International Journal of Computational Intelligence Systems
Vol. **13(1)**, 2020, *pp.* 193–200
DOI: https://doi.org/10.2991/ijcis.d.200214.003; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

A Single Machine Scheduling with Periodic Maintenance and Uncertain Processing Time

Jiayu Shen[1,*], Yuanguo Zhu[2]

[1] *Department of Public Basic Courses, Nanjing Institute of Industry Technology, Nanjing 210023, Jiangsu, People's Republic of China*
[2] *School of Science, Nanjing University of Science and Technology, Nanjing 210094, Jiangsu, People's Republic of China*

ARTICLE INFO

Article History
Received 19 Apr 2019
Accepted 13 Feb 2020

Keywords
Single machine
Periodic maintenance
Uncertainty
Chance-constrained
Genetic algorithm

ABSTRACT

A single machine scheduling problem with periodic maintenance is studied in this paper. Due to many uncertainties in reality, the processing time is recognized as an uncertain variable. The aim is to minimize the makespan at a confidence level. An uncertain chance-constrained programming model is developed to delve into the impact of uncertainties on decision variables, and an algorithm for calculating the objective function is proposed. According to the theoretical analysis, a novel method named longest shortest processing time (LSPT) rule is proposed. Subsequently, an improved genetic algorithm (GA-M) combined with LSPT rule is proposed. Numerical experiments are used to verify the feasibility of this model and algorithm.

1. INTRODUCTION

Most studies on scheduling problems assume that the machine is always available. However, this assumption does not hold in reality. If the machine runs constantly, it is prone to failure. To reduce the wastage and expand life expectancy of a machine, preventive maintenance activities are essential. During the process of maintenance, jobs need to stop and wait. This is due to a lack of coordination between job scheduling and machine maintenance. The purpose of this study is to coordinate their relationship.

The machine maintenance is often treated as an unavailability interval in the literature. Lee [1] studied several scheduling problems where each machine has only one unavailability interval, and carried out theoretical analysis under each scenario. Low *et al.* [2] studied a single machine scheduling problems with availability constraints, and a flexible model and heuristic algorithms were proposed. Lee and Kim [3] considered a single machine scheduling problem with periodic maintenance and a two-phase heuristic algorithm modified from Moore's algorithm was developed. A single machine scheduling problem with periodic maintenance was studied by Benmansour *et al.* [4], and two mixed integer linear models were developed. Four single-machine scheduling problems with a variable machine maintenance were investigated by Ying *et al.* [5] and polynomial-time exact algorithms were proposed. A single machine scheduling problem with flexible periodic maintenances was studied by Cui and Lu [6], and an effective heuristic was developed to solve the non-resumable case. Two machine-related periodic maintenance scheduling models were established by Li *et al.* [7], and two improved heuristic algorithms were proposed. A single machine scheduling problem with periodic maintenance and sequence-dependent set-up times was analyzed by Pacheco *et al.* [8], and an efficient variable neighborhood search approach with memory was proposed. Nesello *et al.* [9] proposed new arc-time index formulations for scheduling with periodic maintenances and given a simple iterative exact algorithm. Perez-Gonzalez and Framinan [10] developed single machine scheduling models with cyclical machine availability periods, and proposed new heuristics based on bin packing assignment rules. Wang *et al.* [11] studied a single machine scheduling problem with deteriorating jobs and flexible periodic maintenance. Two integer programming formulations and a branch-and-price algorithm were proposed.

Most of the mentioned scheduling problems were investigated in a deterministic scenario. Since human behaviors are involved in the process, the decision maker has to take into account all of the possible events during the planning horizon. Frequency is a factual property of indeterminate quantity and does not change with our state of knowledge and preference. When the sample size is large enough and no emergency (e.g. war, flood, earthquake, accident and even rumor) arises, it is likely for us to find a distribution function that is sufficiently close enough to the frequency. In this case, the probability theory is undoubtedly the only legitimate method to address our problem. But, when statistics are unreliable or unavailable, probability theory is not the best choice. Due to the absence of data for these variables, experts should be invited to assess the belief degree that an uncertain event will occur. To deal with the involved human uncertainty, the uncertainty theory was founded by Liu [12] in 2007

* *Corresponding author. Email: fjcyue007@126.com*

and refined it in 2010 [13]. The uncertainty theory is a branch of axiomatic mathematics for modeling human uncertainty, which has been deeply developed in many fields such as uncertain programming [14–16], uncertain scheduling [17–22], uncertain risk analysis [23–25], uncertain calculus [26–28] and uncertain optimal control [29–34].

In this paper, a single machine scheduling problem with periodic maintenance and non-preemptive jobs is studied. There are many uncertainties in the real workshop environment, which cannot be described by random variables. For example, in mold processing, due to machine failures, operational errors, order changes and many other uncertainties, workers usually draw on their past experience to assess the processing time. Therefore, the processing time is considered an uncertain variable owing to the lack of historical data. To address uncertain variables in the problem, a chance-constrained model is established. The aim is to minimize the makespan at a pre-given confidence level. The equivalent form of the uncertain model is obtained in accordance with uncertainty theory. Accordingly, a new algorithm for solving the objective function is proposed. Furthermore, to improve the efficiency of solving the model, an improved genetic algorithm (GA-M) combined with longest shortest processing time (LSPT) rule is proposed. Numerical experiments are made to verify the feasibility of the proposed model and methods.

The rest of this paper is structured as follows. In Section 2, basic definitions and properties about uncertainty theory are introduced. In Section 3, we analyze the problem under an uncertain scenario and built a chance-constrained model. Besides, according to the dominating property, an algorithm is proposed to calculate the objective function. In Section 4, an evolutionary algorithm base on the genetic algorithm (GA) is proposed. In Section 5, numerical experiments are given.

