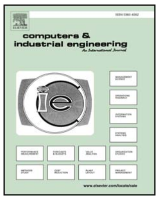
Computers & Industrial Engineering 115 (2018) 342–353

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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



A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops

Weiwei Cui[a,*], Zhiqiang Lu[b], Chen Li[c], Xiaole Han[b]

[a] School of Management, Shanghai University, Shanghai 200444, PR China
[b] School of Mechanical Engineering, Tongji University, Shanghai 201804, PR China
[c] School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, PR China

ARTICLE INFO

Keywords:
Flow-shop
Scheduling
Maintenance
Proactive

ABSTRACT

This paper deals with the integration of production scheduling and maintenance planning in order to optimize the bi-objective of quality robustness and solution robustness for flow shops with failure uncertainty. First, a proactive model is proposed to formulate the problem mathematically. Then, Monte Carlo sampling method is adopted to obtain the objective value for feasible solutions and a surrogate measure is proposed to approximate the objective function efficiently. Based on the sampling method and surrogate measure, a two-loop algorithm is devised to optimize the sequence of jobs, positions of preventive maintenances and idle times simultaneously. Computational results indicate that solution robustness and stability of quality robustness can be significantly improved using our algorithm compared with the solutions obtained by the traditional way.

1. Introduction

Although many researchers have discussed the production scheduling problems, classical scheduling articles assume that machines are always available for processing jobs at all times during the scheduling horizon. However, machines may be unavailable for different reasons in realistic practice, such as unexpected breakdowns or preventive maintenances (PMs). The production scheduling and maintenance activities are interrelate since both of them occupy the machine's capacity, where production depletes the machine's reliability and maintenance restores its reliability. Production scheduling and PM planning should be taken into the integrated optimization model into consideration to balance the utilization and availability of the resource (Wang & Liu, 2014).

There are two classes of research when maintenance part is introduced into the production scheduling problems. For the first class, researchers ignore the unexpected breakdowns and only focus on the capacity of machine; conversely, the second class considers the impact of unexpected breakdowns and performs corrective maintenance (CM) when breakdown occurs.

In the first class, researchers assume that PMs are performed periodically to keep machines in a good condition and a high reliability. Thus, the random breakdowns can be ignored, which results in the deterministic problems. And, the traditional regular objectives, e.g., the makespan, the total flow time, the tardiness, etc., are considered. Since the machine is unavailable to process jobs when PMs are performed, the impact of unavailability of machine on the production needs to be analyzed to search the optimal jobs' sequence.

In the second class, researchers assume that there are unexpected random breakdowns during the execution of scheduling plan. Considering the impact of uncertainty, this kind of problem belongs to the stochastic programming problem, which is completely different from the first class. Since the existence of random variables, the expectation of performance is usually considered in the related articles. Besides, the schedule robustness is also subscribed by many experienced schedulers when some uncertainties are considered. Herroelen and Leus (2005) divided schedule robustness into two groups: solution robustness and quality robustness. They defined solution robustness as the insensitivity of the activity start times to variations in the input data, and quality robustness as the insensitivity of schedule performance (such as project makespan or cost) with respect to disruptions.

In this paper, the coordination between production scheduling and entire maintenance policy is investigated to provide an integrated decision system for the operators in flow shops. To the best of our knowledge, this is the first endeavour to establish a proactive joint model for flow shops with stochastic failure uncertainty to integrate the production scheduling and maintenance policy, including the PMs and CMs, in order to optimize the bi-objective of quality robustness and solution robustness.

The main contributions of this paper are as follows:

- A flow shop scheduling problem subject to the degenerate machines

* Corresponding author.
E-mail address: cuiww67@163.com (W. Cui).

https://doi.org/10.1016/j.cie.2017.11.020
Received 8 February 2017; Received in revised form 16 May 2017; Accepted 21 November 2017
Available online 22 November 2017

with the consideration of robustness is studied.

- A proactive model and a two-loop algorithm are proposed to deal with the uncertainty caused by the unexpected disruptions.
- An efficient and effective surrogate measure based on the analysis of propagation of breakdown's impact is devised to approximate the objective function, which can reduce the computation time of algorithm dramatically.

2. Literature review

Since Johnson's remarkable paper on flow shop scheduling problems (FSPs) was published in 1954, FSPs have attracted a lot of attention from researchers and practitioners. Considering the problem's *NP-complete* nature, exact algorithms such as dynamic algorithm and branch & bound can only be used to solve small-sized problems. Thus, many researchers focus on the development of constructive heuristics or meta-heuristics. In recent years, different kinds of meta-heuristics have been developed to solve large-sized traditional problems, such as Deng and Gu (2012), Wang and Liu (2013), and Kheirandish, Tavakkoli-Moghaddam, and Karimi-Nasab (2015).

Deterministic scheduling problems with non-availability intervals have received considerable attention since the beginning of the 1990s. For the scheduling problems with unavailability constraints, two cases of consideration can be found in literature: (i) the intervals are fixed and known in advance, and (ii) the intervals are flexible and can be scheduled by the operator. For the first case, comprehensive reviews about different kinds of machine systems are provided by Sanlaville and Schmidt (1998), Schmidt (2000), and Ma, Chu, and Zuo (2010). For the research on flow shops, Lee (1997), Cheng and Wang (2000), Breit (2004, 2006), Hadda, Dridi, and Hajri-Gabouj (2010), and Hadda (2012) focused on the two-machine systems; Aggoune(2004), Aggoune and Portmann (2006), Choi, Lee, Leung, and Pinedo (2010), and Shoaardebili and Fattahi (2015) studied multi-machine flow shops. For the second case, no related review paper can be found in this area. The corresponding problem for flow shops can be found in Aggoune (2004), Kubzin and Strusevich (2005, 2006), Allaoui, Lamouri, Artiba, and Aghezzaf (2008), Vahedi-Nouri, Fattahi, and Ramezanian (2013), Vahedi-Nouri, Fattahi, Tavakkoli-Moghaddam, and Ramezanian (2014), Gara-Ali and Espinouse (2014), and Hadda (2015).

When the deterioration of machine is considered, some researchers only consider the expectation of regular objectives, e.g., $E(C_{max})$, which is the expectation of makespan. Cassady and Kutanoglu (2003), Sortrakul, Nachtmann, and Cassady (2005) and Wang and Liu (2014) investigated the integrated model that simultaneously determines production scheduling and PMs planning for optimizing the expectation of regular objectives. Ruiz, García-Díaz, and Maroto (2007), Jabbarizadeh, Zandieh, and Talebi (2009) and Naderi, Zandieh, and Aminnayeri (2011) implicitly considered three different preventive maintenance polices and proposed a simple criterion to schedule preventive maintenance operations to the production sequence.

When the unexpected disruption is considered, some researchers consider the system's robustness. Leon, Wu, and Storer (1994), O'Donovan, Uzsoy, and McKay (1999), Goren and Sabuncuoglu (2008) and Al-Hinai and ElMekkawy (2011) considered the robustness measure in the scheduling problems with random breakdowns. They stated that idle times in the initial schedule can improve the robustness of system. Briskorn, Leung, and Pinedo (2011) also considered the same idea and analyzed the allocation of idle times in a single machine environment. Rahmani, Heydari, Makui, and Zandieh (2013) considered the arrival of new unpredicted jobs and devised a two step procedure in which an initial solution is generated in the first step and a corresponding solution is obtained by an appropriate reactive method to deal with this unexpected event in the second step. Rahmani and Ramezanian (2016) developed a variable neighbourhood search (VNS) algorithm to solve the problem of a dynamic flexible flow shop environment considering the arrival of new unpredicted jobs. Rahmani and Heydari (2014) proposed an initial robust solution which is more insensitive against the fluctuations of uncertain processing times and then developed a reactive approach determining the best modified sequence after any unexpected disruption based on the classical objective and performance. The above articles did not consider the machine's deterioration effect and preventive maintenances (PMs). When the machine's deterioration effect is considered, Weibull probability distribution is commonly assumed to be the failure function of machine. Cui, Lu, and Pan (2014) and Lu, Cui, and Han (2015) assumed that the machine failure is governed by this kind of distribution and investigated the robustness problem for a single machine system to integrate the production scheduling and maintenance policy.

The remainder of this paper is organized as follows. In Section 3, we describe the problem in detail. In Section 4, methods of evaluating the feasible solutions are proposed. In Section 5, a two-loop algorithm is devised according to the division of decision variables. Computational experiments are then given in Section 6 to demonstrate the effectiveness of algorithm we proposed, followed by the conclusions and discussions in Section 7.

3. Problem description and modelling

The following notations are used to formulate the problem.

Indices:

i	index of jobs
j	index of machines
[k]	index of the position in each machine

Sets:

J	set of jobs; $J = \{J_1, J_2, \ldots, J_n\}$
M	set of machines; $M = \{M_1, M_2, \ldots, M_m\}$
O_i	set of operations of J_i; $O_i = \{O_{i1}, O_{i2}, \ldots, O_{ij}, \ldots, O_{im}\}$, O_{ij} is the operation of J_i in M_j

Parameters:

n	number of jobs
m	number of machines
p_{ij}	processing time of O_{ij}
t_{pj}	preventive maintenance time of machine M_j
t_{rj}	corrective maintenance time of machine M_j
β_j	shape parameter of failure function of machine M_j
θ_j	scale parameter of failure function of machine M_j

3.1. Problem assumptions and formulation

A set of jobs J is supposed to be processed on a set of machines M for minimizing the objective of makespan in flow shops. Every job J_i ($J_i \in J$) consists a sequence of m operations $\{O_{i1}, O_{i2}, \ldots, O_{im}\}$ and requires a fixed processing time p_{ij} on machine M_j. Each machine can only process one job and each job can only be processed by one machine at a time. All jobs are available at the beginning of scheduling horizon. The buffer capacity between any two machines is unlimited, i.e., no block exists. The jobs' sequences in different machines are the same, i.e., permutation flow shop is considered here.

Suppose the machines used to process jobs are subject to failure, and the time to failure for the machines are run-based and governed by Weibull probability distributions. $(t_{pj}, t_{rj}, \beta_j, \theta_j)$ is the set of parameters of the Weibull function for machine M_j. Furthermore, we assume $\beta_j > 1$, which means the machine degenerates over time. Since unexpected failures usually cause the instability of the system and quality loss of product, it is practical to perform PMs on machines throughout the production horizon to improve the machines' condition and reduce the risk of unexpected machines' failures. We assume that the PM action restores machine to the "as good as new" condition, i.e., the machine's age becomes zero after PM. Since the unexpected machine's failures cannot be completely eliminated by PMs, corrective maintenance should be taken once the machine fails. Minimal repair policy is adopted, i.e., machine is restored to an operating condition, but the machine's age is not changed. And, the job interrupted by machine failure can be resumed after repair without any additional time penalty.

