Journal of Systems Engineering and Electronics
Vol. 31, No. 2, April 2020, pp.350 – 358

Robust single machine scheduling problem with uncertain job due dates for industrial mass production

YUE Fan[1,2], SONG Shiji[2,*], JIA Peng[3], WU Guangping[1], and ZHAO Han[2,4]

1. Department of Automation, Tsinghua University, Beijing 100084, China;
2. Research Institute of Systems Engineering, Beijing 100072, China;
3. Jiuquan Satellite Launch Centre, Jiuquan 735000, China;
4. Department of Basic Science, Army Logistics Academy, Chongqing 401311, China

Abstract: The single machine scheduling problem which involves uncertain job due dates is one of the most important issues in the real make-to-order environment. To deal with the uncertainty, this paper establishes a robust optimization model by minimizing the maximum tardiness in the worst case scenario over all jobs. Unlike the traditional stochastic programming model which requires exact distributions, our model only needs the information of due date intervals. The worst case scenario for a given sequence that belongs to a set containing only n scenarios is proved, where n is the number of jobs. Then, the model is simplified and reformulated as an equivalent mixed 0-1 integer linear programming (MILP) problem. To solve the MILP problems efficiently, a heuristic approach is proposed based on a robust dominance rule. The experimental results show that the proposed method has the advantages of robustness and high calculating efficiency, and it is feasible for large-scale problems.

Keywords: robust optimization, single machine scheduling, maximum tardiness, uncertain due date.

DOI: 10.23919/JSEE.2020.000012

1. Introduction

Since the job due dates are usually involved in the measure of performance, the single machine scheduling problem which involves uncertain job due dates is one of the most important issues in the real make-to-order environment. This problem is motivated by a real-world application in an iron-making process. Specifically, the blast furnace produces molten iron and taps it into a set of torpedo cars (TPCs) once in a while. Then they are transported to the pretreatment station for dephosphorization, desulfurization, etc. The pretreatment station and TPCs are regarded as a single machine and jobs in this problem, respectively. From blast furnaces to the pretreatment station, TPCs are transported via rail. Since the speed and the route are difficult to be precisely determined, it is hard to predict the release times accurately when TPCs reach the pretreatment station. In general, the job due date is always closely related to the job release time.

The most common relationship between the due date and release time [1] is that $d_j = r_j + p_j + c$, where r_j, d_j and p_j represent the release time, the due date and the processing time of the job j, respectively, c denotes a common slack variable. For the iron-making, once the TPC is not processed completely before the due date, it will have a tardiness. This means the temperature of the molten iron starts to drop. We need to reheat the molten iron in the converter if the temperature drops too much [2,3]. Thus, it is necessary to find a robust schedule by minimizing the maximum tardiness when the worst case scenario occurs.

In order to hedge the uncertainty in the production environment, the influence of uncertain parameters is considered in the single machine scheduling problem [4 – 6]. Fuzzy optimization, stochastic programming and robust optimization are three main approaches to obtain robust schedules [6 – 8]. However, the stochastic optimization requires exact distribution of uncertain parameters, which limits its scope of applications [9,10]. Robust optimization is another way to optimize the system performance in uncertain environments [11 – 13]. The advantage of robust optimization is that it only requires a small amount of information for uncertain parameters, which are readily available in production. It was first introduced in 1995 in [14] and its aim was to minimize the maximum costs in the worst case scenario. There are three common ways to representing uncertain parameters: the interval data, a discrete set of scenarios, and the random variables [15,16]. In [17], the robust optimization model was established, in which the processing times of jobs were un-

Manuscript received September 03, 2018.
*Corresponding author.
This work is supported by the National Natural Science Foundation of China (61503211; U1660202).

certain, and could only take values from closed intervals. The scenarios-based robust optimization models were presented in [18,19]. They considered the uncertain processing times as a discrete set of scenarios. Chang et al. [5] proposed a novel distributional robust optimization model for a single machine scheduling problem with the uncertain job processing times. In this model, the uncertain processing time is regarded as a random variable. They use the mean and variance of the processing times, and develop three polynomial-time algorithms to solve the model. The objective function of the robust optimization has two main forms: absolute robustness and relative robustness. There are two common categories of absolute robust criteria: minimizing the maximum absolute cost [20] and minimizing the maximum regret [21], which is a deviation from the optimal value. Relative robustness is to minimize the maximum relative deviation from the optimal value [22].

