International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: https://www.tandfonline.com/loi/tprs20

A single machine scheduling problem with machine availability constraints and preventive maintenance

Lu Chen, Jinfeng Wang & Wenhui Yang

To cite this article: Lu Chen, Jinfeng Wang & Wenhui Yang (2020): A single machine scheduling problem with machine availability constraints and preventive maintenance, International Journal of Production Research, DOI: 10.1080/00207543.2020.1737336

To link to this article: https://doi.org/10.1080/00207543.2020.1737336

Published online: 11 Mar 2020.

International Journal of Production Research, 2020
https://doi.org/10.1080/00207543.2020.1737336

A single machine scheduling problem with machine availability constraints and preventive maintenance

Lu Chen*, Jinfeng Wang and Wenhui Yang

School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, People's Republic of China

(*Received 18 May 2019; accepted 25 December 2019*)

Considering the impact of machine availability is essential in production scheduling, especially when the target is to minimise total tardiness. In this study, a single machine scheduling problem derived from a rotor production workshop is addressed. We evaluate machine availability by its reliability, which can be restored by preventive maintenance (PM). PM with different improvement effectiveness is considered in the problem formulation. A genetic algorithm (GA) is developed to solve the problem. Emmons dominance rules are applied in the mutation processes of the GA. Computational experiments demonstrate the effectiveness and efficiency of the algorithm. Sensitivity analyses have been conducted to provide useful managerial insights for real workshop scheduling.

Keywords: Single machine scheduling; total tardiness; availability; genetic algorithm; preventive maintenance

1. Introduction

In scheduling, many factors can result in machine unavailability, such as regular maintenance, tool replacements, machine breakdowns, etc. Once one of these situations occurs, the loss of production can be easily quantified. In practice, there are also invisible machine conditions due to different reasons, such as loose linkage, fatigue of parts, misalignment of tighteners, etc. Cassady and Kutanoglu (2005) proved that considering machine degradation conditions in production scheduling led to savings in total tardiness. Moreover, Pandey, Kulkarni, and Vrat (2011) found that machine conditions affected production scheduling in terms of completion time and processing quality.

Over the past few decades, there has been no uniform manner for quantifying the impact of machine condition on availability. Pan, Liao, and Xi (2010), Cui, Lu, and Pan (2014) and Salmasnia and Mirabadi-Dastjerd (2017) defined the state of the machine with dummy age. Wang and Xia (2007), Zhao and Tang (2012) and Strusevich and Rustogi (2016) used deteriorating processing time to represent the effects of the machine condition on scheduling. Measuring machine condition with reliability was widely used in works of literature (Liao and Juan 2007; Lu, Cui, and Han 2015; Wang et al. 2017). A machine with low reliability may result in lower product quality and consequently, loss of time. When machine condition deteriorates, some active or passive unavailable intervals are necessary to restore the machine to a normal state.

A single machine scheduling problem is studied in this paper. This work is derived from a real rotor production workshop in Shanghai Electric, where the production process requires high reliability of equipment. The factory runs under make-to-order mode, and maintenance is very often delayed due to high pressure on production. As a result, machine breakdown rate is at a high level. Statistics show that the loss of production for one machine is averagely 1000 hours per year due to unexpected breakdown. The objective of this study is to minimise the total tardiness time, with the consideration of machine availability constraint.

The remainder of the paper is organised as follows. Section 2 reviews the related works of literature. The decision problem is formulated as a mixed-integer program in Section 3. Sections 4 presents a genetic algorithm (GA). Section 5 analyses the computational results and derives managerial insights by applying sensitivity analyses. Section 6 provides concluding remarks and perspectives on future research.

2. Literature review

Scheduling with machine availability constraint has attracted much attention in recent decades. Current literature focuses on scheduling with different types of unavailable intervals. In this section, we first review scheduling with single unavailable

*Corresponding author. Email: chenlu@sjtu.edu.cn

interval and periodic unavailable intervals, then discuss scheduling with flexible unavailable intervals. They are followed by a description of our scientific contributions.

2.1. Scheduling with single unavailable interval

In earlier studies, researchers assumed that machines had only one unavailable interval. The pioneering research on scheduling problem with one fixed unavailable period is conducted by Lee (1996). Gunter (2000) proved that such problem was NP-hard. They assumed that the duration and starting time of the interval were known in advance. Kubzin and Strusevich (2006) considered that the duration of the unavailable interval depended on its starting time and had a controllable length. In practice, having only one unavailable interval during the scheduling horizon is idealistic.

2.2. Scheduling with periodic unavailable intervals

Maintenance is the major reason that results in machine unavailability. Time-based maintenance and condition-based maintenance are the two classical maintenance policies.

With time-based maintenance, machines have periodic unavailability. Ji, He, and Cheng (2007) applied the longest processing time (LPT) rule to solve a single machine scheduling problem with periodic maintenance. The objective is to minimise the makespan. Low et al. (2010) dealt with a single machine scheduling problem with flexible and periodic availability constraints. Yu, Zhang, and Steiner (2014) compared the performances of three algorithms to solve a single machine scheduling problem with periodic maintenance. The three algorithms are list scheduling (LS), first LPT and modified longest-processing-time first (MLPT). It was proved that the three algorithms had the same worst-case ratio and same computational complexity.

2.3. Scheduling with flexible unavailable intervals

Ahmadi and Newby (2011) concluded that the condition-based maintenance policy was more realistic. This led to the research focusing on flexible unavailability intervals. Dieulle et al. (2003) considered that the machine failed when its condition dropped below a pre-set critical threshold. The objective was to minimise the total cost, including inspection cost, preventive replacement cost, corrective replacement cost, and cost of inactivity. Cassady and Kutanoglu (2003) proposed an integrated model that simultaneously determines production scheduling and preventive maintenance (PM) planning decisions so that the total weighted tardiness of jobs is minimised. Sbihi and Varnier (2008) considered a single machine scheduling problem with several maintenance periods to minimise the maximum tardiness. Specifically, the periodic maintenance and the flexible maintenance were investigated. Rebai, Kacem, and Adjallah (2012) aimed to minimise the PM cost such that a set of PM tasks had to be continuously run during the schedule horizon on M machines. Yildirim and Nezami (2014) proposed a single machine production planning model to minimise overall system cost. PM was considered when the reliability was lower than the threshold. In addition, the processing time of jobs varied according to machine condition.

Much less attention has been attracted to the production scheduling problem considering multiple levels of maintenance. Chen, Xiao, and Zhang (2015) suggested three levels of maintenance in the single machine scheduling problem: (1) minimal repair undertaken upon failure to restore the machine ‘as bad as old’; (2) replacement to make the machine ‘as good as new’; and (3) PM to make the machine less deteriorated. Mokhtari, Mozdgir, and Kamal-Abadi (2012) investigated a parallel machine scheduling problem with condition-based maintenance. Multiple PM actions were considered. A population-based variable neighbourhood search algorithm was developed to solve the problem.