We need to decide the jobs' sequence just like the traditional flow shop scheduling problem without the maintenance consideration. Meanwhile, we also need to decide the number of PMs and the start time of each PM in the horizon for each machine. It means that we can provide a whole solution including production scheduling and maintenance planning to guide the operators in shops, which can maximize the company profit by avoiding the conflict between production department and maintenance department.

In this paper, the bi-objective of quality robustness and solution robustness is considered. A predictive schedule σ^0 is generated at the beginning of the scheduling horizon. σ^0 is executed on the shop floor and revised using the rescheduling policy when breakdown occurs. At the end of scheduling horizon, we have an actual schedule σ^w. For the quality robustness, the performance measure of $z_1 = E_w(\max_{i,j} C_{ij}^w)$ is selected to minimize the expected makespan. C_{ij}^w is the realized completion time of O_{ij} in scenario w. For the solution robustness, the measure of $z_2 = E_w\left(\sum_{j=1}^m \sum_{i=1}^n (S_{ij}^w - S_{ij}^0)\right)$ is selected to minimize the total deviation between the jobs' start time from actual schedule and that of the initially planned schedule. S_{ij}^w is the realized start time of O_{ij} in scenario w, S_{ij}^0 is the initially planned start time. When considering two performance measures simultaneously, we minimize the objective function $z = \rho_1 z_1 + \rho_2 z_2$, where ρ_1 is a real number between 0 and 1 that represents the weight given to the quality robustness, and $\rho_1 + \rho_2 = 1$.

3.2. Modelling

Since the probability of breakdowns can be predicted for a given solution, the proactive-reactive method is adopted in this paper to handle this uncertain factor. First, an initial plan is made at the beginning of the horizon. A whole solution can be divided into three parts (**X**,**Y**,**ST**). The first one is the jobs' sequence list **X**; the second one is the PMs' positions matrix **Y**; the third one is the jobs' start times matrix **ST**. For instance, there are four jobs and two machines. Fig. 1a) shows the Gantt chart of an initial schedule for this problem, where the jobs' sequence list is represented as $\mathbf{X} = \{2,3,1,4\}$, the PMs' positions matrix is represented as $\mathbf{Y} = \{0,0,0,1;0,0,1,0\}$, and the start times matrix **ST** is represented as $\{S_{2,1}, S_{3,1}, S_{1,1}, S_{4,1}; S_{2,2}, S_{3,2}, S_{1,2}, S_{4,2}\}$. $S_{i,j}$ is the planned start time of O_{ij}. $Y_{ij} = 1$ means that one PM is performed immediately before the ith operation in machine M_j. And, PM is performed before the idle time.

In order to absorb the unpredictable right-shifting instability, we propose to proactively insert idle times into the schedule. As shown in Fig. 1a), there are some idle times which do not exist in the schedules if the unexpected breakdowns are ignored. It means that the planned start times of the jobs are not wholly decided by the jobs' sequence, and are severely affected by the idle times inserted. After the initial plan is made at the beginning, the jobs' sequence and the positions of PMs will not be changed throughout the scheduling horizon. However, the realized start times of operations will be changed. For example, if one breakdown happens during O_{31} as shown in Fig. 1b), then one CM is performed. The operations O_{11} and O_{32} are delayed. And, the start times of operations O_{41}, O_{12} and O_{42} are not delayed because of the existence of idle times. Meanwhile, the jobs' realized start times cannot be earlier than their planned start times. This rescheduling policy is quite reasonable in practice since the initial schedule serves as a basis for planning external activities such as tools change and material procurement. For example, if the realized jobs' sequence is different from the initially planned sequence, it will cause a lot of trouble for the material supply system, since the original material prepared needs to be moved out from the production line and new material needs to be moved in. And, one job cannot be started if the related materials are not ready.

The proactive joint model (JM) can be finally established as follows.

Decision variables:

$x_{i[k]}$	if job J_i is processed in the kth position in machines, $x_{i[k]} = 1$; otherwise, 0.
$y_{[k]j}$	if PM is performed before the kth job in machine j, $y_{[k]j} = 1$; otherwise, 0.
$S_{[k]j}^0$	the planned start time of $O_{[k]j}$.

Auxiliary variables:

$O_{[k]j}$	the operation of the kth job in machine j
$p_{[k]j}$	the processing time of $O_{[k]j}$
$a_{[k]j}$	the machine's age after $O_{[k]j}$
$b_{[k]j}$	the machine's age before $O_{[k]j}$
$S_{[k]j}^w$	the realized start time of $O_{[k]j}$ in scenario w.
$C_{[k]j}^w$	the realized finish time of $O_{[k]j}$ in scenario w.
$\xi_{[k]j}^w$	the realized number of breakdowns when processing $O_{[k]j}$

$$\min \rho_1 E_w(C_{[n]m}^w) + \rho_2 E_w\left(\sum_{j=1}^m \sum_{k=1}^n (S_{[k]j}^w - S_{[k]j}^0)\right)$$

$$S.\ t.\quad \sum_{i=1}^n x_{i[k]} = 1 \quad \forall\ k \tag{1}$$

$$\sum_{k=1}^n x_{i[k]} = 1 \quad \forall\ i \tag{2}$$

$$p_{[k]j} = \sum_{i=1}^n x_{i[k]} p_{ij} \quad \forall\ k, \forall\ j \tag{3}$$

$$S_{[k]j}^0 \geqslant S_{[k-1]j}^0 + p_{[k-1]j} + t_{pj} y_{[k]j} \quad \forall\ j, \forall\ k > 1 \tag{4}$$

$$S_{[k]j}^0 \geqslant S_{[k]j-1}^0 + p_{[k]j-1} \quad \forall\ k, \forall\ j > 1 \tag{5}$$

$$S_{[1]1}^0 = 0 \tag{6}$$

$$b_{[1]j} = 0 \quad \forall\ j \tag{7}$$

$$b_{[k]j} = a_{[k-1]j}(1 - y_{[k]j}) \quad \forall\ k > 1, \forall\ j \tag{8}$$

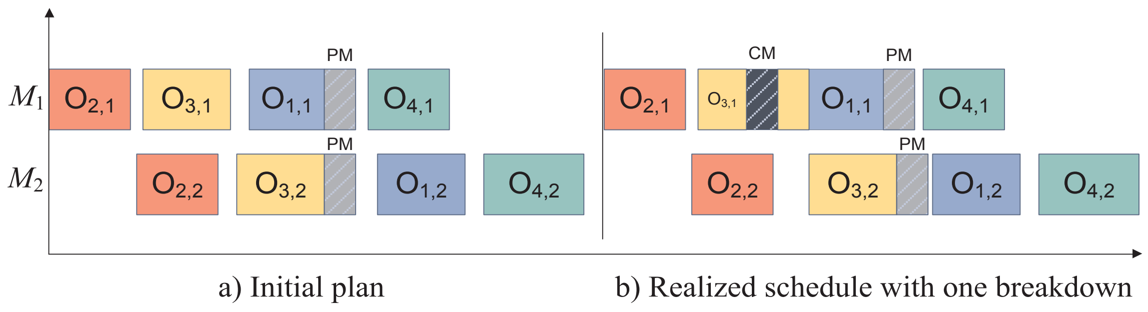


Fig. 1. Gantt chart of an example.

$$a_{[k]j} = b_{[k]j} + p_{[k]j} \quad \forall\, k ,\forall\, j \tag{9}$$

$$C^w_{[k]j} = S^w_{[k]j} + p_{[k]j} + \xi^w_{[k]j} t_{rj} \quad \forall\, k ,\forall\, j \tag{10}$$

$$S^w_{[k]j} = \max\{C^w_{[k-1]j} + t_{pj} y_{[k]j}, C^w_{[k]j-1}, S^0_{[k]j}\} \quad \forall\, k > 1 ,\forall\, j > 1 \tag{11}$$

$$S^w_{[k]1} = \max\{C^w_{[k-1]1} + t_{p1} y_{[k]1}, S^0_{[k]1}\} \quad \forall\, k > 1 \tag{12}$$

$$S^w_{[1]j} = \max\{C^w_{[1]j-1}, S^0_{[1]j}\} \quad \forall\, j > 1 \tag{13}$$

$$S^w_{[1]1} = 0 \tag{14}$$

$$x_{i[k]}, y_{[k]j} \text{ are } binaries \quad \forall\, i ,\forall\, j ,\forall\, k \tag{15}$$

$$p_{[k]j}, S^0_{[k]j}, a_{[k]j}, b_{[k]j}, C^w_{[k]j}, S^w_{[k]j} \geqslant 0 \quad \forall\, k ,\forall\, j \tag{16}$$

Constraints (1) and (2) ensure that each job can only be placed at one position in each machine and one position can only be occupied by one job. Constraint (3) specifies the processing time of $O_{[k]j}$. Constraint (4) ensures that the operation $O_{[k]j}$ starts no earlier than the sum of the finish time of its immediate predecessor job in M_j and the possible PM time. Constraint (5) ensures that the start time of one job in M_j is not earlier than its finish time in $M_{j\text{-}1}$. Constraint (6) means the first operation of the first job starts at time zero. Constraint (7) means the machine's age is 0 at the beginning of the horizon. Constraint (8) and (9) specify the machine's age immediately before and after operation $O_{[k]j}$, respectively. Constraint (10) establishes the link between the realized start and finish time of $O_{[k]j}$. Constraints (11)–(14) ensure that the actual start time of $O_{[k]j}$ is not earlier than its planned time, nor the sum of actual finish time of $O_{[k-1]j}$ and possible PM time, nor the actual finish time of $O_{[k]j-1}$.

4. Evaluation of the objective

4.1. Simulation based evaluation

For the objective function,

$$\begin{aligned} z &= \rho_1 E_w(C^w_{[n]m}) + \rho_2 E_w\left(\sum_{j=1}^{m}\sum_{k=1}^{n}(S^w_{[k]j} - S^0_{[k]j})\right) \\ &= \rho_1 E_w(S^w_{[n]m} + p_{[n]m} + \xi^w_{[n]m} t_{rm}) + \rho_2 E_w\left(\sum_{j=1}^{m}\sum_{k=1}^{n}(S^w_{[k]j} - S^0_{[k]j})\right) \\ &= \rho_1 (E_w(S^w_{[n]m}) + p_{[n]m} + t_{rm} E_w(\xi^w_{[n]m})) + \rho_2 \sum_{j=1}^{m}\sum_{k=1}^{n} E_w(S^w_{[k]j}) \\ &\quad -\rho_2 \sum_{j=1}^{m}\sum_{k=1}^{n} S^0_{[k]j} \end{aligned} \tag{17}$$

$p_{[n]m}$, $E_w(\xi^w_{[n]m})$ and $S^0_{[k]j}$ can be obtained in polynomial time for a given solution σ^0. However, the exact value of $E_w(S^w_{[k]j})$ cannot be obtained. Thus, the objective value cannot be obtained in polynomial time even for a given solution. From (11)–(14), we can find that $S^w_{[k]j}$ is depended on the value of $\xi^w_{[k]j}$.