2. PRELIMINARIES OF UNCERTAINTY THEORY

The uncertainty theory is founded by Liu [12] in 2007 and refined by Liu [13] in 2010 as a branch of axiomatic mathematics for modeling human uncertainty. Let Γ be a nonempty set, $\mathcal{L}$ a σ-algebra over Γ and each element Λ in $\mathcal{L}$ is called an event. Uncertain measure is defined as a function from $\mathcal{L}$ to $[0,1]$. In detail, Liu [12] gives the concept of uncertain measure as follows:

A set function $\mathcal{M}$ from $\mathcal{L}$ to $[0,1]$ is called an uncertain measure if it satisfies $\mathcal{M}\{\Gamma\} = 1$ for the universal set Γ; $\mathcal{M}\{\Lambda\}+\mathcal{M}\{\Lambda^c\} = 1$ for any event Λ; $\mathcal{M}\left\{\bigcup_{i=1}^{\infty}\Lambda_i\right\} \le \sum_{i=1}^{\infty}\mathcal{M}\{\Lambda_i\}$ for every countable sequence of events $\Lambda_1, \Lambda_2, \cdots$.

Besides, the product uncertain measure on the product σ-algebra $\mathcal{L}$ was defined by Liu [35] as follows: Let $(\Gamma_k, \mathcal{L}_k, \mathcal{M}_k)$ be uncertainty spaces for $k = 1, 2, \cdots$. The product uncertain measure $\mathcal{M}$ is an uncertain measure satisfying $\mathcal{M}\left\{\prod_{i=1}^{\infty}\Lambda_k\right\} = \bigwedge_{i=1}^{\infty}\mathcal{M}_k\{\Lambda_k\}$, where Λ_k are arbitrarily chosen events from $\mathcal{L}_k$ for $k = 1, 2, \cdots$, respectively. Uncertain measure has following two useful properties: (i) for any events $\Lambda_1 \subset \Lambda_2$, we have $\mathcal{M}\{\Lambda_1\} \le \mathcal{M}\{\Lambda_2\}$; (ii) for any events Λ_1 and Λ_2, we have

$$\mathcal{M}\{\Lambda_1\} + \mathcal{M}\{\Lambda_2\} - 1 \le \mathcal{M}\{\Lambda_1 \cap \Lambda_2\} \le \mathcal{M}\{\Lambda_1\} \wedge \mathcal{M}\{\Lambda_2\}. \quad (1)$$

The uncertain distribution Φ of an uncertain variable ξ is defined by $\Phi(x) = \mathcal{M}\{\xi \le x\}$ for any real number x. Let ξ be an uncertain variable with continuous uncertainty distribution Φ. Then for any real number x, we have $\mathcal{M}\{\xi \le x\} = \Phi(x)$, $\mathcal{M}\{\xi \ge x\} = 1 - \Phi(x)$.

The uncertain variables $\xi_1, \xi_2, \cdots, \xi_m$ are said to be independent (Liu [35]) if

$$\mathcal{M}\left\{\bigcap_{i=1}^{m}(\xi_i \in B_i)\right\} = \min_{1\le i\le m}\mathcal{M}\{\xi_i \in B_i\}$$

for any Borel sets $B_1, B_2, \cdots, B_n$ of real numbers.

Example 1. An uncertain variable ξ is called normal if it has a normal uncertainty distribution

$$\Phi(x) = \left(1 + \exp\left(\frac{\pi(e-x)}{\sqrt{3}\sigma}\right)\right)^{-1}, x \in \Re$$

denoted by $\mathcal{N}(e,\sigma)$ where e and σ are real numbers with $\sigma > 0$.

Definition 1. [12] An uncertain distribution $\Phi(x)$ is said to be regular if its inverse function $\Phi^{-1}(x)$ exists and is unique for each $\alpha \in (0,1)$. Then the inverse function Φ^{-1} is called the inverse uncertainty distribution of ξ.

Example 2. The inverse uncertainty distribution of normal uncertain variable $\mathcal{N}(e,\sigma)$ is

$$\Phi^{-1}(\alpha) = e + \frac{\sigma\sqrt{3}}{\pi}\ln\frac{\alpha}{1-\alpha}.$$

Theorem 1. *[13] Let $\xi_1, \xi_2, \cdots, \xi_n$ be independent regular uncertain variables with uncertainty distributions $\Phi_1, \Phi_2, \cdots, \Phi_n$, respectively. If function $f(x_1, \cdots, x_m, x_{m+1}, \cdots, x_n)$ is strictly increasing with respect to $x_1, x_2, \cdots, x_m$, and strictly decreasing with respect to $x_{m+1}, x_{m+2}, \cdots, x_n$, then*

$$\xi = f(\xi_1, \xi_2, \cdots, \xi_n)$$

is an uncertain variable with inverse uncertainty distribution

$$\Psi^{-1}(\alpha) = f(\Phi_1^{-1}(\alpha), \cdots, \Phi_m^{-1}(\alpha), \Phi_{m+1}^{-1}(1-\alpha), \cdots, \Phi_n^{-1}(1-\alpha)).$$

Definition 2. [13] Let ξ_1 and ξ_2 be independent normal uncertain variables $\mathcal{N}(e_1,\sigma_1)$ and $\mathcal{N}(e_2,\sigma_2)$, respectively. σ_1 and σ_2 denote the standard deviation of normal uncertainty distribution. Then the sum $\xi_1 + \xi_2$ is also a normal uncertain variable $\mathcal{N}(e_1+e_2, \sigma_1+\sigma_2)$, i.e.,

$$\mathcal{N}(e_1,\sigma_1) + \mathcal{N}(e_2,\sigma_2) = \mathcal{N}(e_1+e_2, \sigma_1+\sigma_2),$$

where $\sigma_1 > 0$ and $\sigma_2 > 0$ are real numbers. The product of a normal uncertain variable $\mathcal{N}(e,\sigma)$ and a scalar number $k > 0$ is also a normal uncertain variable $\mathcal{N}(ke, k\sigma)$, i.e.,

$$k \cdot \mathcal{N}(e,\sigma) = \mathcal{N}(ke, k\sigma),$$

where σ is a real number with $\sigma > 0$ and σ denotes the standard deviation of normal uncertainty distribution.

Definition 3. [12] Let ξ be an uncertain variable, and $\alpha \in (0,1]$. Then

$$\xi_{\sup}(\alpha) = \sup\{r \mid \mathcal{M}\{\xi \geq r\} \geq \alpha\}$$

is called the optimistic value to ξ, and

$$\xi_{\inf}(\alpha) = \inf\{r \mid \mathcal{M}\{\xi \leq r\} \geq \alpha\}$$

is called the pessimistic value to ξ.