Not many studies have taken into account machine failure. Salmasnia and Mirabadi-Dastjerd (2017) is the only work that we found to consider maintenance and machine failure at the same time. They presented an integrated mathematical model for production scheduling and PM planning for a single machine. The objective was to optimise weighted completion time and maintenance cost simultaneously.

Besides maintenance, there exists other situations that may lead to machine unavailability. Rubén and Carlos (2011) investigated the unrelated parallel machine problem with resource-assignable sequence and machine-dependent set-up times. Machines were not available during set-up times. The objective function is to minimise makespan and the number of human resources assigned to the set-up. Joo and Kim (2015) studied a parallel machine scheduling problem with set time and production availability constraints, where production availability refers to the physical limitation of machines. For example, some jobs cannot be processed on a specific machine. The objective function is also to minimise makespan.

2.4. Summary and contributions of the paper

In the context of scheduling with availability constraints, most pervious studies consider either available or unavailable status of machines. In our paper, we measure machine condition by its reliability, based on which machine availability is defined. The major contributions of our paper are:

(1) The effectiveness of different types of PM in terms of avoiding the potential risk of a machine breakdown, and their impact on production availability are formulated at the same time.
(2) A GA is developed to solve the problem. The classical Emmons dominance rules are applied to narrow the search space and improve the efficiency of the algorithm.
(3) The effectiveness and the efficiency of our approach are illustrated using instances generated from a real production workshop. The feasibility of applying our approach in different scenarios is also analysed.

3. Mathematical model

Before formulating the problem, we first define machine reliability and different levels of PM.

3.1. Machine reliability

We assume that machine failure function follows Weibull probability distribution, where the failure rate is proportional to a power of time. The Weibull distribution is flexible to characterise failure distributions in all areas of the bathtub curve. The machine addressed in this paper is to wear-out period with an increasing failure rate. Let u be the dummy age of the machine, and $\rho(u)$ be the rate of occurrence of failure. According to the power-law process (Ascher and Feingold 1986):

$$\rho(u) = \lambda\beta u^{\beta-1} \tag{1}$$

where λ and β are the scale and shape parameters of the system, respectively.

Thus, when $\beta > 1$, the failure rate is increasing, which is indicative of a wear out failure. The reliability of the machine during a given time period $[u_1,u_2]$ R_u (i.e. the probability of no failure over the period) is defined as follows:

$$R_u = e^{-\int_{u_1}^{u_2} \rho(u)\,\mathrm{d}u} = e^{-\lambda[(u_2)^\beta-(u_1)^\beta]} \tag{2}$$

Two different levels of PM are considered. They are imperfect preventive maintenance (IPM) and perfect preventive maintenance (PPM). IPM only reduces machine dummy age to some extent, while PPM reduces machine dummy age to 0. Improvement factor is used to define the proportion of machine dummy age reduction after an IPM (Salmasnia and Mirabadi-Dastjerd 2017).

Figure 1 illustrates the evolvement of machine reliability with its dummy age. After the processing of each job, the reliability degrades with the increase of the accumulated processing time. And curve descent varies with the processing time of different jobs. Once the reliability is expected to drop below the threshold (i.e. before processing job 4 and job 7), a PM will be triggered to restore the machine condition. Thus, a decision has to be made to select either an IPM or a PPM. A trade-off exists between time spent in maintenance and the benefits of improving machine reliability. The objective is to minimise the total tardiness.

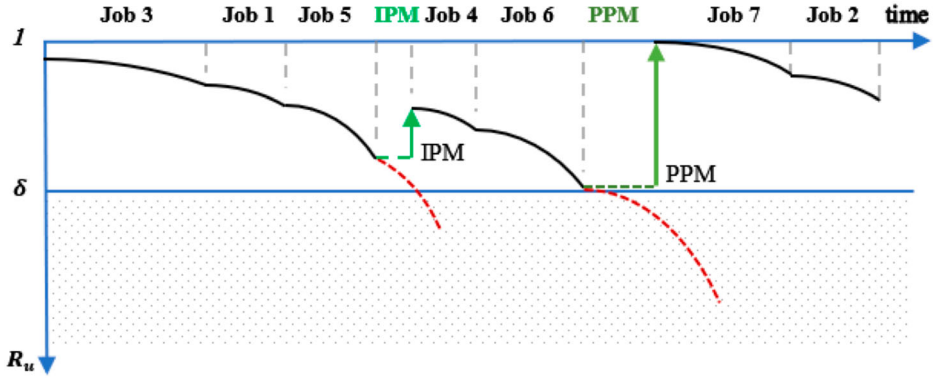

Figure 1. Evolvement of machine reliability.

3.2. Problem formulation

The single machine scheduling problem is formally described as follows. A set of jobs (denoted as N) needs to be processed on a single machine with its reliability identified. IPM or PPM can be inserted between every two adjacent jobs to keep the reliability of the machine at an acceptable level δ. The scheduling problem is to sequence the jobs on the machine to minimise the total tardiness. Some assumptions are made:

(1) All jobs are available for processing at time zero.
(2) Machine's dummy age is assumed to be zero before start.
(3) The set-up time between the two jobs is ignored.
(4) In the case of any failure, processing of a job will resume after repair with no extra processing time incurred.

Notations are defined as follows:

M	A large positive value.
i	Job index, $i \in N$.
j	Position index, $j \in N$.
p_i	Processing time of job i, $i \in N$.
d_i	Due date of job i, $i \in N$.
T_{IPM}	Duration time of an IPM.
T_{PPM}	Duration time of a PPM.
T_F	Penalty time of failure. It is a measurement of potential risk of machine breakdown.
u_{bj}	Machine's dummy age before the job in position j is processed, $j \in N$.
u_{fj}	Machine's dummy age after the job in position j is processed, $j \in N$.
θ	Improvement factor of the IPM, $\theta \in (0,1)$.
R_j	Machine reliability after the job in position j is processed, $j \in N$.
F_j	Expected cumulative number of failures at position j, $j \in N$.
δ	Reliability threshold.

Based on the optimal PM interval (T^*) given by Ebeling (2008), the threshold of machine reliability is defined as

$$\delta = e^{-\lambda \cdot (T^*)^\beta} = e^{-\frac{T_{\mathrm{PPM}}}{T_F(\beta-1)}} \tag{3}$$

The expected cumulative number of failures at position j is

$$F_j = \int_{u_{b1}}^{u_{fj}} \rho(u)\,\mathrm{d}u = \int_{u_{b1}}^{u_{fj}} \lambda\beta u^{\beta-1}\,\mathrm{d}u = \lambda[(u_{fj})^\beta - (u_{b1})^\beta] \tag{4}$$

where $u_{b1} = 0$.

The decision variables are:

x_{ij}	$= 1$, if job i is processed in position j; 0, otherwise. $i, j \in N$.
y_j	$= 1$, if an IPM is inserted at the beginning of position j before the job is processed; 0, otherwise. $j \in N$.
z_j	$= 1$, if an PPM is inserted at the beginning of position j before the job is processed; 0, otherwise. $j \in N$.
s_j	Start time of the machine in position j, $j \in N$.
c_i	Completion time of job i, $i \in N$.