The minimizing maximum tardiness problem with unequal release time $(1|r_j|T_{max})$ is a non-deterministic polynomial hard problem (NP-hard) without considering the setting of the due date [23]. Some researchers proposed many algorithms early based on branch and bound which were able to give exact solutions to small-scale problems, but for large-scale problems, they were computationally expensive [24 – 26]. In recent years, many heuristics were proposed to solve large-scale problems, such as the self-adaptive differential evolution heuristic approach [27], the genetic algorithm [28], and the approximate algorithm [29]. In general, if the deterministic problem is NP-hard, the corresponding robust optimization problem is also NP-hard [30] in the presence of uncertainty. For solving a robust single machine scheduling problem, some researchers firstly identified the worst case scenario, then developed efficient algorithms from the methods for deterministic problems in [31]. Other researchers proposed fast and effective algorithms such as meta-heuristics and genetic algorithms in [32,33]. These algorithms can solve robust solutions in a short time, especially for large-scale problems.

In this paper, we establish a min-max robust optimization model with the uncertain job due date, which only requires time intervals of job due dates and can be easily estimated from historical data in practice. We prove that the worst case scenario for a given sequence belongs to a set containing only n scenarios, where n is the number of jobs. The robust model is simplified and reformulated as an equivalent mixed 0-1 integer linear programming (MILP) problem. Then, an efficient heuristic approach robust dominance heuristic (RDH), based on the proven robust dominance rule is proposed to solve the MILP problem. The experimental results show that the proposed method has the advantages of robustness and high calculating efficiency, and it is feasible for large-scale problems.

The rest of this paper is organized as follows. Section 2 provides the establishment of the robust optimization model. And a property is derived to reformulate the model. In Section 3, we propose the RDH to solve the model efficiently. Numerical experiments and results are detailed in Section 4. Section 5 gives a conclusion for the paper.

2. Problem formulation and reformulation

We suppose that n jobs in set $J = \{1, 2, \ldots, n\}$ are processed on a single machine. It assumes that there is no machine breakdown or preemption.

A feasible schedule is a job sequence, which can be represented by a matrix $\boldsymbol{x}=[x_{ij}]$ $(\forall i, j = 1, 2, \ldots, n)$, where

$$x_{ij} = \begin{cases} 1, & \text{job } j \text{ is assigned to the } i\text{th position of} \\ & \text{the sequence} \\ 0, & \text{otherwise} \end{cases}.$$

It is obvious that the set of feasible schedules can be given as follows:

$$X = \left\{ \boldsymbol{x} \middle| \sum_{i=1}^{n} x_{ij} = 1, \forall j = 1, 2, \ldots, n; \sum_{j=1}^{n} x_{ij} = 1, \right.$$
$$\left. \forall i = 1, 2, \ldots, n; x_{ij} \in \{0, 1\}, \forall i, j = 1, 2, \ldots, n \right\}.$$

The maximum tardiness of a given sequence is that $T_{\max} = \max\left[C_i - \sum_{j=1}^{n} x_{ij} d_j\right]^+, \forall i, j = 1, 2, \ldots, n,$ where $[y]^+ = \max\{y, 0\}$, d_j is the due date of job j, C_i is the completion time of job at the ith position, we assume $C_0 = 0$. Let $f(\boldsymbol{x}) = T_{\max}$, then the deterministic single machine scheduling problem aims to find a schedule in $\boldsymbol{X}$ with the minimal $f(\boldsymbol{x})$, therefore, the problem can be formulated as the following mathematical model, we denote it as D-TP:

$$\text{(D-TP)} \qquad \min f(\boldsymbol{x}) = T_{\max} \tag{1}$$

$$\text{s.t.} \quad f(\boldsymbol{x}) \geqslant C_i - \sum_{j=1}^{n} x_{ij} d_j, \quad \forall i = 1, 2, \ldots, n \tag{2}$$

$$f(\boldsymbol{x}) \geqslant 0 \tag{3}$$

$$C_i \geqslant C_{i-1} + \sum_{j=1}^{n} x_{ij} p_j, \quad \forall i = 1, 2, \ldots, n \tag{4}$$

$$C_i \geqslant \sum_{j=1}^{n} x_{ij} (r_j + p_j), \quad \forall i = 1, 2, \ldots, n \tag{5}$$

$$d_j = r_j + p_j + c, \quad \forall j = 1, 2, \ldots, n \tag{6}$$

$$\boldsymbol{x} \in X \tag{7}$$

where p_j is the processing time of job j, r_j is the release time of job j. Equation (1) is the objective function. Equations (2) and (3) state that $f(\boldsymbol{x})$ is no less than the tardiness

of every job in the sequence $\boldsymbol{x}$ and is no less than zero. Equation (4) ensures that only one job is processed at a time. Equation (5) ensures that the machine only processes one job at a time. Equation (6) means that the lead time of each job is equal to their processing time plus a common slack variable. Equation (7) means that the best schedule belongs to a set of feasible schedules.