Uncertain programming is a type of mathematical programming involving uncertain variables. Assume that x is a decision vector, ξ is an uncertain vector, $f(x,\xi)$ is an objective function, and $g_j(x,\xi)$ are constraint functions, $j = 1,2,\cdots,p$. To obtain a decision with minimum objective value subject to chance constraints, Liu [36] proposed the following uncertain programming model.

$$\begin{cases} \min\limits_{x} \bar{f} \\ \text{subject to}: \\ \quad \mathcal{M}\{f(x,\xi) \leq \bar{f}\} \geq \beta \\ \quad \mathcal{M}\{g_j(x,\xi) \leq 0\} \geq \alpha_j, \qquad j = 1,2,\cdots,p, \end{cases}$$

where α_j and β are specified confidence levels for $j = 1,2,\cdots,p$, and $\min \bar{f}$ is the pessimistic value.

A key problem is how to solve the above model. Under the framework of uncertainty theory, an uncertain programming model can be transformed into a crisp one if the related functions are monotone [13].

3. MODEL UNDER UNCERTAIN ENVIRONMENT

There are n independent non-resumable jobs $J_1, J_2, \ldots, J_n$ to be processed on a single machine. The processing time of job J_i is ξ_i. Let the length of each availability interval be T and the length of each maintenance interval be t. $\max\limits_{1\leq i\leq n} \xi_i \leq T$, otherwise there is trivially no feasible schedule. Moreover, the machine is available at time zero. The processing time of a job is often considered a deterministic parameter in the literature. In fact, the processing time has a great relationship with the experience of workers who assess approximate time based on personal experience. Therefore, it is reasonable to treat the processing time of a job as an uncertain variable. Since the uncertain variables cannot be directly compared in size, we rank them at a confidence level. $\xi \leq \eta$ if and only if $\mathcal{M}\{\xi \leq \eta\} \geq \alpha$ for two uncertain variables ξ, η and a confidence level α. According to the definition of uncertainty distribution, we have $\Phi_{\xi-\eta}(0) \geq \alpha$, where $\Phi_{\xi-\eta}$ denotes the uncertainty distribution of $\xi - \eta$. Without loss of generality, assume that M availability intervals are used. The objective function is

$$f = (M-1)(T+t) + Sum_M, \tag{2}$$

where Sum_M denotes the total processing time of jobs at the Mth availability interval.

In practice, the decision maker always consider the impact of risks and emergencies, and find a boundary as a reference for the optimal schedule. In this scenario, the concept of chance-constrained [37] can be adopted to address this problem. The decision maker has to assess a value in advance, such that there exists an optimistic solution x satisfying $\mathcal{M}\left\{f(x) \leq \bar{f}\right\} \geq \alpha$, where $\alpha \in (0,1)$ is a predetermined confidence level. For instance, set $\alpha = 0.9$, the decision maker needs to determine a target $\bar{f}$ and search a solution x to satisfy $\mathcal{M}\left\{f(x) \leq \bar{f}\right\} \geq 0.9$. This suggests that if the decision maker takes a solution x, the target aim value should be less than $\bar{f}$ with confidence level at least 90 %.

It is noted that the aim of this paper is to minimize makespan at a pre-given confidence level, which means to find the supremum of the maximum completion time (makespan). It corresponds to the definition of pessimism value (Definition 3). Therefore, we build a chance-constrained model in the sense of pessimism.

$$\begin{cases} \min\limits_{x} \bar{f} \\ \text{subject to}: \\ \quad \mathcal{M}\{f \leq \bar{f}\} \geq \alpha, \end{cases} \tag{3}$$

where x denotes the order of jobs and it is the decision variable.

According to the definition of the pessimistic value (Definition 3), the model is equivalent to a crisp one. The equivalent objective function is:

$$\min_{x} \Psi_f^{-1}(\alpha), \tag{4}$$

where the Ψ_f^{-1} denotes the inverse uncertainty distribution of f.

According to Theorem 1, it yields

$$\Psi_f^{-1}(\alpha) = (M-1)(T+t) + \Phi_{Sum_M}^{-1}(\alpha). \tag{5}$$

Likewise, to obtain the value of $\Psi_f^{-1}(\alpha)$, we should calculate values of $\Phi_{Sum_M}^{-1}(\alpha)$ and $(M-1)(T+t)$. Note that we need to know which jobs in the last occupied availability interval and how many availability intervals are occupied. It indicates that how to obtain the value of M and which jobs in the last availability interval are two critical problems. Subsequently, an algorithm named **Algorithm 1** for the two critical problems is proposed. Since the decision variable is the order of jobs, the order of the jobs has been drawn after using heuristic algorithms. The order is $\xi_{i_1}, \xi_{i_2}, \ldots, \xi_{i_n}$.

Algorithm 1:

- Start to inspect from the first available interval. Because each availability interval can accommodate at least one job, we take turns to examine whether the following conditions can be satisfied: $\mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} \leq T\right\} \geq \alpha$, $\mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} + \xi_{i_3} \leq T\right\} \geq \alpha$, $\cdots$, $\mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} + \xi_{i_3} + \cdots + \xi_{i_n} \leq T\right\} \geq \alpha$.

  If all of the above $n-1$ conditions are satisfied, we have $M = 1$.

  If $\mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} \leq T\right\} \geq \alpha, \ldots, \mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} + \cdots + \xi_{i_j} \leq T\right\} \geq \alpha$, $\mathcal{M}\left\{\xi_{i_1} + \xi_{i_2} + \cdots + \xi_{i_j} + \xi_{i_{j+1}} \leq T\right\} < \alpha$, then assign jobs

$J_{i_1}, \ldots, J_{i_j}$ to the first available interval. It indicates that these jobs will be processed in the first available interval.

- Repeat the above operation on the remaining jobs $J_{i_{j+1}}, \ldots, J_{i_n}$ from the following available intervals until all jobs are arranged to available intervals. Finally, we can know which jobs in the last availability interval and obtain the value of M.