The mathematical model is formulated as

$$\text{Minimise} \quad \sum_{i \in N} \max\{c_i - d_i, 0\} \tag{5}$$

subject to

$$\sum_{i \in N} x_{ij} = 1, \quad \forall j \in N \tag{6}$$

$$\sum_{j \in N} x_{ij} = 1, \quad \forall i \in N \tag{7}$$

$$s_{j+1} \geq x_{ij} \cdot c_i, \quad \forall i,j \in N \tag{8}$$

$$c_i + M(1 - x_{ij}) \geq s_j + p_i + T_{\text{IPM}} \cdot y_j + T_{\text{PPM}} \cdot z_j + T_F \cdot (F_j - F_{j-1}), \quad \forall i,j \in N \tag{9}$$

$$u_{bj} = u_{fj-1}(1 - \theta \cdot y_j - z_j), \quad \forall j \in N \tag{10}$$

$$u_{fj} \geq u_{bj} + x_{ij} \cdot p_i, \quad \forall j \in N \tag{11}$$

$$R_j = e^{-\lambda[(u_{fj})^\beta - (u_{bj})^\beta]}, \quad \forall j \in N \tag{12}$$

$$R_j \geq \delta, \quad \forall j \in N \tag{13}$$

$$y_j + z_j \leq 1, \quad \forall j \in N \tag{14}$$

$$x_{ij}, y_j, z_j \in \{0,1\}, \quad s_j, c_i \geq 0, \quad \forall i,j \in N \tag{15}$$

The objective function (5) minimises the total tardiness. Constraints (6) and (7) ensure that each position can receive only one job, and each job can be assigned to one position. Constraints (8) define that the relationship between the completion time of the job and the start time of the subsequent position, and $s_1 = 0$. Constraints (9) calculate the time between two adjacent positions, which includes job processing time, PM time, and penalty time of machine breakdown. Constraints (10) and (11) calculate the dummy age of the machine, where $u_{f0} = 0$. Constraints (12) calculate the reliability of the machine after the job in position j is processed. Constraints (13) ensure that the reliability of the machine is greater than the threshold at each position. Constraints (14) ensure that only one type of PM can be selected for each position. The domains of the decision variables are defined in constraints (15).

Since the commercial solvers cannot obtain optimal solutions for the real problems in a limited time, a GA is proposed.

4. Genetic algorithm

In this section, a GA is developed to solve the problem. The basic elements of the GA are described in the following sub-sections.

4.1. Encoding and decoding

The chromosome consists of indices of jobs. Suppose there are seven jobs to be processed on the machine. Figure 2 shows the encoding chromosome for the sample scheduling problem.

Based on the sequence of jobs represented by the chromosome, a complete schedule can be constructed by inserting PM as described in the decoding algorithm.

In Line 3, the reliability of the machine in each position is calculated. If a maintenance is needed, an IPM is first selected in Line 7. If the reliability could not be restored above the threshold, then a PPM is selected instead of an IPM. Prioritising IPM over PPM is based on the observation of real practice in the workshop, where maintenance with a shorter pause of production is always more preferable.

The complete schedule for the sample chromosome shown in Figure 2 is thus constructed as follows. From the second position, the reliability is calculated to check if a maintenance is needed to keep the reliability above δ. For the forth position, $R_4 < \delta$ and an IPM is sufficient to restore the machine reliability above δ. Thus, an IPM is conducted before processing job 4. For the seventh position, $R_7 < \delta$ and an IPM is not sufficient to restore the machine reliability above δ. Thus, a PPM is conducted before processing job 2. The complete schedule is illustrated in Figure 3.

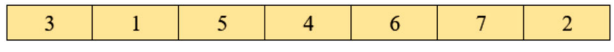

Figure 2. Chromosome representation in the GA.

Figure 3. A complete schedule for the sample problem.

Algorithm 1 Decoding

1. **For** $j = 2$ to $|N|$
2. **do**
3. Calculate R_j as Equation (12)
4. **If** $(R_j \geq \delta)$
5. $y_j = z_j = 0$
6. **Else**
7. Set $y_j = 1$, and calculate R_j
8. **If** $(R_j \geq \delta)$
9. $y_j = 1$, and $z_j = 0$
10. **Else**
11. $y_j = 0$, and $z_j = 1$
12. **End if**
13. **End if**
14. **End for**
15. **Return** $\{y_j, z_j\}$

4.2. Crossover and mutation

Fitness function is used to determine which chromosome will survive and be reproduced to the next generation. The greater the fitness, the higher the probability of survival. Therefore, fitness function f is defined as

$$f = \begin{cases} 1/\sum_{i \in N} \max\{c_i - d_i, 0\}, & \text{if } c_i - d_i > 0, \forall i \in N \\ 1, & \text{if } c_i - d_i \leq 0, \forall i \in N \end{cases}$$

where $\sum_{i \in N} \max\{c_i - d_i, 0\}$ is the objective function value defined in model.

Roulette wheel selection (Goldberg 1989) is used to select the parent chromosomes, and the two-point crossover operator (Sortrakul, Nachtmann, and Cassady 2005) is applied to generate offspring.

As for mutation operations, the classical Emmons dominance rules (Emmons 1969) are used to establish priority relationships between jobs, thereby narrowing the search space (Kanet 2007).

Let B_i and A_i be the sets of jobs that come before and after job i, respectively (written as $B_i \to i$ and $i \to A_i$). Let $j \prec k$ denote one of the following two situations: (1) $p_j < p_k$; or (2) $p_j = p_k$ and $d_j \leq d_k$.

The Emmons rules are given as follows:

THEOREM 1 *For any two jobs j and k with $j \prec k$, if $d_j \leq \max\left(\sum_{i \in B_k} p_i + p_k, d_k\right)$, then $j \to k$ in an optimal solution.*

THEOREM 2 *For any two jobs j and k with $j \prec k$, if* (1) $d_j > \max\left(\sum_{i \in B_k} p_i + p_k, d_k\right)$, *and* (2) $d_j + p_j \geq \sum_{i \in A_k'} p_i$, *where* $A_k' = \{i : i \notin A_k\}$, *then $k \to j$ in an optimal solution.*

The above theorems give sufficient conditions for job precedence relationship. Mutation operation is performed with these precedence dominance rules respected. Define N_{pop} as the population size and p_m as the mutation probability. Only when a random number generated between 0 and 1 is less than p_m, is the mutation operation activated. First, randomly select two jobs satisfying $j \prec k$. Then, determine whether their processing order is consistent with the order derived from Emmons rules. If not, swap their locations in the chromosome. The detail of the mutation operation is described in Algorithm 2.

4.3. The genetic algorithm

A majority (90%) of the initial population is generated randomly. The rest part of the initial population is generated using the earliest due date rule. Preliminary tests proved that this initialisation ensures the diversity of the initial population, and improves the convergence speed of the GA.

The crossover and mutation operators in each generation may alter the best chromosomes in the parent. In order to maintain the best chromosomes at present, we replace the worst individual in the new population with the best one, to ensure that the elite chromosomes are not lost.