According to the maintenance theory, $\xi^w_{[k]j}$ is governed by a Poisson probability distribution with $\lambda_{[k]j}$ and $\Pr(\xi^w_{[k]j} = \eta) = (\lambda_{[k]j})^{\eta} e^{-\lambda_{[k]j}}/\eta!\ \forall\, \eta \in [0,+\infty)$, where $\lambda_{[k]j} = (a_{[k]j}/\theta_j)^{\beta_j} - (b_{[k]j}/\theta_j)^{\beta_j}$. The mathematical formula of the expectation of realized start times cannot be derived, although we can get the expectation of $\xi^w_{[k]j}$. $E(\xi^w_{[k]j}) = \lambda_{[k]j} = (a_{[k]j}/\theta_j)^{\beta_j} - (b_{[k]j}/\theta_j)^{\beta_j}$, where the values $a_{[k]j}$ and $b_{[k]j}$ can be derived according to (7)–(9).

Bonfill, Espuna, and Puigjaner (2008), Vonder, Demeulemeester, and Herroelen (2008), Jahangirian, Eldabi, Naseer, Stergioulas, and Young (2010), etc., adopted the Monte Carlo simulation to handle this kind of difficulty. Next, we will introduce the procedure of Monte Carlo simulation for our problem.

Since the machine's age becomes zero after PM, the degeneration of machine restarts from the moment of performing PM. Each PM period should be handled separately. Let all the operations in a same PM period be a batch, i.e., $B_{lj}\ \forall\ 1 \leqslant l \leqslant L_j$ is the lth batch in M_j, where L_j is the number of batches in M_j. Let q_{lj} be the total processing time of operations in B_{lj}; and n_{lj} be the number of operations in B_{lj}. For example, number of batches in M_1 is 2 in Fig. 1. In B_{11}, there are 3 operations, i.e., $n_{11} = 3$. In B_{21}, there is only one operation.

The machine breakdown can be described as a model of a stochastic point process in a PM period. Since the machine failure is governed by a Weibull probability distribution, the process is a non-homogenous Poisson process during the PM period. The reliability of machine can be obtained by $R_j = e^{-(t_j/\theta_j)^{\beta_j}}$, where t_j is the machine's age. Hence, in each PM period, we can obtain the sampling method to get the τ^{th} breakdown time in machine M_j by $bt_{\tau j} = R_j^{-1}[\Pi^{\tau}_{\chi=1}(1-\varsigma_\chi)]$, where $R_j^{-1}(g)$ is the reverse function of $R_j(g)$ and ς_χ is a uniformly distributed random variable $\varsigma_\chi \sim U(0,1)$.

Denote the sample size as W and $z(\sigma^0)$ as the solution's objective value which is defined in the JM model in Section 3.

The detailed simulation procedure is shown as follows.

Procedure I

```
Step 1. Set $\{z_w\} = \{0,0,...,0\}(w = 1,2,...,W)$; get job list $X$, PMs'
  positions matrix Y and planned start times matrix ST from $\sigma^0$.
Step 2. Calculate $a_{[k]j}, b_{[k]j}\ \forall\ k,j$ based on (7)–(9), and calculate
  $q_{lj}, n_{lj}\ \forall\ l,j$.
Step 3. Set $w = 1$.
Step 4. For $j: =1 to m$, do
  4.1. Set $\{\xi_{[k]j}\ \forall\ k\} = \{0,0,...,0\}$; set $l = 1$; set $n_0 = 0$.
  4.2. Set $\tau = 1$.
  4.3. Generate random sample $\varsigma_\tau \sim U(0,1)$, and set
  $bt_{\tau j} = R_j^{-1}[\Pi^{\tau}_{\chi=1}(1-\varsigma_\chi)]$. If $bt_{\tau j} > q_{lj}$, goto 4.5.
  4.4. If $bt_{\tau j} < a_{[1+n_{l-1}]j}$, set $\xi^w_{[1+n_{l-1}]j} = \xi^w_{[1+n_{l-1}]j} + 1$; otherwise, if $\exists\ \delta$
  satisfies $a_{[\delta+n_{l-1}]j} < bt_{\tau j} < a_{[\delta+1+n_{l-1}]j}(\ \forall\ \delta \leqslant n_l-1)$, then set
  $\xi^w_{[\delta+1+n_{l-1}]j} = \xi^w_{[\delta+1+n_{l-1}]j} + 1$. Set $\tau = \tau + 1$, goto 4.3.
  4.5. $l = l + 1$. If $l \leqslant L_j$, goto 4.2.
Step 5. Calculate $S^w_{[k]j}\ \forall\ k ,\forall\ j$ based on (11)–(14), and calculate $z$
  value $z^w = \rho_1 C^w_{[n]m} + \rho_2 \sum^m_{j=1}\sum^n_{k=1}(S^w_{[k]j} - S^0_{[k]j})$.
Step 6. Set $w = w + 1$. If $w < W$, goto Step 4.
Step 7. Calculate $z(\sigma^0) = \sum^W_{w=1} z^w/W$.
```

4.2. Evaluation based on surrogate measure

One evaluation method is proposed in Section 4.1. However, the optimization method based on simulation is usually quite time consuming. In order to overcome this kind of problem, Goren and Sabuncuoglu (2008) proposed to use the surrogate measure to approximate the real performance of solutions. Therefore, we derive in the approximations for the objective in order to reduce the computation time.

4.2.1. Design of surrogate measure

Let $d^w_{[k]j}$ be the delay of the start time of $O_{[k]j}$ in scenario w. Then, the objective can be obtained by

$$\begin{aligned} z &= \rho_1 z_1 + \rho_2 z_2 \\ &= \rho_1 E_w(S^w_{[n]m} + p_{[n]m} + \xi^w_{[n]m} t_{rm}) + \rho_2 E_w\left(\sum_{j=1}^{m}\sum_{k=1}^{n} d^w_{[k]j}\right) \\ &= \rho_1 E_w(S^0_{[n]m} + d^w_{[n]m} + p_{[n]m} + \xi^w_{[n]m} t_{rm}) + \rho_2 E_w\left(\sum_{j=1}^{m}\sum_{k=1}^{n} d^w_{[k]j}\right) \\ &= \rho_1 (p_{[n]m} + t_{rm} E_w(\xi^w_{[n]m})) + \rho_1 S^0_{[n]m} + \rho_1 E_w(d^w_{[n]m}) \\ &\quad + \rho_2 \sum_{j=1}^{m}\sum_{k=1}^{n} E_w(d^w_{[k]j}) \end{aligned} \tag{18}$$

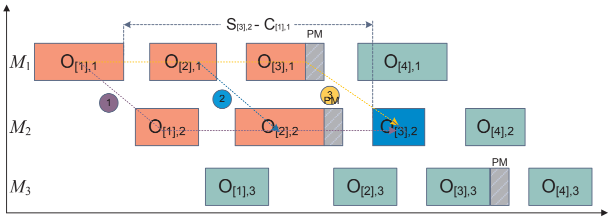


Fig. 2. The Gantt chart for $n = 4$, $m = 3$.

Since $p_{[n]m}$, $E_w(\xi^w_{[n]m})$ and $S^0_{[n]m}$ can be obtained by the mathematical formula, the only difficulty is to handle $E_w(d^w_{[k]j})$. Next, we will analyze the construction of the delay of $O_{[k]j}$ and propose a surrogate measure $\hat{d}_{[k]j}$ to estimate $E_w(d^w_{[k]j})$.

Fig. 2 shows an example with four jobs and three machines. From the perspective of operation $O_{[3]2}$, the other operations can be divided into two sets. The set consisting of $\{O_{[1]1}, O_{[2]1}, O_{[3]1}, O_{[1]2}, O_{[2]2}\}$ is called the key set of $O_{[3]2}$. The set consisting of $\{O_{[4]1}, O_{[4]2}, O_{[1]3}, O_{[2]3}, O_{[3]3}, O_{[4]3}\}$ is called the non-key set of $O_{[3]2}$. For the jobs in the key set, if breakdown happens during processing one job, it may cause the delay of $O_{[3]2}$. However, for the jobs in the non-key set, the breakdown will never cause the delay of $O_{[3]2}$. Take $O_{[1]1}$ as an example. If breakdown happens in $O_{[1]1}$, the realized finish time of $O_{[1]1}$ will be later than the planned finish time. As shown in Fig. 2, there are three paths from $O_{[1]1}$ to $O_{[3]2}$. And, the influence can be delivered to $O_{[3]2}$ through any path.

In path 1, i.e., $O_{[1]1} \to O_{[1]2} \to O_{[2]2} \to O_{[3]2}$, the idle time that can absorb the impact of breakdowns can be obtained by $Idle^{[1]1}_{[3]2}(1) = S_{[3]2} - C_{[1]1} - (p_{[1]2} + p_{[2]2} + t_{p2})$.

In path 2, i.e., $O_{[1]1} \to O_{[2]1} \to O_{[2]2} \to O_{[3]2}$, the idle time can be obtained by $Idle^{[1]1}_{[3]2}(2) = S_{[3]2} - C_{[1]1} - (p_{[2]1} + p_{[2]2} + t_{p2})$.

In path 3, i.e., $O_{[1]1} \to O_{[2]1} \to O_{[3]1} \to O_{[3]2}$, the idle time can be obtained by $Idle^{[1]1}_{[3]2}(3) = S_{[3]2} - C_{[1]1} - (p_{[2]1} + p_{[3]1})$.

It is obvious that the delay of $O_{[3]2}$ caused by $O_{[1]1}$ should be the difference between the breakdown time during $O_{[1]1}$ and the shortest idle time. The path with the shortest idle time is called key path. Since $S_{[3]2} - C_{[1]1}$ is the same for three paths, $Idle^{[1]1}_{[3]2} = S_{[3]2} - C_{[1]1} - \max\{D^{[1]1}_{[3]2}(1), D^{[1]1}_{[3]2}(1), D^{[1]1}_{[3]2}(1)\}$. Next, we will show the meaning of $D^{[1]1}_{[3]2} = \max\{D^{[1]1}_{[3]2}(1), D^{[1]1}_{[3]2}(1), D^{[1]1}_{[3]2}(1)\}$ using the network diagram. In Fig. 3, the weight of circle is the processing time of the operation, the weight of horizontal edge is the possible PM time, the weight of vertical edge is 0. The key path here is the longest path with the largest total weight, where the weights of start circle and end circle are not contained. For $D^{[1]1}_{[3]2}$, path 2 is the key path when the influence of $O_{[1]1}$ is considered for $O_{[3]2}$. It equals to $t_{p1}y_{[2]1} + p_{[2]1} + 0 + p_{[2]2} + t_{p2}y_{[3]2}$. Since $y_{[2]1} = 0, y_{[3]2} = 1$, we can get $D^{[1]1}_{[3]2} = p_{[2]1} + p_{[2]2} + t_{p2}$.