In actual production, it is impossible for us to accurately predict the due date of every job due to various disturbances, but a due date interval can be easily estimated. We assume that the due date interval of job j is $[\underline{d}_j, \overline{d}_j]$. That means job j can be released at any value in its due date interval. Then for a given sequence, there are infinite due date scenarios which can be represented as a set S, i.e., $S = [\underline{d}_1, \overline{d}_1] \times [\underline{d}_2, \overline{d}_2] \times \cdots \times [\underline{d}_n, \overline{d}_n]$ and $\boldsymbol{d} \in S$. S_i is the start processing time of job at the ith position.

For a schedule $\boldsymbol{x} \in X$, we define $T_{\max}$ in the worst case scenario (W-$T_{\max}$) as follows:

$$f^R(\boldsymbol{x}) = \max_{\boldsymbol{d} \in S} f(\boldsymbol{x}, \boldsymbol{d}) \tag{8}$$

where vector $\boldsymbol{d} \in S$ represents a possible due date scenario, and T_i is the tardiness of job in the ith position.

Therefore, this robust single machine scheduling problem (R-TP) takes the following form:

$$\text{(R-TP)} \quad \min_{\boldsymbol{x} \in X} f^R(\boldsymbol{x}) = \min_{\boldsymbol{x} \in X} \max_{\boldsymbol{d} \in S} f(\boldsymbol{x}, \boldsymbol{d}). \tag{9}$$

In the R-TP, since the number of due date scenarios is infinite, the exhaustive attack method is unsuitable for solving this problem. We focus on the identification of the worst case scenario firstly.

Let $T_{\max}(\boldsymbol{x}) = \max \left[C_i - d_{[i]}\right]^+, \forall i = 1, 2, \ldots, n$ represent the maximum tardiness for a given sequence $\boldsymbol{x}$, where $[i]$ denotes the job index in the ith position of $\boldsymbol{x}$. We denote the $T_{\max}(\boldsymbol{x})$ under the worst case scenario as W-$T_{\max}(\boldsymbol{x})$, and suppose that the job in the k-position is the one with the maximum tardiness in a given sequence $\boldsymbol{x}$, then

$$\text{W-}T_{\max}(\boldsymbol{x}) = T_k(\boldsymbol{x}) = [C_k - d_{[k]}]^+ \tag{10}$$

where $[k]$ represents the job index in the kth position.

Because $d_j = r_j + p_j + c$, and the value of C_k is positively related to the value of $r_{[k]}$, then C_k is also positively related to the value of $d_{[k]}$.

We cannot intuitively identify the worst case scenario of due dates. That is because in (10), if we want to maximize the value of C_k, we need to set the due dates of all jobs before the $(k+1)$th position take the maximums in their due date intervals, i.e., $d_{[1]} = \overline{d}_{[1]}, \ldots, d_{[k]} = \overline{d}_{[k]}$. If we want to minimize $d_{[k]}$, we need to set the due date of the job in the kth position that takes the minimum in its due date interval, i.e., $d_{[k]} = \underline{d}_{[k]}$. Therefore, we cannot directly identify the worst case scenario which can make $[C_k - d_{[k]}]^+$ reach the maximum.

It derives that the number of the worst possible scenarios is finite for any sequence.

Property 1 For any sequence $\boldsymbol{x}$, the worst case scenario belongs to a finite scenario set $U = \{\boldsymbol{d}^i, i = 1, 2, \ldots, n\}$. In $\boldsymbol{d}^i$, the due date of job j is defined as follows:

$$\boldsymbol{d}^i_j = \begin{cases} \overline{d}_j, & j \neq [i] \\ \underline{d}_j, & j = [i] \end{cases}. \tag{11}$$

Proof We define a set of worst possible scenarios $S(\boldsymbol{x}) = \arg\max_{\boldsymbol{d} \in S} f(\boldsymbol{x}, \boldsymbol{d})$ for schedule $\boldsymbol{x}$. Suppose that the worst case scenario $\boldsymbol{d}^* \in S(\boldsymbol{x})$ and $\boldsymbol{d}^* \notin U$. If the job in the kth position is the one with the W-$T_{\max}(\boldsymbol{x})$ in the sequence $\boldsymbol{x}$, the due date of this job can be denoted as $d_{[k]}$. Then we can always increase the W-$T_{\max}(\boldsymbol{x})$ by adjusting $d_{[k]}$ to $\underline{d}_{[k]}$ and postpone the due dates of the rest jobs to $\overline{d}_{[l]}, \forall l = 1, 2, \ldots, k-1, k+1, \ldots, n$.