Algorithm 2 Mutation operation

1. **For** $i = 1$ to N_{pop}
2. **do**
3. **If** $(random(0,1) > p_m)$
4. Return
5. **Else**
6. Randomly select two jobs j and k satisfying $j \prec k$
7. **If** $j \rightarrow k$
8. **If** $d_j \leq max(\sum_{i \in B_k} p_i + p_k, d_k)$
9. Return
10. **Elseif** $d_j + p_j \geq \sum_{i \in A_k'} p_i$
11. Swap job j and job k in the chromosome
12. **End if**
13. **Else**
14. **If** $d_j \leq max(\sum_{i \in B_k} p_i + p_k, d_k)$
15. Swap job j and job k in the chromosome
16. **Elseif** $d_j + p_j \geq \sum_{i \in A_k'} p_i$
17. Return
18. **End if**
19. **End if**
20. $i++$
21. **End for**

Algorithm 3 The GA

1. Set parameters N_{pop}, G_{max}, p_c, and p_m.
2. Generate the first population.
3. **while**$(g \leq G_{max})$
4. **do**
5. Select parents based on roulette wheel and remove the parents from the current population.
6. Perform crossover operation for the parents and get two children.
7. Perform mutation operation for the two children and insert the children into the current population.
8. Perform decoding for the population.
9. Calculate fitness for each individual.
10. Find ω and π.
11. Replace ω with the best individual π.
12. Update π^* with the best individual π.
13. $g++$.
14. **End while**
15. **Return** π^*.

Define the notations used in algorithms as follows:

g	Current generation.
N_{pop}	Size of the population.
G_{max}	Maximum generation.
p_c	Crossover probability.
π^*	The best individual.
π	The best individual in the current generation.
ω	The worst individual in the current generation.

The complete algorithm is summarised in Algorithm 3.

5. Computational experiments

We conducted a series of experiments to evaluate the performance of the GA. The experiments were performed on a personal computer with an Intel(R) Core(TM) i5-7400U(2.5GHZ) CPU and 8GB RAM. The GA was implemented in C++.

Table 1. Parameters setting for instances generation.

Parameters	Distribution form or value
p_i	$U[20, 30]$
d_i	$U[(1 - T - R/2) * \sum_{i\in N} p_i, (1 - T + R/2) * \sum_{i\in N} p_i]$
T	0.1
R	0.5
λ	10^{-6}
β	3
T_{IPM}	2h
T_{IPM}	5h
T_F	10h
θ	0.4
δ	$e^{-\frac{T_{\text{PPM}}}{T_F(\beta-1)}} = 0.78$

Table 2. Comparison of GA and optimal solution.

	Lingo		GA		Dev
$\|N\|$	$\text{Obj}_{\text{Lingo}}$ (h)	CPU (s)	Obj_{GA} (h)	CPU(s)	(%)
3	1.13	1.20	1.13	1.58	0
4	4.67	12.85	4.67	1.73	0
5	8.89	51.44	8.89	1.71	0
6	6.38	179.81	6.38	1.73	0
7	2.12	1289.43	2.12	1.77	0
8	8.85	5400	8.71	2.13	1.58
9	6.22	5400	5.93	2.45	4.66
10	3.91	5400	3.62	2.60	7.42

5.1. Problem settings

There is a series of processes in the rotor production workshop. We selected the CNS boring machine in our experiments. Different sizes of instances were randomly generated based on the information we had obtained from the factory. For a given rotor i, its processing time p_i is obtained from a discrete uniform distribution on [20, 30] (unit: *hour*). Due date d_i is generated by two parameters T and R controlling, respectively, tightness and range of due dates. Specifically, d_i is obtained from a discrete uniform distribution on $[(1 - T - R/2) * \sum_{i\in N} p_i, (1 - T + R/2) * \sum_{i\in N} p_i]$. Table 1 summarises the parameters.

5.2. Small instances

LINGO 18.0 solver is used to solve small instances to optimum. Some preliminary experiments on the parameter calibration for GA were conducted. We set the parameters as follows: crossover probability $p_c = 0.8$, mutation probability $p_m = 0.1$, maximum generation $G_{\max} = 200$, and population size $N_{\text{pop}} = 50$.

Our first set of experiments compares the GA solutions with the optimal solutions obtained from Lingo. We consider small instances with up to 10 jobs. For each size, 10 instances were generated randomly.

Table 2 compares the performance of the GA with the optimal solutions. The average objective function value and the average CPU time in seconds for both methods are given in the table. We retrieve the solution from the Lingo solver when the computational time reaches the limit of 5400 seconds. The last column shows the deviation between solutions obtained from GA and those from Lingo. It is calculated as:

$$\text{Dev} = \begin{cases} \dfrac{\text{Obj}_{\text{Lingo}} - \text{Obj}_{\text{GA}}}{\text{Obj}_{\text{Lingo}}} \times 100\%, & \text{Obj}_{\text{Lingo}} > 0 \\ (\text{Obj}_{\text{Lingo}} - \text{Obj}_{\text{GA}}) \times 100\%, & \text{Obj}_{\text{Lingo}} = 0 \end{cases}$$

where $\text{Obj}_{\text{Lingo}}$ is the average objective value obtained from Lingo, and Obj_{GA} is the average objective value obtained from the GA.

Table 3. Performance of the GA on medium and large instances.

	GA		GA-IPM			GA-PPM		
$\|N\|$	Obj_{TPM} (h)	CPU (s)	Obj_{IPM} (h)	CPU (s)	Dev_{IPM} (%)	Obj_{PPM} (h)	CPU (s)	Dev_{PPM} (%)
20	8	3.21	21	3.01	162.5	14	2.87	75.00
30	17	4.64	36	4.08	111.76	26	3.91	52.94
40	94	5.88	140	5.04	48.94	137	4.86	45.74
50	237	6.72	335	6.35	41.35	308	5.79	29.96
60	295	6.94	407	6.94	37.97	320	6.93	8.47
70	470	8.35	603	8.14	28.30	499	7.31	6.17
80	711	9.21	924	9.41	29.96	742	9.45	4.36
90	1142	10.37	1458	9.88	27.76	1198	9.53	4.90
100	1445	10.28	1793	10.56	24.08	1511	10.78	4.57

From the results, the GA finds optimal solutions for all the instances up to eight jobs in much less computational time. When the number of jobs increases, Lingo solver fails to get an optimal solution within the time limit. And the GA solutions are better than the Lingo solutions by 1.58% to 7.42%.

GA is used to solve the instances in the following experiments.

5.3. *Effects of multiple levels of PM*

To evaluate the impact of considering different levels of PM on the GA's performance, we conduct a set of experiments on medium and large instances in this section. We modify our GA in two different ways:

(1) GA-IPM: Set $z_j = 0$ for each position $j, j \in N$. Thus, only IPM is considered.
(2) GA-PPM: Set $y_j = 0$ for each position $j, j \in N$. Thus, only PPM is considered.