We know the expected breakdown time during $O_{[1]1}$ equals to $t_{r1}[(a_{[1]1}/\theta_1)^{\beta_1} - (b_{[1]1}/\theta_1)^{\beta_1}]$. The probability of no breakdown during $O_{[1]1}$ equals to $e^{-[(a_{[1]1}/\theta_1)^{\beta_1} - (b_{[1]1}/\theta_1)^{\beta_1}]}$. The probability of breakdown during $O_{[1]1}$ equals to $\Pr_{[1]1} = 1 - e^{-[(a_{[1]1}/\theta_1)^{\beta_1} - (b_{[1]1}/\theta_1)^{\beta_1}]}$. Let $\hat{t}_{[1]1}$ be the expected breakdown time during $O_{[1]1}$ on the condition that breakdown happens during processing $O_{[1]1}$, then $\hat{t}_{[1]1} = t_{r1}[(a_{[1]1}/\theta_1)^{\beta_1} - (b_{[1]1}/\theta_1)^{\beta_1}]/(1 - e^{-[(a_{[1]1}/\theta_1)^{\beta_1} - (b_{[1]1}/\theta_1)^{\beta_1}]})$. Even though the realized number of breakdowns may be infinity in theory, we assume there are only 2 cases. The first is no breakdown, the other is one breakdown with the time equal to $\hat{t}_{[1]1}$; it is an approximation for simplifying the calculation.

$O_{[1]1}$ is a job in the key set of $O_{[3]2}$. And, the number of jobs in the key set of $O_{[3]2}$ is 5. Then, the number of total scenarios is $2^5 = 32$. It is the exponential time complexity, which is still time consuming.

At last, we propose the following surrogate measure to approximate the delay.

$$\hat{d}_{[k]j} = \sum_{k'=1}^{k} \sum_{j'=1}^{j-1} \Pr_{[k']j'} \max\{\hat{t}_{[k']j'} - \max(S^0_{[k]j} - C^0_{[k']j'} - D^{[k']j'}_{[k]j} - \hat{d}_{[k']j'}, 0), 0\}$$
$$+ \sum_{k'=1}^{k-1} \Pr_{[k']j} \max\{\hat{t}_{[k']j} - \max(S^0_{[k]j} - C^0_{[k']j} - D^{[k']j}_{[k]j} - \hat{d}_{[k']j}, 0), 0\} \quad \forall\, k,j \tag{19}$$

Then, we can get the approximation of objective function using $\hat{d}_{[k]j}$ within a polynomial time.

4.2.2. Validation of surrogate measure

A simulation based study is executed to evaluate the correlation between the proposed measure and the corresponding robustness. This study indicates the ability of the proposed surrogate measure to predict the expected performance of schedule.

The decision variables can be divided into two critical parts. The first part consists of jobs' sequence $\boldsymbol{X}$ and PMs' positions $\mathbf{Y}$, which belong to the integer discrete variables. The second part consists of the jobs' start times **ST**, which belong to the continuous variables. In order to guarantee the solution's feasibility, idle time of each position in each machine is adopted as the decision variable instead of start time. Also, the start times of jobs can be derived according to the jobs' sequence, PMs' positions and the idle times. Then, a solution can be represented by $(\boldsymbol{X}, \mathbf{Y}, \mathbf{I})$, where $\mathbf{I}$ is the matrix containing the idle times.

A partial schedule means the schedule only consists of jobs' sequence $\boldsymbol{X}$ and PMs' positions $\mathbf{Y}$. Based on a partial schedule, we can construct a whole schedule with the determination of idle times $\mathbf{I}$. For each randomly generated case, we generate N_x instances. For each instance, we generate N_r random partial schedules. For each partial schedule, we generate N_b random $\mathbf{I}$. For each whole schedule, we randomly generate N_s samples in the simulation. The following procedure is used in the experiments. For each case, the average R2 value can be obtained by $\sum_{Nx_i \in Nx} R2_{Nx_i}/Nx$. $R2_{Nx_i}$ is the R2 value for instance Nx_i, which can be obtained by the following procedure.

Procedure II

```
For $i = 1 \ldots N_r$, a partial schedule $\sigma_i^{\mathrm{P}}$ is generated randomly. Do
  For $j = 1 \ldots N_b$, do
  (1) An entire schedule $\sigma_{ij}$ is generated by generating the idle
      times randomly based on the $\sigma_i^{\mathrm{P}}$. The idle time of each job is
      randomly generated from the interval $[0, j]$.
  (2) Calculate the estimated objective value $\hat{z}$, which is based on
      the surrogate measure (19).
  (3) Calculate the sample average
      $z = \sum_{w=1}^{N_s} \{ \sum_{j=1}^{m} \sum_{k=1}^{n} (s^w_{[k]j} - s^0_{[k]j}) + C^w_{[k]j} \}/N_s$ through
      simulation.
  (4) Record a pair data $(\hat{z}, z)$.
  END.
END
```

Perform a simple linear regression using $N_r N_b$ pairs of data.

The test problems are set as follow. Considering the parameters of the failure function, we set $\theta = \{60,80,100\}$, $\beta = \{2,3\}$ and $(t_r, t_p) = \{(8,12),(10,12),(12,12)\}$ respectively, which yields 18 breakdown parameter combinations D1–D18. For these 18 combinations, the

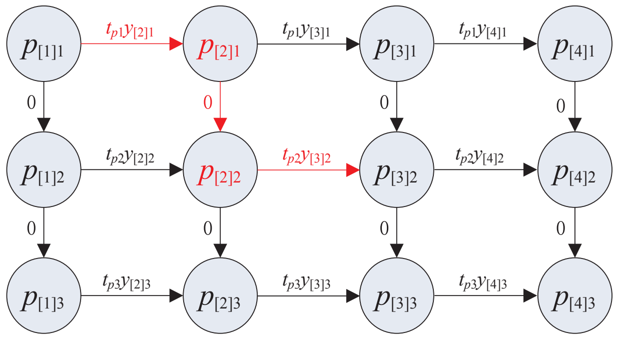


Fig. 3. The network diagram of $\boldsymbol{D}$.

Table 1
Average and standard deviation of R2.

θ,β,t_p,t_r	0.9		0.7		0.5		0.3		0.1	
	avg	*std*	*avg*	*std*	*avg*	*Std*	*avg*	*std*	*avg*	*Std*
60, 2, 12, 8	0.990	0.002	0.990	0.002	0.988	0.002	0.988	0.002	0.988	0.002
60, 2, 12, 10	0.992	0.001	0.991	0.001	0.990	0.001	0.990	0.001	0.990	0.001
60, 2, 12, 12	0.993	0.003	0.991	0.002	0.991	0.002	0.990	0.002	0.990	0.002
80, 2, 12, 8	0.986	0.002	0.988	0.002	0.986	0.002	0.985	0.002	0.984	0.002
80, 2, 12, 10	0.990	0.002	0.988	0.002	0.987	0.002	0.986	0.002	0.985	0.002
80, 2, 12, 12	0.992	0.002	0.990	0.002	0.989	0.002	0.988	0.002	0.988	0.002
100, 2, 12, 8	0.971	0.004	0.983	0.004	0.980	0.004	0.978	0.004	0.977	0.004
100, 2, 12, 10	0.984	0.002	0.985	0.001	0.983	0.002	0.981	0.002	0.981	0.002
100, 2, 12, 12	0.991	0.001	0.988	0.001	0.986	0.002	0.985	0.002	0.985	0.002
60, 3, 12, 8	0.963	0.005	0.978	0.004	0.975	0.004	0.973	0.004	0.972	0.004
80, 3, 12, 10	0.976	0.005	0.981	0.004	0.978	0.004	0.976	0.004	0.975	0.004
100, 3, 12, 12	0.981	0.003	0.980	0.003	0.977	0.003	0.976	0.003	0.975	0.003
60, 3, 12, 8	0.925	0.007	0.975	0.004	0.971	0.004	0.969	0.004	0.968	0.004
80, 3, 12, 10	0.955	0.006	0.977	0.003	0.974	0.003	0.971	0.003	0.970	0.003
100, 3, 12, 12	0.968	0.002	0.977	0.002	0.974	0.002	0.972	0.002	0.971	0.002
60, 3, 12, 8	0.868	0.009	0.975	0.003	0.971	0.003	0.969	0.003	0.967	0.003
80, 3, 12, 10	0.914	0.003	0.972	0.003	0.968	0.003	0.966	0.003	0.964	0.003
100, 3, 12, 12	0.943	0.005	0.973	0.003	0.969	0.003	0.966	0.003	0.965	0.003
Random	0.983	0.004	0.985	0.004	0.983	0.004	0.982	0.005	0.981	0.004
Average	0.967	0.004	0.982	0.003	0.980	0.003	0.978	0.003	0.978	0.003

failure function for each machine is the same. The case, where the failure functions of dissimilar machines are different, was also conducted. θ_j is uniformly generated from the interval $[60, 100]$. β_j is uniformly generated from interval $[2, 3]$. t_{pj} is uniformly generated from interval $[12, 15]$. t_{rj} is uniformly generated from the interval $[8, 12]$. Then, we have 19 combinations. For each combination, we have 5 cases when the solution weight ρ_2 is set to $\{0.9, 0.7, 0.5, 0.3, 0.1\}$. Then, we have 95 cases in total. For each randomly generated case, we conducted the experiments with $N_x = 10$, $N_r = 10$, $N_b = 20$ and $N_s = 10{,}000$.

We recorded the average value of R2 for each case and also recorded the standard deviation to show the stability of the correlation coefficient in the Table 1. The average values of all the instances for each ρ_2 are also shown in the last row of "Average".

From Table 1, we can find that the correlation between the objective and the surrogate measure is very high and the standard deviation of R2 value is very small. The accuracy is not related to the instances' parameters settings. It means that the surrogate measure we proposed is very effective. The adoption of this measure can guarantee the effectiveness of the two-loop algorithm and reduce the computation time dramatically at the same time.