Through the above adjustment, the worst case scenario $\boldsymbol{d}^* \in U$. □

Using Property 1, we can reduce the set of the worst possible scenarios from S to U with only n elements. Then we can further reformulate the R-TP into the following MILP:

$$\text{(MILP)} \quad \min_{\boldsymbol{x} \in X} f^R(\boldsymbol{x}) = \min_{\boldsymbol{x} \in X} \max_{\boldsymbol{d} \in U} f(\boldsymbol{x}, \boldsymbol{d}). \tag{12}$$

When $\underline{d}_j = \overline{d}_j$ $(j = 1, 2, \ldots, n)$, the MILP is equivalent to the D-TP. However, the due dates of all jobs belong to their intervals. Then we need to consider n scenarios in set U at the same time when we solve the MILP problem. Then the MILP has n^2 binary variables, (n^2+1) continuous variables, and $\mathcal{O}(n^2)$ constraints. The number of variables and constraints of the MILP is an order of magnitude higher than the D-TP. Due to the NP-hardness of the D-TP, the MILP is also an NP-hard problem and is more difficult to solve. To solve the MILP problem efficiently, we propose an RDH based on a robust dominance rule in the following section.

3. RDH

To illustrate the process of the RDH, we first introduce some definitions. In a given sequence $\boldsymbol{x}$, if there are two jobs i and j, such that $r_i \leqslant r_j$, $p_i \leqslant p_j$ and job j precedes job i, we define these two jobs as an incompatible pair (j, i). Let B be the set of jobs from job j to i in the sequence $\boldsymbol{x}$. Then we denote the set of jobs before and after B as B_- and B_+, respectively.

We suppose that there is an incompatible pair (j, i) in $\boldsymbol{x}$. We want to prove that if we move job i to the front of job j, the $T_{\max}$ of $\boldsymbol{x}$ will be reduced or remain unchanged.

Firstly, we define that job i dominates job j, if both inequalities $r_i \leqslant r_j$ and $p_i \leqslant p_j$ hold. Then we will prove this conclusion in the following dominance rule.

Theorem 1 Dominance rule. Consider two arbitrary jobs i and j in a given sequence $\boldsymbol{x}$ for the D-TP. If both inequalities $r_i \leqslant r_j$ and $p_i \leqslant p_j$ hold, the sequence in which i precedes j has no larger W-$T_{\max}$ value than other sequences.

Proof We design the following adjustment method to make sure that the $T_{\max}$ of the new sequence will be no larger than $T_{\max}$ of the unadjusted sequence:

(i) For a sequence $\boldsymbol{x}$, we search from scratch and find the last incompatible pair (j, i). The process start time and the completion time of the jobs in set B are expressed as S_B and C_B.

(ii) We divide set B into four subsets: $\{i\}, \{j\}, K$, and L, where $K = \{k : r_k \leqslant S_B + p_i, k \in B \setminus \{i, j\}\}$, and $L = \{k : r_k > S_B + p_i, k \in B \setminus \{i, j\}\}$. Let $|K| = h1$, $|L| = h2$, $[l]^K$ and $[l]^L$ be the index of job in the lth position in sets K and L, respectively.

(iii) Let $i \rightarrow j$ represent the job i precedes job j, then we schedule a new sequence $\boldsymbol{x}'$ as $B_- \rightarrow i \rightarrow K \rightarrow j \rightarrow L \rightarrow B_+$ and keep the order of jobs in set B_-,K, L and B_+ unchanged. Fig. 1 shows the order of jobs in $\boldsymbol{x}'$.

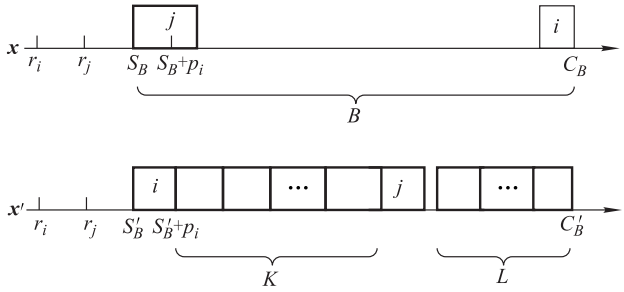


Fig. 1 Illustration for the dominance rule

In $\boldsymbol{x}$ and $\boldsymbol{x}'$, we set the earliest starting time of jobs in set B are S_B and S'_B. We set the total completion time of the jobs in set B are C_B and C'_B. Then, $T_{\max}$ of the two sequences can be expressed as follows:

$$\begin{cases} T_{\max}(\boldsymbol{x}) = \max\{T^{B_-}_{\max}(\boldsymbol{x}), T^{B}_{\max}(\boldsymbol{x}), T^{B_+}_{\max}(\boldsymbol{x})\} \\ T_{\max}(\boldsymbol{x}') = \max\{T^{B_-}_{\max}(\boldsymbol{x}'), T^{B}_{\max}(\boldsymbol{x}'), T^{B_+}_{\max}(\boldsymbol{x}')\} \end{cases} \tag{13}$$

where $T^{B}_{\max}(\boldsymbol{x})$ expresses the $T_{\max}$ of jobs of set B in $\boldsymbol{x}$.