Table 3 summarises the results. It reports the objective function value and the CPU time for the three algorithms. The solutions obtained from the original GA are used as benchmarks, and the deviations of the two modified GA are also given in the table. The deviations are calculated as:

$$\text{Dev}_{\text{IPM}} = \begin{cases} \dfrac{\text{Obj}_{\text{IPM}} - \text{Obj}_{\text{TPM}}}{\text{Obj}_{\text{TPM}}} \times 100\%, & \text{Obj}_{\text{TPM}} > 0 \\ (\text{Obj}_{\text{IPM}} - \text{Obj}_{\text{TPM}}) \times 100\%, & \text{Obj}_{\text{TPM}} = 0 \end{cases}$$

$$\text{Dev}_{\text{PPM}} = \begin{cases} \dfrac{\text{Obj}_{\text{PPM}} - \text{Obj}_{\text{TPM}}}{\text{Obj}_{\text{TPM}}} \times 100\%, & \text{Obj}_{\text{TPM}} > 0 \\ (\text{Obj}_{\text{PPM}} - \text{Obj}_{\text{TPM}}) \times 100\%, & \text{Obj}_{\text{TPM}} = 0 \end{cases}$$

where Obj_{TPM}, Obj_{IPM}, and Obj_{PPM} are the objective function value of the solution from GA, GA-IPM, and GA-PPM respectively.

Table 3 shows that GA can solve medium and large instances within seconds. We also observe that:

(1) The original GA considering multiple levels of PM outperforms the other two algorithms. The deviation of the objective function value ranges from 4.57% to 162.5%. As the problem size increases, both Dev_{IPM} and Dev_{PPM} decrease. The results demonstrate the superiority of our model to consider multiple level of PM in terms of reducing total tardiness.
(2) Comparing with GA-IPM, GA-PPM gives better solutions. In other words, it is more preferable to use PPM to restore machine condition to a better level. The time spent in maintenance can be compensated by reducing the risk of potential machine breakdown in production.

5.4. *Sensitivity analyses*

Sensitivity analyses were carried out by varying the following parameters: (1) improvement factor of IPM (θ); (2) reliability threshold (δ); and (3) due date tightness (T and R). We generated medium and large instances to analyse the impact of these parameters on the performance of the GA.

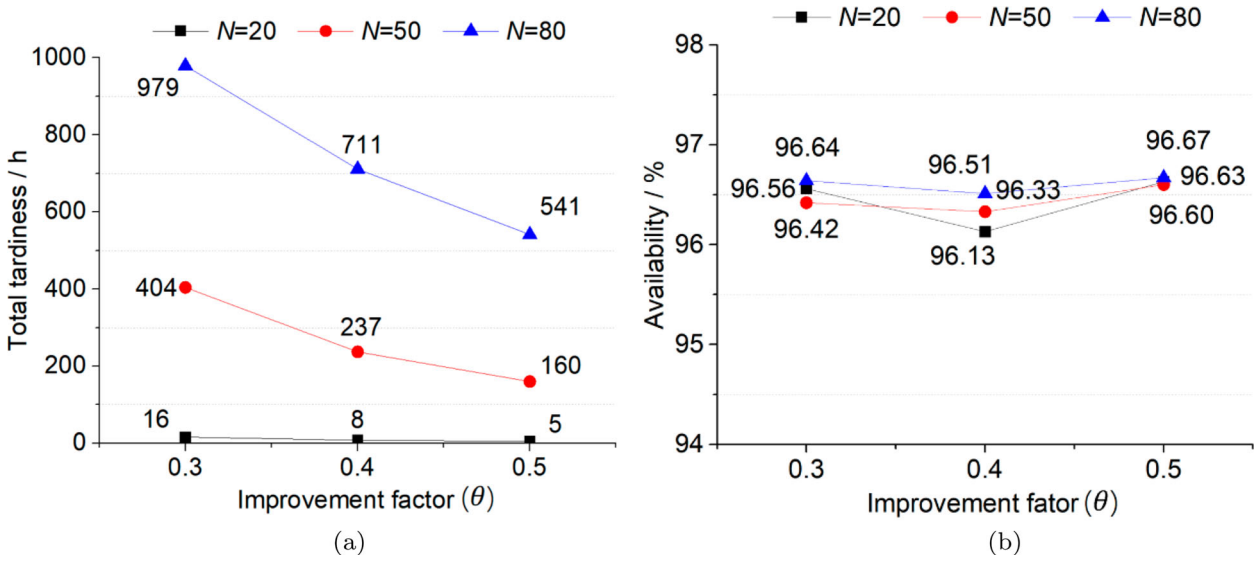

Figure 4. Sensitivity analysis to improvement factor. (a) Average total tardiness. (b) Average availability.

5.4.1. *Improvement factor of IPM*

Three levels of improvement factor are considered, namely small improvement ($\theta = 0.3$), medium improvement ($\theta = 0.4$), and large improvement ($\theta = 0.5$).The value of δ is set to 0.78 in this set of experiments.

Ten instances with different sizes were randomly generated. Figure 4(a) shows the average objective function value with different improvement factor. Figure 4(b) shows the average availability of the machine at a different level of θ. For each instance, the availability of the machine is defined as the ratio of available time to the makespan.

Figure 4 shows that:

(1) As θ increases, the objective function value decreases. And this trend becomes more apparent for larger problems. The reason may be that as θ increases, the IPM becomes more preferable and is more often adopted. Therefore, the total time spent on maintenance decreases. This helps reduce the total tardiness.
(2) With 20 jobs, the average objective function value decreases from 16 h to 5 h as θ increases from 0.3 to 0.5. With 50 jobs, the average objective function value decreases from 404 h to 160 h as θ increases from 0.3 to 0.5. With 80 jobs, the average objective function value decreases from 979 h to 541 h as θ increases from 0.3 to 0.5.
(3) The availability of the machine does not change significantly as the improvement factor increases. The reason is that the total maintenance time is similar to different improvement factor. And the expected downtime does not change since the reliability threshold is kept at a high level.

Since IPM is not as effective as PPM in terms of restoring machine's reliability, more times of maintenance may be needed. To explore its impact on the total tardiness, we compare the times of the two types of maintenance conducted with different level of θ. The results are illustrated in Figure 5.

Figure 5 shows that with the increase of improvement factor (θ), more IPM are carried out and the number of PPM decreases until 0. This tendency gets more apparent with larger problem size. The reason is that on the premise of constant IPM time (T_{IPM}), IPM becomes more cost-effective than PPM. This observation concurs with our previous conclusion drawn from Figure 4.

The managerial value of this observation is that by means of increase improvement factor of IMP, it is very useful to reduce the high pressure imposed on product delivery.

5.4.2. *Reliability threshold*

In this section, we alter the level of δ to see its impact on the GA's performance. Three levels of reliability threshold are considered, namely low level ($\delta = 0.5$), medium level ($\delta = 0.6$), and high level ($\delta = 0.78$). The maximum level of δ is the critical reliability threshold calculated by Equation (3). The value of θ is set to 0.4 in this set of experiments.

Ten instances with different sizes were randomly generated. Figure 6(a) shows the average objective function value at different level of δ. Figure 6(b) shows the average availability of the machine at different level of δ.