5. A two-loop algorithm

$$z = \rho_1(p_{[n]m} + t_{rm}E(\xi^w_{[n]m})) + \rho_1 S^0_{[n]m} + \rho_1 E(d^w_{[n]m}) + \rho_2 \sum_{j=1}^{m}\sum_{k=1}^{n} E(d^w_{[k]j})$$
$$= f_1 + f_2$$
$$f_1 = \rho_1(p_{[n]m} + t_{rm}E(\xi^w_{[n]m}))$$
$$f_2 = \rho_1 S^0_{[n]m} + \rho_1 E(d^w_{[n]m}) + \rho_2 \sum_{j=1}^{m}\sum_{k=1}^{n} E(d^w_{[k]j}) \tag{20}$$

From above Eq. (20), we can find that the first part is only affected by the jobs' sequence and PMs' positions. While, the idle times inserted into the schedule only have an impact on the second part. More idle times lead to larger $S^0_{[n]m}$ and smaller $d^w_{[k]j}$, which results in a compromise.

Combining the division of decision variables in Section 4.2.2 and the analysis in last paragraph, a two-loop search algorithm is developed to solve this problem. The outer loop is based on the local search to optimize the jobs' sequence and the maintenance theory to optimize the PMs' positions for each sequence. The inner loop adopts the genetic algorithm to optimize the idle times when jobs' sequence and PMs' positions are fixed. And, the result of inner loop will return to the outer loop as an evaluation criterion to guide the search direction in outer loop.

5.1. The outer loop

A whole solution is represented by $(\boldsymbol{X},\mathbf{Y},\mathbf{I})$. We want to search for $\boldsymbol{X}$ and $\mathbf{Y}$ in the outer loop. Three different kinds of criteria are proposed to evaluate a jobs' sequence $\boldsymbol{X}$.

Criterion 1. Assuming the machines are always available, the makespan for $(\boldsymbol{X},0,0)$ can be calculated directly. 0 means that we do not need to perform the PMs and insert the idle times into the schedule. The makespan is denoted as $z^1(\boldsymbol{X})$.

Criterion 2. The machine's failure uncertainty is considered. According to the maintenance theory, in order to maximize the availability of the machine, differentiation and algebraic analysis results in an optimal PM interval of $T^*_j = \theta_j\{t_{pj}/[t_{rj}(\beta_j - 1)]\}^{1/\beta_j}$. Insert the PMs into each machine as late as possible, while keeping the constraint that the machine's age is always smaller than T^*_j. Then, we get the PMs' positions matrix $\mathbf{Y}$. We do not insert the idle times into the schedule, then we get a solution $(\boldsymbol{X},\mathbf{Y},0)$. For this solution, we calculate the objective using the surrogate measure as the method in Section 4.2.1. It is denoted as $z^2(\boldsymbol{X})$.

Criterion 3. The machine's failure uncertainty is considered and the PMs' position $\mathbf{Y}$ is decided similar to that in criterion 2. Then, the idle times matrix $\mathbf{I}$ is decided according to **Algorithm II**. Then, $z^3(\boldsymbol{X})$ can be calculated using the simulation **Procedure** for $(\boldsymbol{X},\mathbf{Y},\mathbf{I})$.

The *NEH* heuristic (Nawaz, Enscore, & Ham, 1983) has been so far commonly regarded as the best constructive heuristic for $Fm//C_{max}$. Assuming the machines are always available, the start point of the jobs' sequence $\boldsymbol{X}^0$ is obtained by the *NEH* method. Then, a local search is adopted to improve the solution. The local search is performed on the jobs' sequence. For $\boldsymbol{X}^0$, a neighbour can be obtained by changing the positions of any two jobs. The neighbourhood of $\boldsymbol{X}^0$ is denoted as $N(\boldsymbol{X}^0)$. Then, the size of the neighbourhood is $n(n-1)/2$. If the performance of a new jobs' sequence is better than the old one, the new jobs' sequence is adopted to replace the old one. This work will be done repeatedly until no improvement is found. After the **Algorithm I** stops, we can get a final jobs' sequence $\boldsymbol{X}$. For this $\boldsymbol{X}$, the PMs' positions and idle times are already obtained in Criterion 3.

```
Algorithm I:
    Step 1. Set $z^* = z^3(\boldsymbol{X}^0)$.
    Step 2. Construct the neighbourhood of $\boldsymbol{X}^0$.
    Step 3. For each $\boldsymbol{X}$ in $N(\boldsymbol{X}^0)$, calculate $z^1(\boldsymbol{X})$ and $z^2(\boldsymbol{X})$.
    Step 4. Choose the sequence with the smallest $z^1(\boldsymbol{X})$. Denote it $\boldsymbol{X}_1^0$.
    Step 5. Choose the sequence with the smallest $z^2(\boldsymbol{X})$. Denote it $\boldsymbol{X}_2^0$ .
    Step 6. Calculate $z^3(\boldsymbol{X}_1^0)$ and $z^3(\boldsymbol{X}_2^0)$ .
    Step 7. If $z^3(\boldsymbol{X}_1^0) < z^3(\boldsymbol{X}_2^0)$, set $\boldsymbol{X}_c^0 = \boldsymbol{X}_1^0$; else, set $\boldsymbol{X}_c^0 = \boldsymbol{X}_2^0$.
    Step 8. If $z^* > z^3(\boldsymbol{X}_c^0)$, set $\boldsymbol{X}^0 = \boldsymbol{X}_c^0$, goto Step 2; else, stop.
```

5.2. The inner loop

When jobs' sequence and PMs' positions are fixed, we only need to optimize the idle times, which are continuous variables. From the surrogate measure mentioned above (19), we know that the approximated objective function is not convex, which means the local minimum cannot be guaranteed to be global minimum. Thus, a genetic algorithm is adopted to search for the global minimum instead of algorithms based on gradient descent.

The detailed procedures for the genetic algorithm can be found as follows.

Algorithm II:

The representation of solution. The chromosome is represented by a matrix **I**, whose elements are non-negative real values. I_{kj} is the idle time inserted on the kth position of machine M_j. We can get the planned start times of jobs for each **I**.

$$\mathbf{I} = \begin{bmatrix} I_{11} & I_{21} & \dots & I_{n1} \\ I_{12} & I_{22} & \dots & I_{n2} \\ . & . & I_{kj} & . \\ I_{1m} & I_{2m} & \dots & I_{nm} \end{bmatrix}.$$

Population initialization. N is the size of the population. N is set to 200 in this study. 200 individuals are generated randomly here.

Fitness function. Fitness function, which is relevant to the objective function, is used to determine which chromosome will be reproduced and survived into the next generation. The fitness f can thus be defined as $f(ch) = 1/z(ch)$, where z is the objective value of chromosome ch. $z(ch)$ is calculated based on the surrogate measure. In addition, we introduce the linear acceleration function to our algorithm. Linear acceleration, i.e., linear scaling, adjust the fitness values of chromosomes such that the best individual gets a number of expected offspring and thus prevent it from reproducing too many. Thus, the new fitness f' can be obtained by $f'(ch) = \lambda f(ch) + \gamma$. λ and γ can be obtained by $\lambda\overline{f(ch_i)} + \gamma = \overline{f(ch_i)}$, $\lambda\max_{1\leqslant i\leqslant N}\{f(ch_i)\} + \gamma = M\overline{f(ch_i)}$. ch_i represents the chromosome in the population and $\overline{f(ch_i)}$ represents the average fitness value of all chromosomes. The selection probability of ch_i can be obtained by $P(ch_i) = f'(ch_i)/\sum_{i=1}^{N} f'(ch_i)$.

Genetic Operator. For selection, the roulette-wheel method is adopted. For crossover, the linear recombination is adopted. The probability of crossover is set to 0.75. For two individuals ch_1 and ch_2, α_1 and α_2 are generated randomly from the interval $[-0.25, 1.25]$. Then, two offspring can be obtained by $off_1 = \alpha_1 ch_1 + (1-\alpha_1)ch_2$, $off_2 = (1-\alpha_2)ch_1 + \alpha_2 ch_2$. For mutation, Gauss mutation is adopted, where the probability of mutation is set to 0.05. Gauss mutation is a widely used genetic operator, which adds a unit Gaussian distributed random value to the chosen gene.

Population evolution. Elite strategy is adopted, and the best individual in last generation is kept into the new generation directly. Then, all the offsprings, except the worst one, evolve into the new generation.

Stop criterion. The iteration stops once the maximum generation is reached, which is set to 500. The computation time is easier to control for this stop criterion.

6. Numerical results

In order to examine the computational performance of the proactive model and the proposed two-loop algorithm, the solution approach was coded in C # platform and tested on a large set of randomly generated instances. Then, the program was run in a personal computer with an Intel Core i3 3.30 GHz CPU and 4 GB RAM.

6.1. Comparison between the proactive model and the traditional way

In order to demonstrate the effectiveness of our joint model, the comparison between JM and the traditional way is presented in this subsection. In the traditional way, the production scheduling and maintenance policy are determined independently, and no idle times are inserted in the schedule. The traditional way can be described as follows. The jobs' sequence is determined according to *NEH* heuristic firstly. T_j^*, which can maximize the machine's availability, is adopted; the PMs are inserted into the jobs' sequence as late as possible as long as the machine's age does not exceed T_j^*. Let S1 be the solution obtained by this way.

The parameters of instances are set as follows. 20 combinations are generated when we set the number of jobs $n = \{20, 40, 60, 80, 100\}$, and the number of machines $m = \{5, 10, 15, 20\}$. For each combination, we set the weight of solution robustness $\rho_2 = \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Then, we get 100 cases. For each case, we generate ten instances for a total of 1000 instances. For each instance, the parameters are set as follows. The processing times of jobs are integers uniformly generated from an interval $[9, 29]$. θ_j is uniformly generated from the interval $[60, 100]$; β_j is uniformly generated from the interval $[2, 3]$; t_{pj} is uniformly generated from the interval $[12, 15]$; and t_{rj} is uniformly generated from the interval $[8, 12]$.

6.1.1. Significance of idle times

The idle times inserted in the schedule have a huge impact on the problem. Thus, we want to validate the significance of idle times through numerical experiments. From solution S1, we can get another solution S2 by inserting the idle times which can be calculated using **Procedure II**. It means that the only difference between S1 and S2 is that the idle times are not inserted in S1.