A dominant property for two adjacent jobs (i and j) is derived. The processing time of the two jobs are ξ_i and ξ_j. Let I and I' be two schedules where the difference between I and I' is a pairwise interchange of jobs i and j, i.e., $I = (\pi ij\pi')$ and $I' = (\pi ji\pi')$ where π and π' denote the partial sequences. Let $C_{\max}$ and C'_{max} denote the makespan of schedule I and I', respectively. If job i and j are in the same interval, the exchange of their location has no effect on the makespan. We just need to consider them in different intervals. In schedule I, job i and j are in kth and $(k+1)$th available interval, respectively. Assume that total processing time of jobs in the kth and $(k+1)$th available interval are A and B, respectively. α is a given confidence level.

Three scenarios for the relationship between job i and j are shown below.

1. $\mathcal{M}\{\xi_i > \xi_j\} \geq \alpha$. After exchanging the position of job i and j, there are two cases.
   - $\mathcal{M}\{B - \xi_j + \xi_i > T\} \geq \alpha$. Since
   $$\{C_{max} \leq C'_{max}\} \supseteq \{\xi_i > \xi_j\} \cap \{B - \xi_j + \xi_i > T\},$$
   it yields
   $$\mathcal{M}\{C_{max} \leq C'_{max}\} \geq \mathcal{M}\{\xi_i > \xi_j\} + \mathcal{M}\{B - \xi_j + \xi_i > T\} - 1 \geq 2\alpha - 1$$
   by Eq. (1).
   - $\mathcal{M}\{B - \xi_j + \xi_i \leq T\} \geq \alpha$. Since
   $$\{C_{max} = C'_{max}\} \supseteq \{\xi_i > \xi_j\} \cap \{B - \xi_j + \xi_i \leq T\},$$
   it yields $\mathcal{M}\{C_{max} = C'_{max}\} \geq 2\alpha - 1$.
2. $\mathcal{M}\{\xi_i = \xi_j\} \geq \alpha$. After exchanging the position of job i and j, it yields
   $$\mathcal{M}\{C_{max} = C'_{max}\} \geq \alpha$$
   because $\{C_{max} = C'_{max}\} \supseteq \{\xi_i = \xi_j\}$.
3. $\mathcal{M}\{\xi_i < \xi_j\} \geq \alpha$. After exchanging the position of job i and j, there are two cases.
   - $\mathcal{M}\{A - \xi_i + \xi_j > T\} \geq \alpha$, then job i and j are at the same available interval. If $\mathcal{M}\{B + \xi_i \geq T\} \geq \alpha$, and according to
   $$\{C_{max} \leq C'_{max}\} \supseteq \{\xi_i < \xi_j\} \cap \{A - \xi_i + \xi_j > T\} \cap \{B + \xi_i \geq T\}$$
   it yields $\mathcal{M}\{C_{max} \leq C'_{max}\} \geq 3\alpha - 2$.
   If $\mathcal{M}\{B + \xi_i < T\} \geq \alpha$, we have
   $\mathcal{M}\{C_{max} = C'_{max}\} \geq 3\alpha - 2$.
   - $\mathcal{M}\{A - \xi_i + \xi_j \leq T\} \geq \alpha$. Since
   $$\{C_{max} \geq C'_{max}\} \supseteq \{B - \xi_j + \xi_i \leq T\} \supseteq \{\xi_i < \xi_j\} \cap \{A - \xi_i + \xi_j \leq T\},$$
   it yields $\mathcal{M}\{C_{max} \geq C'_{max}\} \geq 2\alpha - 1$.

Based on the above discussion, a domination property under an uncertain scenario can be deduced.

Property 1. *In schedule I, for two adjacent job i and j, if* $\mathcal{M}\{\xi_i < \xi_j\} \geq \alpha$ *and* $\mathcal{M}\{A - \xi_i + \xi_j \leq T\} \geq \alpha$, *then schedule I′ is better than schedule I under confidence level* $2\alpha - 1$, *where* α *is a given confidence level.*

4. HEURISTIC METHOD

The longest processing time (LPT) rule is often used to solve the scheduling problem with minimal makespan [6,38–41]. The shortest processing time (SPT) rule is often used to solve the scheduling problem with minimal the total completion time [42–45].

The LPT and SPT rules are described as follows:

LPT: Reorder all the jobs in non-increasing order of their processing times, and process them consecutively as early as possible.

SPT: Reorder all the jobs in non-decreasing order of their processing times, and process them consecutively as early as possible.

After integrating LPT and SPT, the **LSPT** rule is described as follows:

First, we sort the jobs in descending order of processing time. Then, the jobs are arranged to the first available interval in sequence. The first two steps are to arrange the jobs according to the LPT rule. If the remaining space of the first available interval cannot arrange jobs in accordance with LPT rule, then arrange the jobs with the shortest processing time in the remaining interval of the first available interval. If there is still a free position in the first available interval, continue to arrange the jobs in ascending order of processing time. The next two steps are similar to the SPT rule. In each subsequent available interval, first use the LPT rule to arrange the jobs. If the remaining space cannot continue to arrange them, use the SPT rule. Each available interval is arranged in this way until all the jobs are arranged.

The main idea of the LSPT rule is to make the total processing time of jobs within a availability interval close to T as far as possible. Therefore, LSPT rule is better than that of LPT rule.

The GA had been widely used for solving scheduling problems in recent years [46–50]. It has been proven to be suitable for different types of scheduling problems, and various improved GAs had been proposed to speed up the convergence speed and reduce errors. In general, the initial solutions are randomly generated in GA, which reduces the execution efficiency of the algorithm. Some literature have shown that the better solution can be found quickly by seeding the initial solution with heuristic rules in an artificial intelligence algorithm [51–53]. Therefore, we use the LSPT rule and Property 1 improve accuracy and efficiency of the initial solution. An improved algorithm named **GA-M** is proposed.