From Figure 6, we can see that:

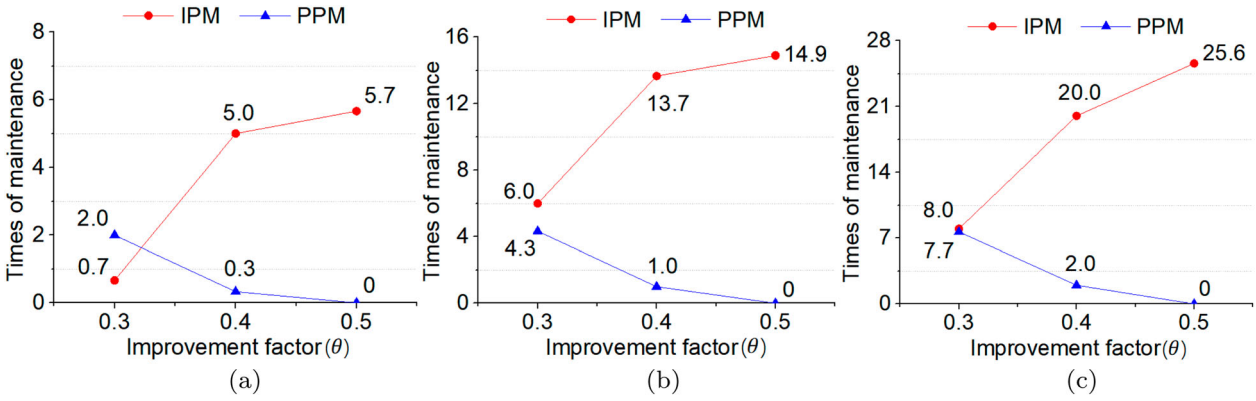

Figure 5. Times of conducting IPM and PPM with different levels of θ. (a) $N = 20$. (b) $N = 50$. (c) $N = 80$.

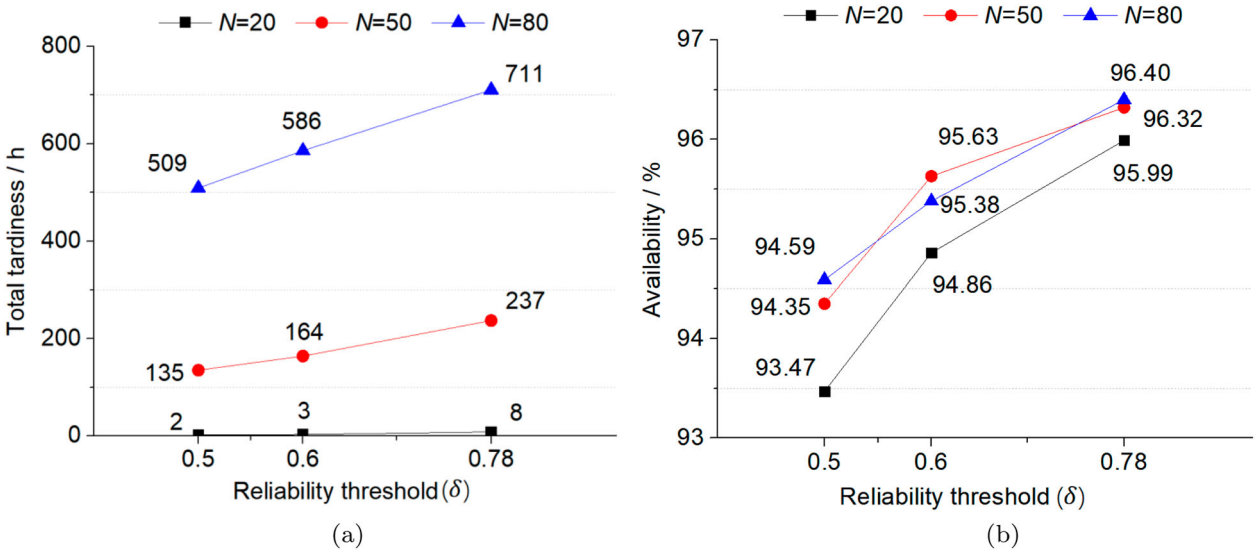

Figure 6. Sensitivity analysis to reliability threshold. (a) Average total tardiness. (b) Average availability.

(1) As shown in Figure 6(a), when δ increases, the objective function value also increases. And this trend becomes more apparent for larger problems. The reason is that as δ increases more maintenance is needed to keep the machine reliability at an acceptable level. This results in longer maintenance time and larger total tardiness.
(2) With 20 jobs, the average objective function value increases from 2 h to 8 h as δ increases from 0.5 to 0.78. With 50 jobs, the average objective function value increases from 135 h to 237 h as δ increases from 0.5 to 0.78. With 80 jobs, the average objective function value increases from 509 h to 711 h as δ increases from 0.5 to 0.78.
(3) As shown in Figure 6(b), when δ increases, the availability of the machine improves. With 20 jobs, the availability increases from 93.47% to 95.99%. With 50 jobs, the availability increases from 94.35% to 96.32%. With 80 jobs, the availability increases from 94.59% to 96.40%. From the results, we see that even though the time for maintenance increases with higher reliability threshold, the expected down time decreases. Thus, the availability of the machine improves.

In order to understand how much more maintenance time is needed and what type of maintenance is more conducted, we compare the times of conducting IPM and PPM with different level of δ. The results are illustrated in Figure 7.

From Figure 7, we can see that:

(1) As δ increases, more IPM are carried out, and the number of PPM decreases. This tendency gets more apparent with larger problem size.
(2) When δ equals 0.78, the times of IPM is much more than the times of PPM. This reveals the fact that in most of the cases, IPM is effective enough to keep the machine reliability at an acceptable level. However, this does not mean

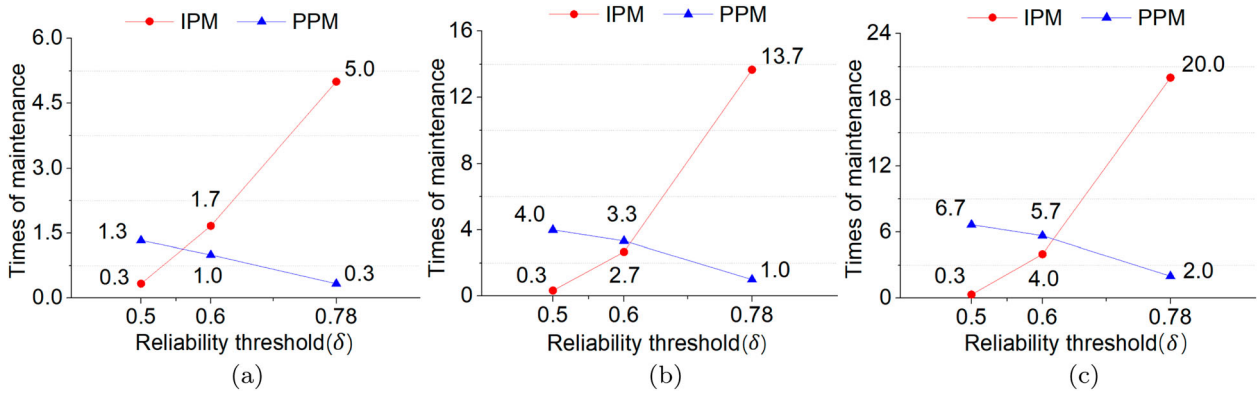

Figure 7. Times of conducting IPM and PPM with different levels of δ. (a) $N = 20$. (b) $N = 50$. (c) $N = 80$.