Let $z_1(S)$ be the quality robustness and $z_2(S)$ be the solution robustness of schedule S. Fig. 4(a) shows the $z_1(S_2)/z_1(S_1)$ ratios for all instances, Fig. 4(b) shows the $z_2(S_2)/z_2(S_1)$ ratios for the same instances, and Fig. 4(c) shows the $z(S_2)/z(S_1)$ ratios for the same instances. In Fig. 4(a), all ratios exceed 1, which means the quality robustness of S2 is worse than that of S1. In Fig. 4(b) and (c), all ratios are less than 1, which means the robustness performance and objective value of S2 are always better than those of S1. Moreover, different variation tendencies of these ratios under different ρ_2 values can be observed. For example, when $\rho_2 = 0.1$, the quality robustness of S1 and S2 are almost the same, while the solution robustness of S2 is under 10 percent of S1's value. On

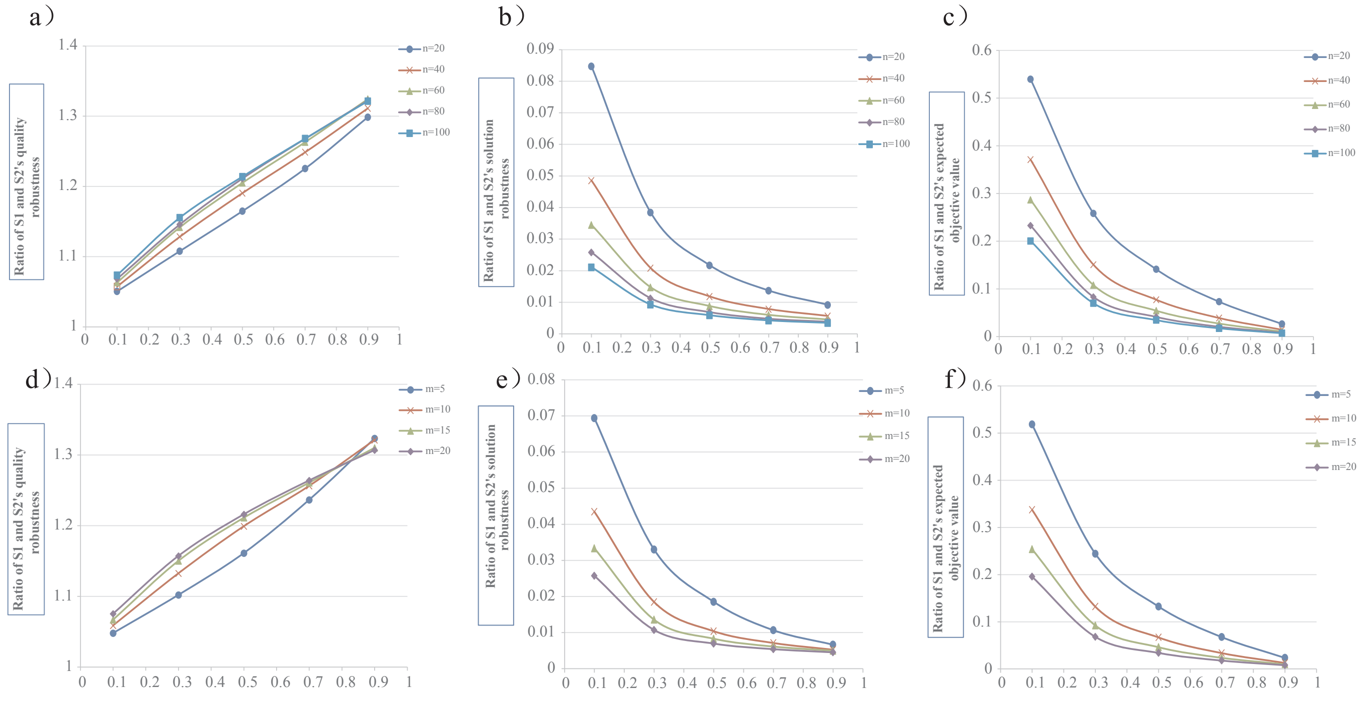


Fig. 4. The comparison between S1 and S2 under different weight ρ_2.

the other end, when $\rho_2 = 0.9$, the quality robustness of S2 is nearly 30 percent more than S1's value, while the solution robustness of S2 is only about one percent of S1's value. Considering both solution and quality robustness, in Fig. 4(c), the improvement of S2 compared with S1 is higher with larger ρ_2, e.g., the $z(S_2)$ is only about five percent of $z(S_1)$ when $\rho_2 = 0.9$. This can be explained as follows. When ρ_2 is larger, more idle times are inserted into S2, which makes the ratio $z_1(S_2)/z_1(S_1)$ larger. But, the ratio $z_2(S_2)/z_2(S_1)$ is much smaller at the same time. Since the objective $z(S)$ equals to $(1-\rho_2)z_1(S) + \rho_2 z_2(S)$, it is easy to get the ratio $z(S_2)/z(S_1)$ will be smaller with larger ρ_2. Thus, the insertion of buffer times will slightly affect the makespan more or less; while, it can evidently improve the system's robustness.

The data in Fig. 4a–c are classified based on n. It is evident that the improvement of our model compared with traditional method is larger when n is larger. The data in Fig. 4d–f are classified based on m. We can know that the improvement is larger when m is larger.

6.1.2. Analysis of solutions

The impact of idle times on the objective value is discussed in Section 6.1.1. The amount of idle times and the places where they are inserted are discussed in this subsection. In addition, the ratio between the idle time and makespan is also discussed in order to settle the role of idle times.

For a system with m machines and n jobs, there will be m x n positions to insert the idle times. Fig. 5 shows the total amount of idle times for the obtained solutions S2. It is reasonable that the amount increases with the increment of ρ_2, i.e., more idle times will be inserted when the weight of solution robustness is larger. Besides, Fig. 5a shows that the amount is basically proportional to the value of n; Fig. 5b shows that the amount is basically proportional to the value of m.

For a flow shop system in the traditional research field, machines may be idle because of starvation. However, the machine may be idle even if it is not starve in our study considering the existence of inserted buffer times. That is the reason why the makespan degenerates gradually with the insertion of buffer times. Let AIT be the total idle times which could be avoided in the whole system. Fig. 6 shows the ratio R_1 between AIT and makespan of S2, which can be calculated as $AIT/z_1(S_2)$. Firstly, AIT is very large compared with $z_1(S_2)$, e.g., AIT is almost 4 times larger than $z_1(S_2)$ when $m = 20$, $\rho_2 = 0.9$. It is reasonable since the makespan is the finish time of last job in the last machine and AIT is the sum of idle times in all machines. Secondly, R_1 increases with the increment of the number of jobs; while, this kind of relationship weakens gradually when n is close to 100.

In order to analyze the time construction in the last machine, Fig. 7

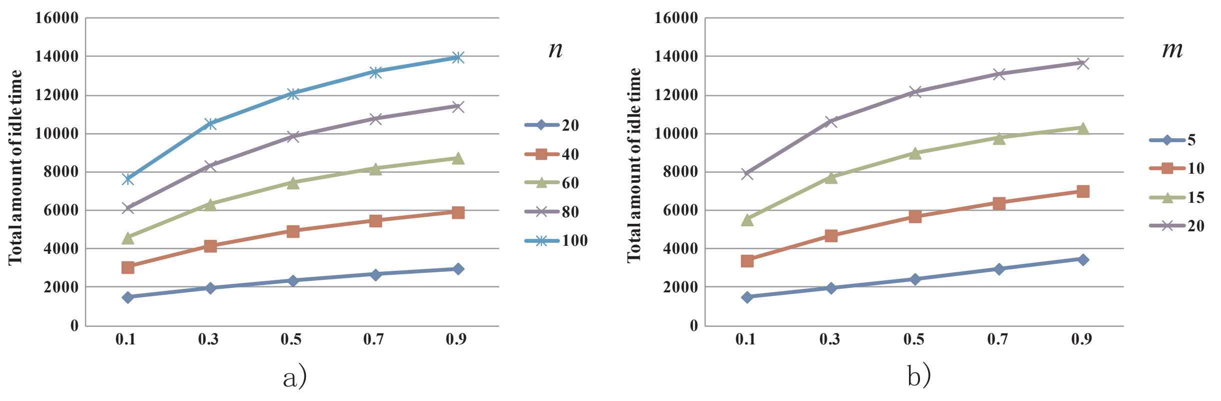


Fig. 5. The total amount of idle times in the whole system.

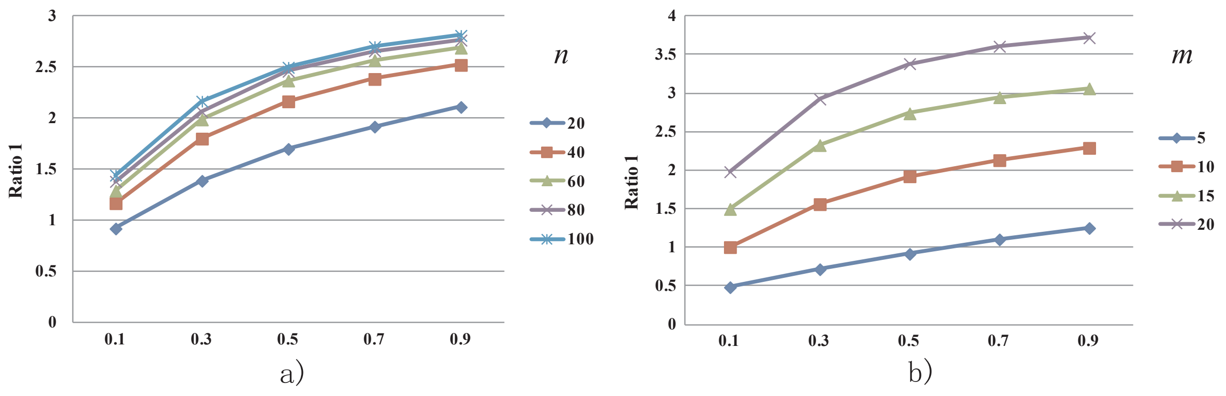


Fig. 6. The ratio between the total amount of idle time and the makespan.

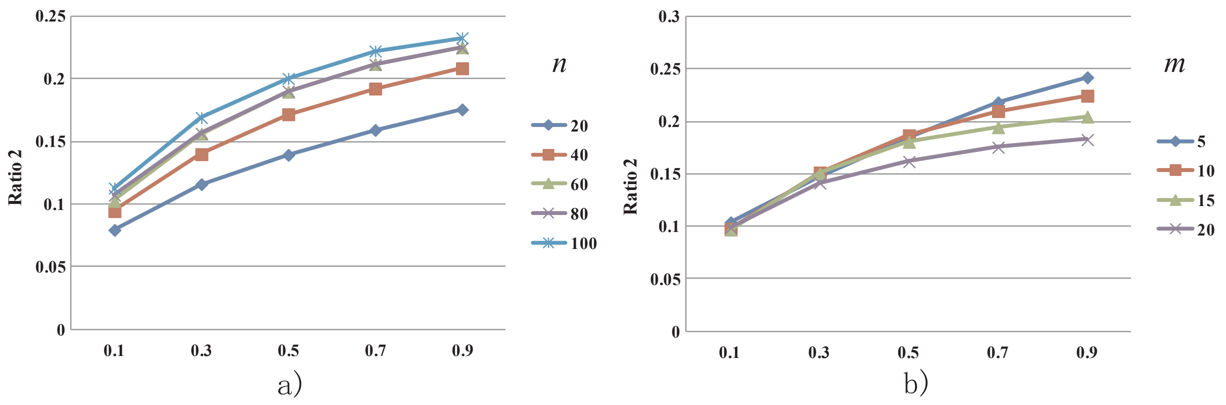


Fig. 7. The ratio between the amount of idle time in machine M_m and the makespan.

shows the ratio R_2 between AIT_m and makespan of S2, where AIT_m is the idle time that could be avoided in the machine M_m. Firstly, R_2 is around 0.15, which means AIT_m is only a small part of makespan. Secondly, R_2 increases with the increment of the number of jobs. Thirdly, R_2 is not related to the value of m when $\rho_2 < 0.5$; while, R_2 decreases with the increment of m when $\rho_2 > 0.5$.