Algorithm GA-M:

- **Step 1: Representation**
  The algorithm imitates the process of natural evolution in a population of chromosomes with the objective of largely preserving those individuals who are most adaptive to the environment. The chromosome is constructed based on a n character string $[a_1, a_2, \ldots, a_n]$, where a_i denotes the index of job i. It represents the sequence of jobs on the machine. This encoding means that any solution is feasible.
- **Step 2: Initialization**
  Let the population size be *pop_size*, and the crossover probability be P_c and the mutation probability be P_m. The LSPT rule is used to obtain an initial chromosome. Since the processing time of each job is an uncertain variable, jobs cannot be sorted directly based on the processing times. We may sort jobs according to $\Phi_{\xi_i}^{-1}(\alpha)$ which is the inverse uncertainty distribution of ξ_i and α is a given confidence level. The other chromosomes are randomly generated by exchanging two genes of the initial chromosome. Repeat this process for *pop_size* – 1 times and we obtain *pop_size* chromosomes. The Property 1 is used to adjust the initial solution. After the initial solution is generated, determine whether the initial solution satisfies property 1. If the condition of Property 1 is satisfied, the position of the two adjacent jobs are exchanged.
- **Step 3: Fitness**
  The fitness of an individual is defined as the possibility that an individual is likely to stay during the selection process. The objective functions of all chromosomes are calculated as their fitness values. **Algorithm** 1 is the key to obtain the value of the objective function.
- **Step 4: Selection process**
  Chromosomes are selected by spinning the roulette wheel. These percentage fitness values can be used to configure roulette. Each time the wheel stops this gives the fitter individuals the greatest chance of being selected for the next generation and subsequent mating pool.
- **Step 5: Crossover**
  The crossover operation uses two-point crossover method: two crossover point are selected, binary string from beginning of chromosome to the first crossover point is copied from one parent, the part from the first to the second crossover point is copied from the second parent and the rest is copied from the first parent. Select two chromosomes randomly and generate a random number r_c. If r_c is smaller than P_c, then crossover is applied to this pair; otherwise, no crossover is performed.
- **Step 6: Mutation**
  The purpose of mutation operation is to maintain diversity in the population as well as to prevent the seek process from getting fall into a local optima. The mutation operation selects two positions in a chromosome randomly and interchange the two positions. In the mutation step, two chromosomes are randomly selected and a random number r_m is generated first. If r_m is smaller than P_m, then mutation is applied to this pair; otherwise, no mutation is performed.
- **Step 7: Iteration**
  When the maximum number of iterations is reached, the process is terminated; otherwise, circulate from the selection process.

5. NUMERICAL EXPERIMENT

Numerical experiments are conducted to assess the performance of the proposed chance-constrained model and hybrid algorithm. Particle swarm optimization (PSO) algorithm provides flexible, high performance mathematical programming solvers for linear programming, mixed integer programming, quadratic programming, and quadratically constrained programming problems. Since the problem in the literature [54] is similar to the problem in this paper, we can use PSO-M in the literature [54] to solve the model. Low [54] applied the "job-to-position" represent the particles and embedded a restarting strategy and three stopping criteria. The experimental results reveal that the performances of the PSO-M is quite satisfactory on both solution accuracy and efficiency. GA, GA-M and PSO-M [54] are used to solve three instances, respectively. GA denotes that initial solutions are randomly generated and without using Property 1.

Run GA, GA-M and PSO-M five times in each instance, respectively. All numerical experiments are performed on a computer with a i7 processor with a speed of 4.0 GHz and 16GB RAM. "No" column lists running index of algorithm. "GA" column lists objective function values obtained by the GA. "GA-M" column lists objective function values obtained by the heuristic method in Section 4. "PSO-M" column lists objective function values obtained by the modified PSO. "time" column lists the CPU time.

Assume that the number of job is 10. Processing time of job i is considered a normal uncertain variable: $\xi_i \sim \mathcal{N}(i, 1)$ for $i = 1, 2, \cdots, 10$. Maintenance time is 1. Length of each availability interval is 12. Confidence levels $\alpha = 0.8$. Population size of GA is 50. In GA, crossover rate is 0.75. Mutation rate is 0.25. Maximum iteration number is 300. In PSO-M, the learning factors are set as $c1 = c2 = 2.0$, inertia weight $w_{max} = 0.9$, $w_{min} = 0.4$ and $V_{max} = 4.0$. The $rand_1$ and $rand_2$ are random variables between 0 and 1. Maximum iteration number is 300.

Assume that the number of job is 20. Processing time of job i is considered a normal uncertain variable: $\xi_i \sim \mathcal{N}(i, 1)$ for $i = 1, 2, \cdots, 20$. Maintenance time is 5. Length of each availability interval is 25. Confidence levels $\alpha = 0.8$. Population size of GA is 50. In GA, crossover rate is 0.8. Mutation rate is 0.2. Maximum iteration number is 200. In PSO-M, the learning factors are set as $c1 = c2 = 2.0$, inertia weight $w_{max} = 0.9$, $w_{min} = 0.4$ and $V_{max} = 4.0$. The $rand_1$ and $rand_2$ are random variables between 0 and 1. Maximum iteration number is 200.

Assume that the number of job is 50. Processing time of job i is considered a normal uncertain variable: $\xi_i \sim \mathcal{N}(i, 1)$ for $i = 1, 2, \cdots, 50$. Maintenance time is 10. Length of each availability interval is 55. Confidence levels $\alpha = 0.8$. Population size of GA is 50. In GA, crossover rate is 0.7. Mutation rate is 0.3. Maximum iteration number is 200. In PSO-M, the learning factors are set as $c1 = c2 = 2.0$, inertia weight $w_{max} = 0.9$, $w_{min} = 0.4$ and $V_{max} = 4.0$. The $rand_1$ and $rand_2$ are random variables between 0 and 1. Maximum iteration number is 200.

In addition, we give an example about the operation process of Algorithm 1. If the measure is less than α, the job arrangement cannot be adopted.

The schedule result obtained by the LPT rule is

$$(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)(\text{sequence of jobs})$$

for small scale (10 jobs). $T = 12$ and $t = 1$. Algorithm 1 is the key to obtain the result. Detailed procedure is as follows.

$$\mathcal{M}\{\xi_{10} \leq T\} = \Phi(T) \approx 0.95 > \alpha = 0.8$$

and

$$\mathcal{M}\{\xi_{10} + \xi_9 \leq T\} = \Phi(T) \approx 0 < \alpha$$

where $T = 12$. Subsequently, job 10 will be arranged to the first availability interval.