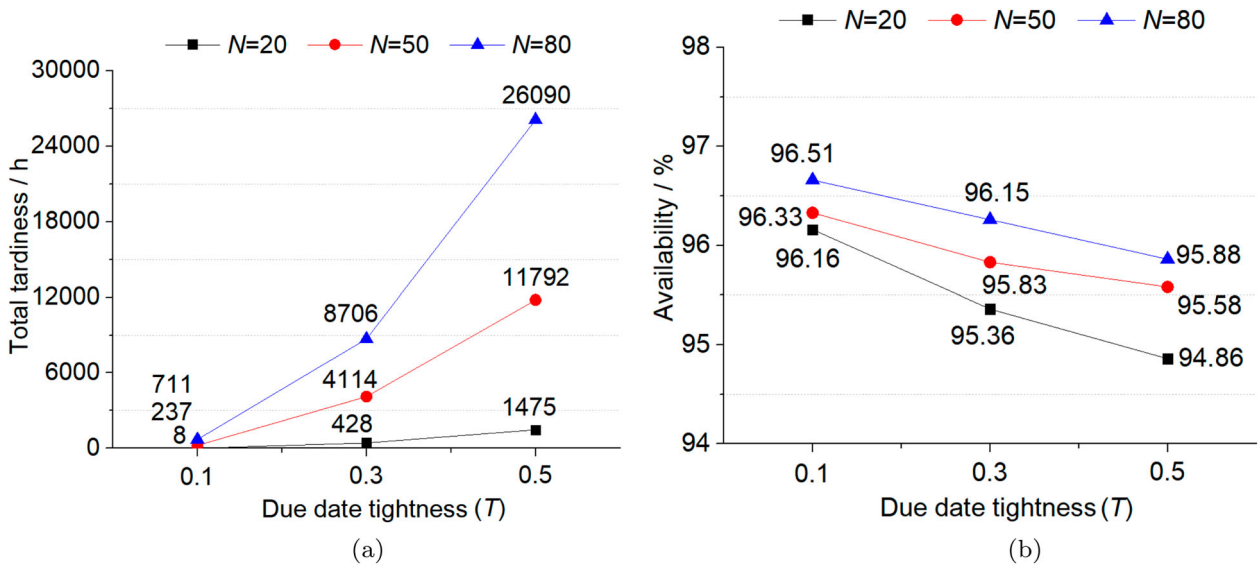

Figure 8. Sensitivity analysis to due date. (a) Average total tardiness. (b) Average availability.

that no PPM is needed. If we look again at the results in Table 3 in Section 5.3., considering only IPM will incur a larger total tardiness.

The above observations demonstrate again that considering different types of PM is beneficial to improve machine efficiency. It also concurs with the real practice in the workshop. PPM is much less carried out since it incurs a long time pause of production and often results in a delivery delay.

5.4.3. Due date tightness

Three different levels of due date are considered, namely loose due date ($T = 0.1$, $R = 0.5$), medium tight due date ($T = 0.3$, $R = 0.5$), and tight due date ($T = 0.5$, $R = 0.5$). In this set of experiments, the value of improvement factor (θ) and the reliability threshold (δ) is set to 0.4 and 0.78, respectively.

Ten instances with different sizes were randomly generated. Figure 8(a) illustrates how the average total tardiness increases as the due date gets tighter. Figure 8(b) illustrates how the average availability changes with different levels of due date.

Similarly, we compare the times of conducting IPM and PPM with different level of due date tightness in Figure 9.

From Figure 8, we can see that as due dates get tighter, the total tardiness increases dramatically and the availability of the machine decreases a little bit. However, the times of conducting both IPM and PPM do not change significantly as shown in Figure 9.

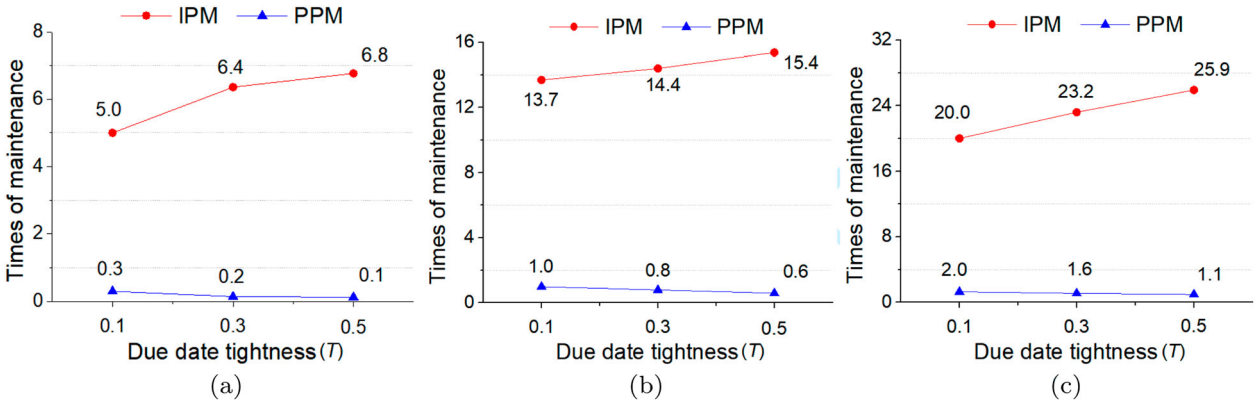

Figure 9. Times of conducting IPM and PPM with different levels of due date. (a) $N = 20$. (b) $N = 50$. (c) $N = 80$.

6. Conclusions

In this paper, a single machine scheduling problem with availability constraints in a rotor manufacturing workshop is addressed. The problem consists of determining the sequence of jobs and maintenance activities. Different types of PM with different levels of improvement factor are considered. A mathematical model is formulated to solve this problem, with the objective to minimise the total tardiness. A GA is developed to solve the problem.

Experimental results demonstrate that our model considering different types of PM outperforms those with only one type of PM, in terms of reducing total tardiness. Sensitivity analyses provide valuable information about the impact on total tardiness and machine availability of the improvement factor, reliability threshold, and due date tightness. It is observed that more IPM are carried out when its improvement factor increases. It is also interesting to find out that more IPM are carried out also when the reliability threshold increases. As for the availability of the machine, it improves as the reliability threshold increases even though more maintenance time is needed. The reason is that the expected down time decreases with higher reliability threshold.

The managerial implication is twofold: (1) the results give the managers insights into the trade-off between different types of PM and the benefits they bring; (2) it is helpful for the decision-makers to determine appropriate scheduling and maintenance strategies under different scenarios.