Obviously, $T^{B_-}_{\max}(\boldsymbol{x}) = T^{B_-}_{\max}(\boldsymbol{x}')$.

The second items in $T_{\max}(\boldsymbol{x})$ and $T_{\max}(\boldsymbol{x}')$ can be expanded as follows:

$$\begin{cases} T^{B}_{\max}(\boldsymbol{x}) = \max\{T_i(\boldsymbol{x}), T_j(\boldsymbol{x}), T^{K}_{\max}(\boldsymbol{x}), T^{L}_{\max}(\boldsymbol{x})\} \\ T^{B}_{\max}(\boldsymbol{x}') = \max\{T_i(\boldsymbol{x}'), T_j(\boldsymbol{x}'), T^{K}_{\max}(\boldsymbol{x}'), T^{L}_{\max}(\boldsymbol{x}')\} \end{cases}. \tag{14}$$

In $\boldsymbol{x}$, $T_i(\boldsymbol{x}) = C_B - p_i - r_i$, $T_j(\boldsymbol{x}) = S_B - r_j$. Since $C_B - p_i \geqslant S_B$ and $r_i \leqslant r_j$, $T_i(\boldsymbol{x}) \geqslant T_j(\boldsymbol{x})$, i.e., $\max\{T_i(\boldsymbol{x}), T_j(\boldsymbol{x})\} \geqslant T_i(\boldsymbol{x})$.

In $\boldsymbol{x}'$, $T_i(\boldsymbol{x}') = S'_B - r_i$, $T_j(\boldsymbol{x}') = S'_j - r_j$. Since $S'_B = S_B$, $C_B - p_i \geqslant S_B$, $C_B - p_i \geqslant S_j$ and $r_i \leqslant r_j$, $\max\{T_i(\boldsymbol{x}'), T_j(\boldsymbol{x}')\} \leqslant T_i(\boldsymbol{x})$.

When we adjust the sequence from $\boldsymbol{x}$ to $\boldsymbol{x}'$, the earliest process start time of jobs in set K is advanced from $S_B + p_j$ to $S'_B + p_i$. Thus, the tardiness of these jobs will not be increased, i.e., $T^{K}_{\max}(\boldsymbol{x}') \leqslant T^{K}_{\max}(\boldsymbol{x})$.

In $\boldsymbol{x}'$, jobs from job i to job j are continuously processing. This suggests that fewer idle times are observed in $\boldsymbol{x}'$. Therefore, $C'_B \leqslant C_B$ and $T^{L}_{\max}(\boldsymbol{x}') \leqslant C'_B - S'_B - p_i \leqslant C_B - p_i - r_i = T_i(\boldsymbol{x})$. In summary, $T^{B}_{\max}(\boldsymbol{x}') \leqslant T^{B}_{\max}(\boldsymbol{x})$.

Because $C'_B \leqslant C_B$, S_{B_+} may be decreased. It implies that T_k $(k \in B_+)$ will not increase. Then $T^{B_+}_{\max}(\boldsymbol{x}') \leqslant T^{B_+}_{\max}(\boldsymbol{x})$.

Based on the above analysis, we know that $T_{\max}(\boldsymbol{x}') \leqslant T_{\max}(\boldsymbol{x})$.

If there is a sequence that does not satisfy the dominance rule, we can always decrease $T_{\max}$ by adjusting job positions based on the above adjustment method. □

Next, we derive the robust dominance rule based on Theorem 1 for the problem R-TP.

Theorem 2 Robust dominance rule. Consider two arbitrary jobs i and j in a given sequence $\boldsymbol{x}$ for the MILP. If both inequalities $\overline{r}_i \leqslant \underline{r}_j$ and $p_i \leqslant p_j$ hold, the sequence in which job i precedes job j has no larger W-$T_{\max}$ value than other sequences. In this case, job i dominates job j.