Since

$$\mathcal{M}\{\xi_9 + \xi_8 \leq T\} = \Phi(T) \approx 0.002 < \alpha,$$

job 9 will be arranged to the second availability interval.

Likewise,

$$\mathcal{M}\{\xi_8 + \xi_7 \leq T\} = \Phi(T) \approx 0.06 < \alpha,$$

job 8 will be arranged to the third availability interval.

The job 7 will be arranged to the forth availability interval by

$$\mathcal{M}\{\xi_7 + \xi_6 \leq T\} = \Phi(T) \approx 0.29 < \alpha.$$

The job 6 will be arranged to the fifth availability interval by

$$\mathcal{M}\{\xi_6 + \xi_5 \leq T\} = \Phi(T) \approx 0.71 < \alpha.$$

The jobs 5 and 4 will be arranged to the sixth availability interval by

$$\mathcal{M}\{\xi_5 + \xi_4 \leq T\} = \Phi(T) \approx 0.95 > \alpha,$$

and

$$\mathcal{M}\{\xi_5 + \xi_4 + \xi_3 \leq T\} = \Phi(T) = 0.5 < \alpha.$$

The jobs 3, 2 and 1 will be arranged to the seventh availability interval by

$$\mathcal{M}\{\xi_3 + \xi_2 \leq T\} = \Phi(T) \approx 0.999 > \alpha,$$

and

$$\mathcal{M}\{\xi_3 + \xi_2 + \xi_1 \leq T\} = \Phi(T) \approx 0.97 > \alpha.$$

According to the above discussion, it yields $M = 7$ and $Sum_M = \xi_3 + \xi_2 + \xi_1$. And according to the Definition 3, it yields $\xi_3 + \xi_2 + \xi_1 \sim \mathcal{N}(6, 3)$. Then according to Example 2, it yields $\Phi^{-1}_{Sum_M}(\alpha) = 6 + \frac{3\sqrt{3}}{\pi} \ln \frac{0.8}{1-0.8} \approx 8.3$. It implies that the total processing time of jobs in the Mth availability interval is 8.3 at the confidence level 0.8. According to Eq. (5), it yields $\Psi_f^{-1}(\alpha) = (7-1)(12+1)+8.3 = 86.3$ and it implies that the makespan is 86.3 under confidence level 0.8.

The schedule result obtained by the LSPT rule is

$$(10, 9, 1, 8, 2, 7, 3, 6, 4, 5)(\text{sequence of jobs}).$$

$$\mathcal{M}\{\xi_{10} \leq T\} = \Phi(T) \approx 0.95 > \alpha = 0.8,$$

$$\mathcal{M}\{\xi_{10} + \xi_9 \leq T\} = \Phi(T) \approx 0 < \alpha,$$

$$\mathcal{M}\{\xi_{10} + \xi_1 \leq T\} = \Phi(T) \approx 0.71 < \alpha,$$

and job 10 will be arranged to the first availability interval.

Since

$$\mathcal{M}\{\xi_9 + \xi_8 \leq T\} = \Phi(T) \approx 0.002 < \alpha,$$

$$\mathcal{M}\{\xi_9 + \xi_1 \leq T\} = \Phi(T) \approx 0.86 > \alpha,$$

$$\mathcal{M}\{\xi_9 + \xi_1 + \xi_2 \leq T\} = \Phi(T) = 0.5 < \alpha,$$

jobs 9 and 1 will be arranged to the second availability interval.

Likewise, jobs 8 and 2 will be arranged to the third availability interval by

$$\mathcal{M}\{\xi_8 + \xi_7 \leq T\} = \Phi(T) \approx 0.06 < \alpha,$$

$$\mathcal{M}\{\xi_8 + \xi_2 \leq T\} = \Phi(T) \approx 0.86 > \alpha,$$

$$\mathcal{M}\{\xi_8 + \xi_2 + \xi_3 \leq T\} = \Phi(T) \approx 0.35 < \alpha.$$

The jobs 7 and 3 will be arranged to the forth availability interval by

$$\mathcal{M}\{\xi_7 + \xi_6 \leq T\} = \Phi(T) \approx 0.29 < \alpha,$$

$$\mathcal{M}\{\xi_7 + \xi_3 \leq T\} = \Phi(T) \approx 0.86 > \alpha.$$

The jobs 6 and 4 will be arranged to the fifth availability interval by

$$\mathcal{M}\{\xi_6 + \xi_5 \leq T\} = \Phi(T) \approx 0.71 < \alpha,$$

$$\mathcal{M}\{\xi_6 + \xi_4 \leq T\} = \Phi(T) \approx 0.86 > \alpha.$$

At last, job 5 will be arranged to the fifth availability interval.

According to the above discussion, it yields $M = 6$ and $Sum_M = \xi_5$. Since $\xi_5 \sim \mathcal{N}(5, 1)$, and according to Example 2, it yields $\Phi^{-1}_{Sum_M}(\alpha) = 5 + \frac{\sqrt{3}}{\pi} \ln \frac{0.8}{1-0.8} \approx 5.76$. It reveals that the total processing time of jobs in the Mth availability interval is 5.76 at the confidence level 0.8. According to Eq. (5), it yields $\Psi_f^{-1}(\alpha) = (6-1)(12+1) + 5.76 = 70.76$ and it reveals that the makespan is 70.76 under confidence level 0.8.

Tables 1 and 2 indicate that three algorithms can solve the model effectively. Table 3 indicates that the time required for GA-M are significantly less than the other two algorithms at large scale. The results obtained by GA-M are better than the other two algorithms. It reveals that LSPT and Property 1 play significant roles in the efficiency of the algorithm. In addition, the example of Algorithm 1 reveals that the LSPT outperforms LPT in the case of small scale obviously. The result suggests that LSPT can effectively reduce the amount of occupied availability interval.

Table 1 Results for 10 jobs.

No	GA	Time (s)	GA-M	Time (s)	PSO-M	Time (s)
1	71.9	15	67.5	15	71.6	13
2	72.7	13	66.1	17	70.1	15
3	71.5	12	67.8	15	70.4	19
4	70.7	15	70.3	19	72.7	12
5	69.4	10	68.9	13	71.0	17

GA, genetic algorithm; GA-M, improved genetic algorithm; PSO, particle swarm optimization.

