Scheduling in job shops with machine breakdowns: an experimental study. Comput Ind Eng 36(1):137–162
40. Dominic PDD, Kaliyamoorthy S, Kumar M (2004) Efficient dispatching rules for dynamic job shop scheduling. Int J Adv Manuf Technol 24(1–2):70–75
41. Zandieh M, Adibi MA (2010) Dynamic job shop scheduling using variable neighbourhood search. Int J Prod Res 48(8):2449–2458
42. Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG (1979) Optimization and approximation in deterministic machine scheduling: a survey. Ann Discret Math 5:287–326
43. Chou FD, Chang PC, Wang HM (2006) A hybrid genetic algorithm to minimize makespan for the single batch machine dynamic scheduling problem. Int J Adv Manuf Technol 31(3–4):350–359
44. Mirsanei HS, Karimi B, Jolai F (2009) Flow shop scheduling with two batch processing machines and nonidentical job sizes. Int J Adv Manuf Technol 45(5–6):553–572
45. Tang L, Liu P (2009) Minimizing makespan in a two-machine flowshop scheduling with batching and release time. Math Comput Model 49(5–6):1071–1077