The proposed approach can be extended to solve the scheduling problem under more complicated environment settings (i.e. parallel machine, flow shop). Another interesting extension could be to consider maintenance with humanity and material resource constraints, thus to obtain more realistic solutions with consideration of the real situation.

Acknowledgments

We thank the referees for their valuable comments.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by National Natural Science Foundation of China (Project number: 51775347).

References

Ahmadi, R., and M. Newby. 2011. “Maintenance Scheduling of a Manufacturing System Subject to Deterioration.” *Reliability Engineering & System Safety* 96 (10): 1411–1420.

Ascher, H., and H. Feingold. 1986. “Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes.” *Microelectronics Reliability* 26 (5): 993–993.

Cassady, C. R., and E. Kutanoglu. 2003. “Minimizing Job Tardiness Using Integrated Preventive Maintenance Planning and Production Scheduling.” *IIE Transactions* 35 (6): 503–513.

Cassady, C. R., and E. Kutanoglu. 2005. “Integrating Preventive Maintenance Planning and Production Scheduling for a Single Machine.” *IEEE Transactions on Reliability* 54 (2): 304–309.

Chen, X., L. Xiao, and X. Zhang. 2015. "A Production Scheduling Problem Considering Random Failure and Imperfect Preventive Maintenance." *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 229 (1): 26–35.

Cui, W. W., Z. Lu, and E. Pan. 2014. "Integrated Production Scheduling and Maintenance Policy for Robustness in a Single Machine." *Computers & Operations Research* 47: 81–91.

Dieulle, L., C. Bérenguer, A. Grall, and M. Roussignol. 2003. "Sequential Condition-based Maintenance Scheduling for a Deteriorating System." *European Journal of Operational Research* 150 (2): 451–461.

Ebeling, C. E. 2008. *An Introduction to Reliability and Maintainability Engineering*. Long Grove, IL: Waveland Press.

Emmons, H. 1969. "One-machine Sequencing to Minimize Certain Functions of Job Tardiness." *Operations Research* 17 (4): 701–715.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA: Addison-Wesley.

Gunter, S. 2000. "Scheduling with Limited Machine Availability." *European Journal of Operational Research* 121 (1): 1–15.

Ji, M., Y. He, and T. C. E. Cheng. 2007. "Single-machine Scheduling with Periodic Maintenance to Minimize Makespan." *Computers & Operations Research* 34 (6): 1764–1770.

Joo, C. M., and B. S. Kim. 2015. "Hybrid Genetic Algorithms with Dispatching Rules for Unrelated Parallel Machine Scheduling with Setup Time and Production Availability." *Computers & Industrial Engineering* 85 (C): 102–109.

Kanet, J. J. 2007. "New Precedence Theorems for One-machine Weighted Tardiness." *Mathematics of Operations Research* 32 (3): 579–588.

Kubzin, M. A., and V. A. Strusevich. 2006. "Planning Machine Maintenance in Two-Machine Shop Scheduling." *Operations Research* 54 (4): 789–800.

Lee, C. Y. 1996. "Machine Scheduling with An Availability Constraint." *Journal of Global Optimization* 9 (3–4): 395–416.

Liao, C. J., and H. C. Juan. 2007. "An Ant Colony Optimization for Single-machine Tardiness Scheduling with Sequence-dependent Setups." *Computers & Operations Research* 34 (7): 1899–1909.

Low, C., M. Ji, C. J. Hsu, and C. T. Su. 2010. "Minimizing the Makespan in a Single Machine Scheduling Problems with Flexible and Periodic Maintenance." *Applied Mathematical Modelling* 34 (7): 1899–1909.

Lu, Z., W. W. Cui, and X. Han. 2015. "Integrated Production and Preventive Maintenance Scheduling for a Single Machine with Failure Uncertainty." *Computers & Industrial Engineering* 80: 236–244.

Mokhtari, H., A. Mozdgir, and I. N. Kamal-Abadi. 2012. "A Reliability/availability Approach to Joint Production and Maintenance Scheduling with Multiple Preventive Maintenance Services." *International Journal of Production Research* 50 (20): 5906–5925.

Pan, E., W. Liao, and L. Xi. 2010. "Single-machine-based Production Scheduling Model Integrated Preventive Maintenance Planning." *The International Journal of Advanced Manufacturing Technology* 50 (1–4): 365–375.

Pandey, D., M. S. Kulkarni, and P. Vrat. 2011. "A Methodology for Joint Optimization for Maintenance Planning, Process Quality and Production Scheduling." *Computers & Industrial Engineering* 61 (4): 1098–1106.

Rebai, M., I. Kacem, and H. K. Adjallah. 2012. *Earliness-tardiness Minimization on a Single Machine to Schedule Preventive Maintenance Tasks: Metaheuristic and Exact Methods*. New York: Springer-Verlag.

Rubén, R., and A. R. Carlos. 2011. "Scheduling Unrelated Parallel Machines with Resource-assignable Sequence-dependent Setup Times." *International Journal of Advanced Manufacturing Technology* 57 (5–8): 777–794.

Salmasnia, A., and D. Mirabadi-Dastjerd. 2017. "Joint Production and Preventive Maintenance Scheduling for a Single Degraded Machine by Considering Machine Failures." *TOP* 50 (1–4): 365–375.

Sbihi, M., and C. Varnier. 2008. "Single-machine Scheduling with Periodic and Flexible Periodic Maintenance to Minimize Maximum Tardiness." *Computers & Industrial Engineering* 55 (4): 830–840.

Sortrakul, N., H. L. Nachtmann, and C. R. Cassady. 2005. "Genetic Algorithms for Integrated Preventive Maintenance Planning and Production Scheduling for a Single Machine." *Computers in Industrial* 56 (2): 161–168.

Strusevich, V. A., and K. Rustogi. 2016. *Scheduling with Time-Changing Effects and Rate-Modifying Activities*. Heidelberg, Germany: Springer International Publishing.

Wang, J. B., and Z. Q. Xia. 2007. "Single Machine Scheduling Problems with Controllable Processing Times and Total Absolute Differences Penalties." *European Journal of Operational Research* 177 (1): 638–645.

Wang, Z., C. Xiao, X. Lin, and Y. Y. Lu. 2017. "Single Machine Total Absolute Differences Penalties Minimization Scheduling with a Deteriorating and Resource-Dependent Maintenance Activity." *The Computer Journal* 61 (1): 1–6.

Yildirim, M. B., and F. G. Nezami. 2014. "Integrated Maintenance and Production Planning with Energy Consumption and Minimal Repair." *The International Journal of Advanced Manufacturing Technology* 74 (9–12): 1419–1430.

Yu, X., Y. Zhang, and G. Steiner. 2014. "Single-machine Scheduling with Periodic Maintenance to Minimize Makespan Revisited." *Journal of Scheduling* 17 (3): 263–270.

Zhao, C., and H. Tang. 2012. "A Note to Due-window Assignment and Single Machine Scheduling with Deteriorating Jobs and a Rate-modifying Activity." *Computers & Operations Research* 39 (6): 1300–1303.