Table 2 Results for 20 jobs.

No	GA	Time (s)	GA-M	Time (s)	PSO-M	Time (s)
1	253.5	20	245.3	26	251.3	23
2	251.7	22	247.5	25	253.2	25
3	258.8	26	245.5	25	251.7	25
4	253.3	26	247.1	24	255.9	26
5	254.5	28	243.4	25	258.3	26

GA, genetic algorithm; GA-M, improved genetic algorithm; PSO, particle swarm optimization.

Table 3 Results for 50 jobs.

No	GA	Time (s)	GA-M	Time (s)	PSO-M	Time (s)
1	1389.3	73	1330.2	68	1387.3	75
2	1385.5	79	1328.7	69	1389.8	74
3	1397.1	72	1330.6	69	1386.2	73
4	1392.2	69	1328.9	65	1397.4	75
5	1389.1	73	1325.1	68	1380.1	75

GA, genetic algorithm; GA-M, improved genetic algorithm; PSO, particle swarm optimization.

6. CONCLUSIONS

In this paper, a single machine scheduling problem with periodic maintenance was studied. To achieve the more accurate right decision in the turbulent and complex modern society, the uncertain factors were considered in the study. The processing time was considered an uncertain variable due to numerous practical historical data are unavailable or untrustworthy. To study the effects of uncertain variables on the scheduling problem, an uncertain chance-constrained model was proposed. In the model, pessimistic value was adopted to solve the imprecise objective function, and the chance-constrained programming method was adopted to regulate the confidence level of imprecise constraint satisfaction. Based on the uncertainty theory, the equivalent form of the chance-constrained model was obtained. Besides, a key algorithm for solving the model was also proposed. An efficient rule called LSPT combining the advantages of both LPT and SPT was proposed. To solve this model effectively, an algorithm GA-M combining the LSPT rule and Property 1 was proposed. Numerical experiments suggested that the proposed GA-M algorithm performs well.

For future work, it would be interesting to develop an improved method to obtain better solution for the problem. Future research will focus also on different shop environment (flow shop, job shop and open shop). In addition, other performance measures, for example total completion time, total weighted completion time and tardiness of jobs would also be considered.

CONFLICT OF INTERESTS

The authors declare they have no conflicts of interest.

AUTHORS' CONTRIBUTIONS

Jiayu Shen and Yuanguo Zhu contributed to the design and implementation of the research, to the analysis of the results, and to the writing of the manuscript.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (No.61673011) and Research Foundation of NIIT (YK18-10-02, YK18-10-03).

REFERENCES

[1] C. Lee, Machine scheduling with an availability constraint, J. Global Optim. 9 (1996), 395–416.
[2] C. Low, M. Ji, C. Hsu, C. Su, Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance, Appl. Math. Model. 34 (2010), 334–342.
[3] J. Lee, Y. Kim, Minimizing the number of tardy jobs in a single-machine scheduling problem with periodic maintenance, Comput. Oper. Res. Vol.39, No.9,39 (2012), 2196–2205.
[4] R. Benmansour, H. Allaoui, A. Artiba, S. Hanafi, Minimizing the weighted sum of maximum earliness and maximum tardiness costs on a single machine with periodic preventive maintenance, Comput. Oper. Res. 47 (2014), 106–113.
[5] K. Ying, C. Lu, J. Chen, Exact algorithms for single-machine scheduling problems with a variable maintenance, Comput. Indus. Eng. 98 (2016), 427–433.
[6] W. Cui, Z. Lu, Minimizing the makespan on a single machine with flexible maintenances and jobs' release dates, Comput. Oper. Res. 80 (2017), 11–22.
[7] G. Li, M. Liu, S. Sethi, D. Xu, Parallel-machine scheduling with machine-dependent maintenance periodic recycles, Int. J. Prod. Econ. 186 (2017), 1–7.
[8] J. Pacheco, S. Porras, S. Casado, B. Baruque, Variable neighborhood search with memory for a single-machine scheduling problem with periodic maintenance and sequence-dependent set-up times, Knowl. Based Syst. 145 (2018), 236–249.
[9] V. Nesello, A. Subramanian, M. Battarra, G. Laporte, Exact solution of the single-machine scheduling problem with periodic maintenances and sequence-dependent setup times, Eur. J. Oper. Res. 266 (2018), 498–507.
[10] P. Perez-Gonzalez, J. Framinan, Single machine scheduling with periodic machine availability, Comput. Ind. Eng. 123 (2018), 180–188.
[11] T. Wang, R. Baldacci, A. Lim, Q. Hu, A branch-and-price algorithm for scheduling of deteriorating jobs and flexible periodic maintenance on a single machine, Eur. J. Oper. Res. 271 (2018), 826–838.
[12] B. Liu, Uncertainty Theory, second ed., Springer-Verlag, Berlin, 2007.