There are two difficulties to optimize the allocation of buffer times even if the jobs' sequence has already been fixed. One is the machines' relationship in the flow shop; the other one is the deterioration of machine. If we ignore the relationship between upstream and downstream machines, i.e., it is assumed that the machines are independent, more buffer times should be inserted into the machines whose β_j is larger. And, for each machine, more buffer times should be inserted into the positions where the machine's age is larger. If we consider the machines' relationship and ignore the deterioration phenomenon, i.e., it is assumed that the breakdown probability is constant, more buffer times should be inserted into the position which is not in the critical chain of this flow shop. For a Gantt chart of production scheduling in flow shop, the critical chain is the longest path from the first job in the first machine to the last job in the last machine. In addition, the buffer times in different positions interrelate with each other. For example, if some buffer times are inserted into one position, then less buffer time will be inserted into the next position. Above in all, the allocation of buffer times will be erratic in the whole system, which can be shown in an example. Fig. 8 shows the contour map of inserted buffer times for an instance where $n = 20$, $m = 5$. That is reason we use the genetic algorithm to search the optimal allocation of buffer times instead of using the constructive heuristic rules.

6.1.3. Significance of local search

The figures in Section 6.1.1 show that the idle times have a significant impact on the solution's performance. In this subsection, we want to know the influence of jobs' sequence. Take S2 as the start point, we can get another solution S3 using the two-loop algorithm. Table 2 shows the improvement of S3 compared with S2, which is calculated as $(z(S_2)-z(S_3))/z(S_2)*100\%$. From the table, we can find that the improvement is limited especially for the case with a small ρ_2.

From the perspective of algorithm complexity, searching the jobs' sequence in the outer loop results in calling the inner loop frequently, which will increase the computation time dramatically since the inner loop is a genetic algorithm. Thus, we only use a local search to handle the jobs' sequence instead of some other complex search methods in order to control the computation time. Thus, the managers should pay more attention to the buffer times rather than the jobs' sequences. However, although the inserted idle time will partially counteract the impact of jobs' sequence when the weight of solution robustness is

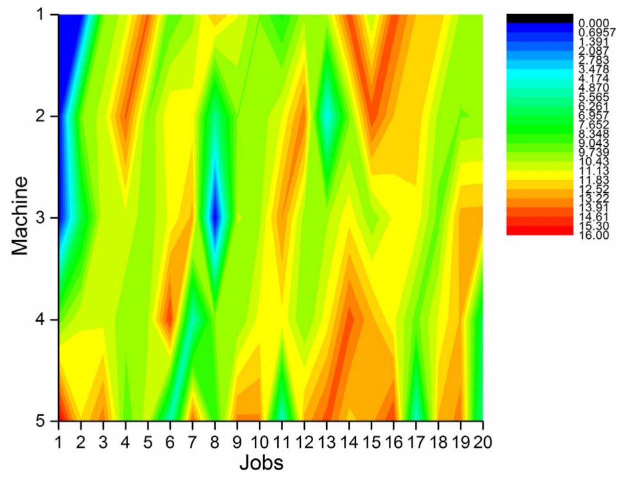


Fig. 8. The allocation of idle times in 100 positions.

Table 2
The comparison between S2 and S3.

(n, m)	ρ_2				
	0.1	0.3	0.5	0.7	0.9
(20, *)	0.45	0.54	0.52	1.04	3.09
(40, *)	0.27	0.39	0.46	0.95	2.56
(60, *)	0.22	0.32	0.56	1.07	2.36
(80, *)	0.17	0.22	0.56	0.79	2.33
(100, *)	0.20	0.23	0.51	1.25	2.64
(*, 5)	0.28	0.34	0.48	0.84	1.95
(*, 10)	0.21	0.29	0.47	0.85	1.82
(*, 15)	0.29	0.35	0.41	0.96	3.42
(*, 20)	0.27	0.38	0.72	1.43	3.20
Average	0.26	0.34	0.52	1.02	2.60

large, the influence of jobs' sequence is still worth investigating. It will affect the amount of idle times and the places where they will be inserted. One advantage of the two-loop algorithm is that it can find the optimal allocation of idle times for different jobs' sequences automatically. Besides, when the weight of solution robustness becomes larger, the impact of jobs' sequence on the final objective value becomes larger.

6.2. Balancing solution robustness and quality robustness

Obviously, there is a trade-off between quality robustness and solution robustness in our problem. Though the algorithms based on the non-dominated solutions can get better Pareto curve when solving multi-objective problems, we use a linear combination of two objectives by varying the weight ρ_2 to obtain different biased solutions to simplify our study. One instance in the test set is selected randomly to get its Pareto curve. The number of jobs is 20 and the number of machines is 5. ρ_2 is set to $\{0.01, 0.04, 0.07, 0.1, 0.3, 0.5, 0.7, 0.9\}$. Fig. 9 depicts the Pareto curve between two objectives, which shows the trade-off between quality robustness and solution robustness. Each Pareto point identifies a scheduling solution according to different preferences for two objectives. The traditional solution is also illustrated in Fig. 9. The Pareto curve in Fig. 9 shows that the performance of solution robustness improves fast and the performance of quality robustness deteriorates slowly at the left of the curve. It is obvious that the quality robustness of the traditional solution is the best, since there are no buffers in this solution. Meanwhile, the performance of solution robustness of the

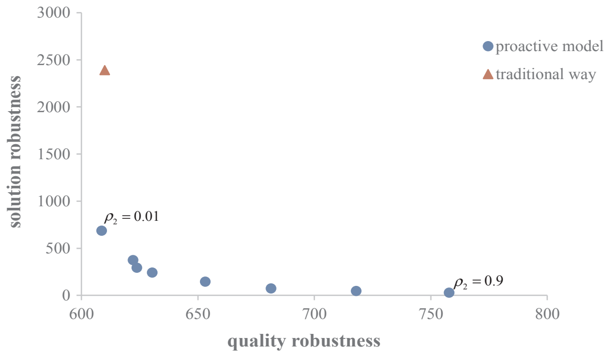


Fig. 9. The balance between quality robustness and solution robustness.

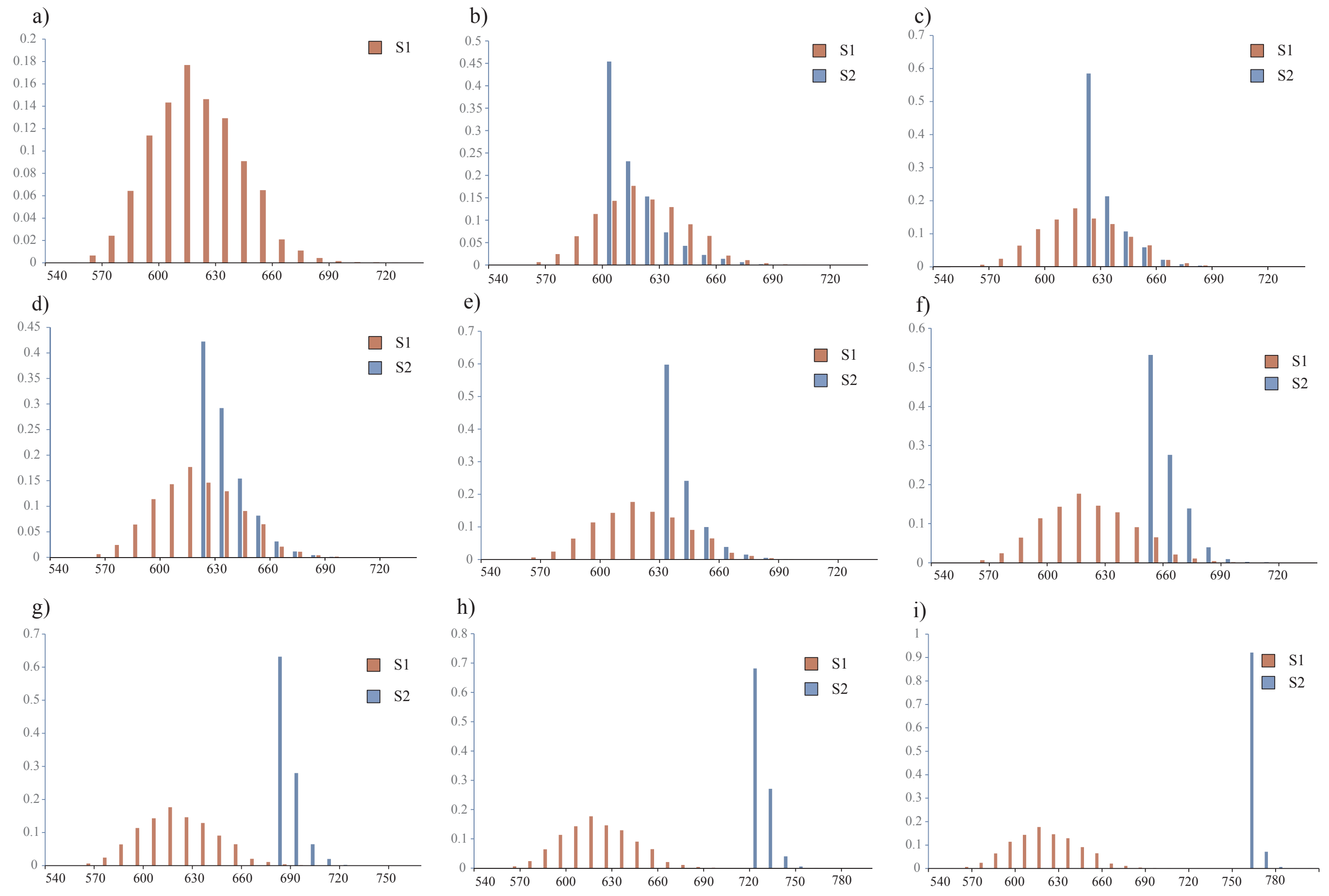


Fig. 10. Probability distributions of S1's and S2's C_{max}. (a) ρ_2 = 0, (b) ρ_2 = 0.01, (c) ρ_2 = 0.04, (d) ρ_2 = 0.07, (e) ρ_2 = 0.1, (f) ρ_2 = 0.3, (g) ρ_2 = 0.5, (h) ρ_2 = 0.7, and (i) ρ_2 = 0.9.

traditional solution is the worst. We can significantly improve the performance of solution robustness with a little sacrifice of quality robustness if we insert some idle times into this schedule. As shown in this figure, the quality robustness of the traditional solution and that of $\rho_2 = 0.01$ are almost the same. Meanwhile, the solution robustness of the traditional solution is 2390.4 while the solution robustness of $\rho_2 = 0.01$ is only 688.3. In addition, when $\rho_2 = 0.9$, the solution robustness is almost zero, which means the stability of system is quite well.