Proof In the MILP, $\overline{r}_i \leqslant \underline{r}_j$ means $\boldsymbol{r}^k_i \leqslant \boldsymbol{r}^k_j, \forall k = 1, 2, \ldots, n$. For any k, the MILP can be regarded as a D-TP. Based on Theorem 1, we know that the sequence where job i precedes job j has no larger W-$T_{\max}$ value than others if $r^k_i \leqslant r^k_j$ and $p_i \leqslant p_j$. Thus, the robust dominance rule holds for the problem MILP. □

Based on the Theorem 2, we propose our RDH to generate the optimal solution, and the steps of the RDH are listed as follows:

Step 1 Generate initial solution $\hat{\boldsymbol{x}}$.

The initial solution $\hat{\boldsymbol{x}}$ is given by the first come first served (FCFS) rule when $r_j = \hat{r}_j = 0.5(\underline{r}_j + \overline{r}_j)$.

Step 2 Search for the incompatible pair (j, i).

After k $(k = 1, 2, \ldots)$ iterations, we search from scratch and find the last incompatible pair (j, i) in $\boldsymbol{x}^k$. If it can be found, go to Step 3, otherwise exit the process and use the current solution as the optimal solution generated by the RDH.

Step 3 Group jobs in $\boldsymbol{x}^k$.

Divide unscheduled jobs into four blocks: $\{i\}, \{j\}, K$, and L, where $K = \{k : \overline{r}_k \leqslant S_B + p_i, k \in B \setminus \{i, j\}\}$, and $L = \{k : \underline{r}_k > S_B + p_i, k \in B \setminus \{i, j\}\}$.

Step 4 Adjust the order of the blocks and generate a new sequence.

Reschedule a new sequence $\boldsymbol{x}^{(k+1)}$ as $B_- \rightarrow i \rightarrow K \rightarrow j \rightarrow L \rightarrow B_+$ and keep the order of jobs in set B_-, K, L and B_+ unchanged. Then return $\boldsymbol{x}^{(k+1)}$ to Step 2.

Theoretically, since the RDH is an algorithm based on Theorem 2, the optimization goal can be continuously improved when searching for a better solution by the RDH. The algorithm can converge quickly and be efficient. Therefore, for the large-scale problem that the business solver IBM ILOG CPLEX optimization studio (CPLEX) based on the branch and bound method cannot solve in an acceptable time, the RDH can quickly find the suboptimal solution.

The pseudocode of the RDH is given in Algorithm 1. In the RDH, we use i to denote the position of the job in $\hat{\boldsymbol{x}}$ with W-$T_{\max}$, and use P to keep candidate sequences.

Algorithm 1

Require: $\hat{\boldsymbol{x}}$, U, $\boldsymbol{p}$ and n.

Initialize $P = \{\hat{\boldsymbol{x}}\}$.

Find the job with W-$T_{\max}$ in $\hat{\boldsymbol{x}}$ and denote its position as i, its job index is $[i]$.

while $i > 1$ do

Calculate the set of position indices $\phi_i = \{j : j \leqslant i, \text{job } [j] \text{ dominates job } [i]\}$.

if $\phi_i \neq \varnothing$ then

$k = \min\limits_{j \in \phi_i} j$.

else

$k = 1$.

end if

while $i > k$ do

Update $\hat{\boldsymbol{x}}$ by exchanging $[i]$ and $[i-1]$. Add this new sequence into P.

$i = i - 1$.

end while

Let $i = k$.

end while

Select the best sequence with the minimal W-$T_{\max}$ from P and denote it as $\boldsymbol{x}^*$.

Ensure: $\boldsymbol{x}^*$ and $f^R(\boldsymbol{x}^*)$.

Then, we use the following example with six jobs to describe the process of the RDH. We suppose that in the initial sequence $\hat{\boldsymbol{x}}$, the order of jobs is that $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$, where $i \rightarrow j$ represents job i precedes job j, while job 6 is the one with W-$T_{\max}$, job 1 dominates job 3, and job 3 dominates job 6.

Fig. 2 shows the search process of the RDH based on the initial sequence $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$. $\boldsymbol{x}^k$ expresses the sequence obtained by the kth search for $\hat{\boldsymbol{x}}$.

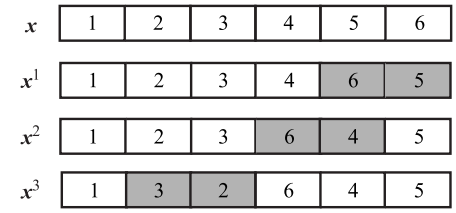


Fig. 2 Search process of RDH based on the initial sequence $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$

Since job 5 does not dominate job 6, we exchange them to generate a new sequence $\boldsymbol{x}^1$. Since job 4 also does not dominate job 6, we exchange them and generate $\boldsymbol{x}^2$. Job 6 cannot be exchanged with job 3 in terms of the fact job 3 dominates job 6, then we fix the position of job 6 and move job 3 forward. We exchange job 3 and job 2 to generate $\boldsymbol{x}^3$. Job 3 cannot be moved forward since job 1 dominates it. Then the RDH terminates. The sequence obtained by the RDH is the robust sequence of the R-TP.