[13] B. Liu, Uncertainty Theory: a Branch of Mathematics for Modeling Human Uncertainty, Springer-Verlag, Berlin, 2010.
[14] H. Ke, S. Chai, R. Cheng, Selling or sharing: business model selection problem for an automobile manufacturer with uncertain information, J. Intell. Fuzzy Syst. 36 (2018), 1–16.
[15] G. Yang, W. Tang, R. Zhao, An uncertain furniture production planning problem with cumulative service levels, Soft Comput. 21 (2017), 1041–1055.
[16] Y. Zhu, Functions of uncertain variables and uncertain programming, J. Uncertain Syst. 6 (2012), 278–288. http://www.worldacademicunion.com/journal/jus/jusVol06No4paper07.pdf
[17] J. Shen, Y. Zhu, Chance-constrained model for uncertain job shop scheduling problem, Soft Comput. 20 (2016), 2383–2391.
[18] J. Shen, Y. Zhu, Uncertain flexible flow shop scheduling problem subject to breakdowns, J. Intell. Fuzzy Syst. 32 (2017), 207–214.
[19] J. Shen, K. Zhu, An uncertain single machine scheduling problem with periodic maintenance, Knowl. Based Syst. 144 (2018), 32–41.
[20] J. Shen, Y. Zhu, An uncertain programming model for single machine scheduling problem with batch delivery, J. Ind. Manage. Optim. 15 (2019), 577–593.
[21] J. Shen, Y. Zhu, A parallel-machine scheduling problem with periodic maintenance under uncertainty, J. Amb. Intell. Hum. Comput. 10 (2019), 3171–3179.
[22] J. Shen, An uncertain parallel machine problem with deterioration and learning effect, Comput. Appl. Math. 38 (2019), 1–17.
[23] S. Li, J. Peng, A new approach to risk comparison via uncertain measure, Ind. Eng. Manage. Syst. 11 (2012), 176–182.
[24] Y. Liu, D. Ralescu, Risk index in uncertain random risk analysis, Int. J. Uncertain. Fuzziness Knowl. Based Syst. 22 (2014), 491–504.
[25] Y. Liu, D. Ralescu, Value-at-risk in uncertain random risk analysis, Inf. Sci. 391–392 (2017), 1–8.
[26] X. Chen, D. Ralescu, Liu process and uncertain calculus, J. Uncertain. Anal. Appl. 1 (2013), 1–12.
[27] K. Yao, Uncertain calculus with renewal process, Fuzzy Optim. Decis. Making. 11 (2012), 285–297.
[28] K. Yao, Multi-dimensional uncertain calculus with Liu process, J. Uncertain Syst. 8 (2014), 244–254. http://www.worldacademicunion.com/journal/jus/jusVol08No4paper02.pdf
[29] Y. Chen, Y. Zhu, Optimistic value model of indefinite LQ optimal control for discrete-time uncertain systems, Asian J. Control. 20 (2018), 495–510.
[30] Y. Chen, Y. Zhu, B. Li, Indefinite LQ optimal control with cross term for discrete-time uncertain systems, Math. Methods Appl. Sci. 42 (2019), 1194–1209.
[31] L. Deng, Y. Chen, Optimal control of uncertain systems with jump under optimistic value criterion, Eur. J. Control. 38 (2017), 7–15.
[32] L. Deng, Z. You, Y. Chen, Optimistic value model of multidimensional uncertain optimal control with jump, Eur. J. Control. 39 (2018), 1–7.
[33] B. Li, Y. Zhu, Y. Sun, G. Aw, K. Teo, Multi-period portfolio selection problem under uncertain environment with bankruptcy constraint, Appl. Math. Model. 56 (2018), 539–550.
[34] H. Yan, Y. Zhu, Bang-bang control model with optimistic value criterion for uncertain switched systems, J. Intell. Manuf. 28 (2017), 527–534.
[35] B. Liu, Some research problems in uncertainty theory, J. Uncertain Syst. 3 (2009), 3–10. http://www.worldacademicunion.com/journal/jus/jusVol03No1paper01.pdf
[36] B. Liu, Theory and Practice of Uncertain Programming, second ed., Springer-Verlag, Berlin, 2009.
[37] A. Charnes, W. Cooper, Constrained-chance programming, Manage. Sci. 6 (1959), 73–79.
[38] W. Biao, E., Yao, Lower bounds and modified LPT algorithm for k-partitioning problems with partition matroid constraint, Appl. Math. 23 (2008), 1–8.
[39] O. Braun, C. Fan, R. Graham, Worst-case analysis of the LPT algorithm for single processor scheduling with time restrictions, OR Spectrum. 38 (2016), 531–540.
[40] C. Koulamas, G. Kyparisis, An improved delayed-start LPT algorithm for a partition problem on two identical parallel machines, Eur. J. Oper. Res. 187 (2008), 660–666.
[41] C. Koulamas, G. Kyparisis, A modified LPT algorithm for the two uniform parallel machine makespan minimization problem, Eur. J. Oper. Res. 196 (2009), 61–68.
[42] Z. Tan, Y. Chen, A. Zhang, On the exact bounds of SPT for scheduling on parallel machines with availability constraints, Int. J. Prod. Econ. 146 (2013), 293–299.
[43] L. Wan, R. Ma, J. Yuan, Primary-secondary bicriteria scheduling on identical machines to minimize the total completion time of all jobs and the maximum T-time of all machines, Theor. Comput. Sci. 518 (2014), 117–123.
[44] A. Zhang, H. Wang, Y. Chen, G. Chen, Scheduling jobs with equal processing times and a single server on parallel identical machines, Discrete Appl. Math. 213 (2016), 196–206.
[45] A. Hamzadayi, G. Yildiz, Modeling and solving static m identical parallel machines scheduling problem with a common server and sequence dependent setup times, Comput. Ind. Eng. 106 (2017), 287–298.
[46] L. Abreu, J. Cunha, B. Prata, J. Framinan, A genetic algorithm for scheduling open shops with sequence-dependent setup times, Comput. Oper. Res. 113 (2020), 104793.
[47] C. Kahraman, O. Engin, I. Kaya, M. Yilmaz, An application of effective genetic algorithms for solving hybrid flow shop scheduling problems, Int. J. Comput. Intell. Syst. 1 (2008), 134–147.
[48] Y. Ouazene, F. Yalaoui, H. Chehade, A. Yalaoui, Workload balancing in identical parallel machine scheduling using a mathematical programming method, Int. J. Comput. Intell. Syst. 7 (2014), 58–67.
[49] S. Wang, M. Liu, A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem, Comput. Oper. Res. 40 (2013), 1064–1075.
[50] C. Yu, Q. Semeraro, A. Matta, A genetic algorithm for the hybrid flow shop scheduling with unrelated machines and machine eligibility, Comput. Oper. Res. 100 (2018), 211–229.
[51] N. Kundakcı, O. Kulak, Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem, Comput. Ind. Eng. 96 (2016), 31–51.
[52] D. Laha, J. Gupta, An improved cuckoo search algorithm for scheduling jobs on identical parallel machines, Comput. Ind. Eng. 126 (2018), 348–360.
[53] E. Pesch, M. Sterna, Late work minimization in flow shops by a genetic algorithm, Comput. Ind. Eng. 57 (2009), 1202–1209.
[54] C. Low, C. Hsu, C. Su, A modified particle swarm optimization algorithm for a single-machine scheduling problem with periodic maintenance, Expert Syst. Appl. 37 (2010), 6429–6434.