Fig. 10a shows the sampled probability distribution of S1's makespan. The expectation of S1's makespan is 609. Fig. 10b–i shows the comparison between the probability distribution of S1's makespan and probability distributions of S2's makespan with different weights. From Fig. 10, we can find that the standard deviation of S1's makepsan is the largest and the standard deviation of S2's makespan becomes smaller when the weight ρ_2 is larger. Comparing the results in Fig. 9, we can get that the stability of makespan is positively correlated to the solution robustness. In Fig. 10b–e, although the $E(C_{max})$ of S1 is less than that of S2, the maximum of the actual C_{max} of S2 and that of S1 are almost the same. Moreover, the actual C_{max} of S1 exceeds its expectation by nearly 50%, while there is a high chance that the actual C_{max} of S2 lies within its expectation. In industry, the market department usually promises a due date to the customers, higher stability of C_{max} means less delay and a better reputation. Hence, even if the solution robustness is not considered, it is also necessary to insert some appropriate idle times into the schedule considering the stability of C_{max}. And, the amount of idle times depends on the manager's opinion about the weight of robustness.

7. Conclusion

This paper proposes a proactive joint model of production scheduling and maintenance planning that considers both quality robustness and solution robustness in flow shops subject to failure uncertainty. In order to avoid the time consuming Monte Carlo sampling, an effective surrogate measure is proposed to approximate the exact value of objective function. The decision variables are divided into two big parts, based on which a two-loop algorithm is proposed to solve the joint model. Numerical results indicate that the performance of solution robustness and stability of quality robustness can be significantly improved using our algorithm compared with the solutions from the traditional algorithm. Meanwhile, it is validated that the impact of idle times is much larger than the impact of jobs' sequence.

This paper works on flow shops with one machine in each stage, and can be extended to the flexible flow shops with multiple machines in each stage. The current proposed idea can also be explored to handle other uncertainties such as variations of jobs' processing times, arrival of new jobs, and job cancellations; and it is interesting to find out how such uncertainties affect the operational decisions.

Acknowledgment

This research is supported by National Natural Science Foundation of China (Nos. 71171130, 61473211 and 71502129).

References

Aggoune, R. (2004). Minimizing the makespan for the flow shop scheduling problem with availability constraints. *European Journal of Operational Research, 153*(3), 534–543.
Aggoune, R., & Portmann, M. C. (2006). Flow shop scheduling problem with limited machine availability: A heuristic approach. *International Journal of Production Economics, 99*(1–2), 4–15.
Al-Hinai, N., & ElMekkawy, T. Y. (2011). Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm. *International Journal of Production Economics, 132*, 279–291.
Allaoui, H., Lamouri, S., Artiba, A., & Aghezzaf, E. (2008). Simultaneously scheduling n jobs and the preventive maintenance on the two-machine flow shop to minimize the makespan. *International Journal of Production Economics, 112*, 161–167.
Bonfill, A., Espuna, A., & Puigjaner, L. (2008). Proactive approach to address the uncertainty in short-term scheduling. *Computers & Chemical Engineering, 32*, 1689–1706.
Breit, J. (2004). An improved approximation algorithm for two-machine flow shop scheduling with an availability constraint. *Information Processing Letters, 90*(6), 273–278.
Breit, J. (2006). A polynomial-time approximation scheme for the two-machine flow shop scheduling problem with an availability constraint. *Computers and Operations Research, 33*(8), 2143–2153.
Briskorn, D., Leung, J., & Pinedo, M. (2011). Robust scheduling on a single machine using time buffers. *IIE Transactions, 43*, 383–398.
Cassady, C. R., & Kutanoglu, E. (2003). Minimizing job tardiness using integrated preventive maintenance planning and production scheduling. *IIE Transactions, 35*, 503–513.
Cheng, T. C. E., & Wang, G. (2000). An improved heuristic for two-machine flowshop scheduling with an availability constraint. *Operations Research Letters, 26*(5), 223–229.
Choi, B. C., Lee, K., Leung, J. Y. T., & Pinedo, M. L. (2010). Flow shops with machine maintenance: Ordered and proportionate cases. *European Journal of Operational Research, 207*, 97–104.
Cui, W. W., Lu, Z., & Pan, E. (2014). Integrated production scheduling and maintenance policy for robustness in a single machine. *Computers & Operations Research, 47*, 81–91.
Deng, G., & Gu, X. (2012). A hybrid discrete differential evolution algorithm for the no-idle permutation flow shop scheduling problem with makespan criterion. *Computers and Operations Research, 39*(9), 2152–2160.
Gara-Ali, A., & Espinouse, M. L. (2014). Erratum to: Simultaneously scheduling n jobs and the preventive maintenance on the two-machine flow shop to minimize the makespan [Int. J. Prod. Econ. 112 (2008) 161–167]. *International Journal of Production Economics, 153*, 361–363.
Goren, S., & Sabuncuoglu, I. (2008). Robustness and stability measures for scheduling: single-machine environment. *IIE Transactions, 40*, 66–83.
Hadda, H. (2012). A polynomial-time approximation scheme for the two machine flow shop problem with several availability constraints. *Optimization Letters, 6*, 559–569.
Hadda, H. (2015). A note on simultaneously scheduling n jobs and the preventive maintenance on the two-machine flow shop to minimize the makespan. *International Journal of Production Economics, 159*, 221–222.
Hadda, H., Dridi, N., & Hajri-Gabouj, S. (2010). An improved heuristic for two-machine flowshop scheduling with an availability constraint and nonresumable jobs. *4OR-Quarterly Journal of Operations Research, 8*(1), 87–99.
Herroelen, W., & Leus, R. (2005). Project scheduling under uncertainty: survey and research potentials. *European Journal of Operational Research, 165*, 289–306.
Jabbarizadeh, F., Zandieh, M., & Talebi, D. (2009). Hybrid flexible flowshops with sequence-dependent setup times and machine availability constraints. *Computers & Industrial Engineering, 57*, 949–957.
Jahangirian, M., Eldabi, T., Naseer, A., Stergioulas, L. K., & Young, T. (2010). Simulation in manufacturing and business: A review. *European Journal of Operational Research, 203*(1), 1–13.
Kheirandish, O., Tavakkoli-Moghaddam, R., & Karimi-Nasab, M. (2015). An artificial bee colony algorithm for a two-stage hybrid flowshop scheduling problem with multilevel product structures and requirement operations. *International Journal of Computer Integrated Manufacturing, 28*(5), 437–450.
Kubzin, M. A., & Strusevich, V. A. (2005). Two-machine flow shop no-wait scheduling with machine maintenance. *4OR-Quarterly Journal of Operations Research, 3*, 303–313.
Kubzin, M. A., & Strusevich, V. A. (2006). Planning machine maintenance in two-machine shop scheduling. *Operations Research, 54*(4), 789–800.
Lee, C. Y. (1997). Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint. *Operations Research Letters, 20*(3), 129–139.
Leon, V. J., Wu, S. D., & Storer, R. H. (1994). Robust measures and robust scheduling for job shops. *IIE Transactions, 26*, 32–43.
Lu, Z., Cui, W. W., & Han, X. (2015). Integrated production and preventive maintenance scheduling for a single machine with failure uncertainty. *Computers & Industrial Engineering, 80*, 236–244.
Ma, Y., Chu, C., & Zuo, C. (2010). A survey of scheduling with deterministic machine availability constraints. *Computers and Industrial Engineering, 58*(2), 199–211.
Naderi, B., Zandieh, M., & Aminnayeri, M. (2011). Incorporating periodic preventive maintenance into flexible flowshop scheduling problems. *Applied Soft Computing, 11*, 2094–2101.
Nawaz, M., Enscore, E., & Ham, I., Jr. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega, the International Journal of Management Science, 11*, 91–95.
O'Donovan, R., Uzsoy, R., & McKay, K. N. (1999). Predictable scheduling on a single machine with breakdowns and sensitive jobs. *International Journal of Production Research, 37*, 4217–4233.
Rahmani, D., & Heydari, M. (2014). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *Journal of Manufacturing Systems, 33*, 84–92.
Rahmani, D., Heydari, M., Makui, A., & Zandieh, M. (2013). A new approach to reducing the effects of stochastic disruptions in flexible flow shop problems with stability and nervousness. *International Journal of Management Science and Engineering Management, 8*(3), 173–178.
Rahmani, D., & Ramezanian, R. (2016). A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study. *Computers & Industrial Engineering, 98*, 360–372.
Ruiz, R., García-Díaz, J. C., & Maroto, C. (2007). Considering scheduling and preventive maintenance in the flowshop sequencing problem. *Computers & Operations Research, 34*, 3314–3330.

Sanlaville, E., & Schmidt, G. (1998). Machine scheduling with availability constraints. *Acta Informatica, 35*(9), 795–811.

Schmidt, G. (2000). Scheduling with limited machine availability. *European Journal of Operational Research, 121*(1), 1–15.

Shoaardebili, N., & Fattahi, P. (2015). Multi-objective meta-heuristics to solve three-stage assembly flow shop scheduling problem with machine availability constraints. *International Journal of Production Research, 53*, 944–968.

Sortrakul, N., Nachtmann, H. L., & Cassady, C. R. (2005). Genetic algorithms for integrated preventive maintenance planning and production scheduling for a single machine. *Computers in Industry, 56*, 161–168.

Vahedi-Nouri, B., Fattahi, P. R., & Ramezanian, R. (2013). Hybrid firefly-simulated annealing algorithm for the flow shop problem with learning effects and flexible maintenance activities. *International Journal of Production Research, 51*(12), 3501–3515.

Vahedi-Nouri, B., Fattahi, P., Tavakkoli-Moghaddam, R., & Ramezanian, R. (2014). A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints. *International Journal of Advanced Manufacturing Technology, 73*, 601–611.

Vonder, S. V., Demeulemeester, E., & Herroelen, W. (2008). Proactive heuristic procedures for robust project scheduling: An experimental analysis. *European Journal of Operational Research, 189*(3), 723–733.

Wang, S., & Liu, M. (2013). A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem. *Computers and Operations Research, 40*(4), 1064–1075.

Wang, S., & Liu, M. (2014). Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method. *International Journal of Production Research, 52*(5), 1495–1508.