4. Numerical experiments and results

In this section, we implement computational experiments to validate the effectiveness and efficiency of the proposed robust model and the proposed RDH based on artificial instances. The experiments are coded in MATLAB and executed on a personal computer with an Intel Core CPU i7-4710K and 8 GB RAM at 2.5 GHz. Experimental settings and results are presented and analyzed in following subsections.

4.1 Experimental setup

4.1.1 Experimental design

(i) Experiments on the performance of RDH

The common method to solve the single machine scheduling problem is to give an initial solution, improve the initial solution by local search (LS), variable neighborhood search (VNS), etc. [34 – 37], and get a better solution. Among these methods, the VNS has the best performance. The main idea of the VNS is to explore multiple neighborhood structures systematically instead of a single neighborhood, and escape local minima (in the case of minimization).

To evaluate the effectiveness and efficiency of the proposed RDH, we compare it with the VNS [34] and CPLEX solver which calculates the MILP with the branch and bound method.

(ii) Experiments on robustness of solutions

To evaluate the robustness of solutions, we compare robust sequence $\boldsymbol{x}^*$ obtained by the RDH with nominal sequence $\hat{\boldsymbol{x}}$ obtained by FCFS according to $\hat{r}_j$. The average W-$T_{\max}$ of $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}^*$ can be calculated. Let DEV express the average reduction of the average W-$T_{\max}$ by adopting robust sequence $\boldsymbol{x}^*$ instead of nominal sequence $\hat{\boldsymbol{x}}$, while

R-DEV is the ratio of the reduction. These four evaluation indexes are defined as follows:

$$\text{W-}T_{\max}(\hat{\boldsymbol{x}}) = \max_{\boldsymbol{r}\in U} f(\hat{\boldsymbol{x}}, \boldsymbol{r}),$$

$$\text{W-}T_{\max}(\boldsymbol{x}^*) = \max_{\boldsymbol{r}\in U} f(\boldsymbol{x}^*, \boldsymbol{r}),$$

$$\text{DEV} = \text{W-}T_{\max}(\hat{\boldsymbol{x}}) - \text{W-}T_{\max}(\boldsymbol{x}^*),$$

$$\text{R-DEV} = [\text{W-}T_{\max}(\hat{\boldsymbol{x}}) - \text{W-}T_{\max}(\boldsymbol{x}^*)]/\text{W-}T_{\max}(\boldsymbol{x}^*).$$

From these definitions, it is clear that the DEV expresses the average reduction of the average W-$T_{\max}$ by adopting robust sequence $\boldsymbol{x}^*$ instead of nominal sequence $\hat{\boldsymbol{x}}$, while R-DEV is the ratio of the reduction.

(iii) Robustness of solutions under different distributions of uncertain release times

Note that the proposed approach does not require any information about the distribution of release times, and these experiments are designed to evaluate the robustness of solutions under some other distribution assumptions. Four classical probability distributions are examined, namely the uniform distribution, normal distribution, poisson distribution, and geometric distribution. We use 10 jobs to evaluate the robustness of the solutions. Average values of release time intervals are set as $\hat{r_j} = [50, 33, 27, 12, 55, 28, 17, 45, 38, 30]$, while processing time values are set as $p_j = [12, 15, 10, 11, 14, 10, 15, 12, 14, 11]$, and the deviation values are set as $\delta_j = [11, 13, 8, 5, 15, 9, 12, 15, 5, 13]$. Let $c = 5$. By setting these parameters, the release time intervals are fixed. Five thousand scenarios are generated based on the intervals $[\hat{r}_j - \delta_j, \hat{r}_j + \delta_j], j = 1, 2, \ldots, n$, according to the four distributions above. The mean objective values and standard deviations of $\hat{\boldsymbol{x}}$ and $\boldsymbol{x}^*$ are calculated, denoted as W-$T_{\max}(\hat{\boldsymbol{x}})$, W-$T_{\max}(\boldsymbol{x}^*)$, SD($\hat{\boldsymbol{x}}$)and SD($\boldsymbol{x}^*$), respectively. We introduce another two indexes to evaluate the robustness of $\boldsymbol{x}^*$, defined as follows:

$$\text{R-DEV}_1 = (\text{W-}T_{\max}(\boldsymbol{x}^*) - \text{W-}T_{\max}(\hat{\boldsymbol{x}}))/\text{W-}T_{\max}(\boldsymbol{x}^*),$$

$$\text{R-DEV}_2 = (\text{SD}(\hat{\boldsymbol{x}}) - \text{SD}(\boldsymbol{x}^*))/\text{SD}(\boldsymbol{x}^*).$$

4.1.2 Data setting

The above first two types of experiments are based on the same data setting, i.e., our experiments consist of the test problems involving $|J| = \{10, 50, 100, 150, 200, 250, 300, 400, 500\}$ jobs and a single machine. Let $\hat{r}_j$ be the average value of the release time interval, and δ_j represents the maximum deviation of the release times from $\hat{r}_j$, that is, $\hat{r}_j = 0.5(\underline{r}_j + \overline{r}_j)$ and $\delta_j = \overline{r}_j - \hat{r}_j$. Then the real release time interval of job j can be denoted as $[\hat{r}_j - \delta_j, \hat{r}_j + \delta_j]$.

To simulate a real single machine scheduling problem of steel-making, we set the average release time gap of two adjacent jobs to be the same with the average processing time. The processing time p_j is a random integer generated from a uniform distribution of interval $[8, 12]$. Then $\hat{r}_j$ is randomly generated from a uniform distribution of interval $[0, 10(n-1)]$. δ_j is randomly generated from a uniform distribution of interval $[5, 15]$. We set $c = 3, 5, 8$ (representing short, medium, long lead time, respectively), then $d_j = r_j + p_j + c$.

The maximum computation time for all experiments is set to 1 800 s (30 min). For problem instances of different sizes, 100 trials are conducted to obtain the average results.

4.2 Experimental results and analysis

4.2.1 Evaluation of the proposed RDH

To evaluate the performance of the proposed RDH, we compare it with variable neighborhood search (VNS) and CPLEX in terms of average W-$T_{\max}$, gap (%) which means the relative deviations of obtained solutions from the best solutions, and CPU time consumption in Table 1. The optimal solutions (or the best solution found by the methods) are highlighted in bold.

Table 1 Computational results of RDH, VNS and CPLEX

c	n	W-$T_{\max}$			Gap /%			CPU time/s		
		RDH	VNS	CPLEX	RDH	VNS	CPLEX	RDH	VNS	CPLEX
2	10	19.36	19.38	**19.17**	0.99	1.10	**0.00**	**0.06**	0.06	24.46
	50	24.49	24.55	**24.19**	1.24	1.49	**0.00**	**0.21**	0.22	137.30
	100	31.64	31.81	**31.28**	1.15	1.69	**0.00**	**0.83**	0.85	845.42
	150	39.17	39.34	**38.50**	1.74	2.18	**0.00**	**1.46**	1.50	1 239.27
	200	**46.23**	46.41	48.37	**0.00**	0.38	4.63	**2.10**	2.18	1 800.00
	250	**59.66**	60.02	64.17	**0.00**	0.61	7.56	**2.65**	2.76	1 800.00
	300	**70.42**	70.98	79.65	**0.00**	0.80	13.11	**3.22**	4.52	1 800.00
	400	**81.61**	82.45	–	**0.00**	1.03	–	**3.87**	5.83	1 800.00
	500	**93.54**	94.77	–	**0.00**	1.31	–	**4.33**	6.45	1 800.00
	10	16.55	16.61	**20.49**	0.98	1.34	**0.00**	**0.06**	0.06	162.72
	50	23.72	23.85	**23.48**	1.02	1.58	**0.00**	**0.22**	0.23	137.30
	100	29.13	29.26	**28.77**	1.25	1.70	**0.00**	**0.95**	1.05	937.46
	150	36.39	36.51	**35.81**	1.62	1.95	**0.00**	**1.74**	2.01	1 353.61

Continued

c	n	W-$T_{\max}$			Gap /%			CPU time/s		
		RDH	VNS	CPLEX	RDH	VNS	CPLEX	RDH	VNS	CPLEX
5	200	**43.76**	43.96	45.25	**0.00**	0.46	3.40	**2.37**	3.42	1 800.00
	250	**56.34**	56.81	60.72	**0.00**	0.83	7.77	**2.95**	4.18	1 800.00
	300	**67.15**	68.13	75.38	**0.00**	1.46	12.26	**3.57**	5.02	1 800.00
	400	**77.28**	78.56	–	**0.00**	1.70	–	**4.22**	6.39	1 800.00
	500	**89.63**	91.62	–	**0.00**	2.22	–	**4.61**	7.51	1 800.00
	10	13.92	13.97	**13.81**	0.80	1.16	**0.00**	**0.06**	0.06	25.39
	50	19.54	19.63	**19.27**	1.40	1.87	**0.00**	**0.22**	0.24	152.47
	100	26.36	26.47	**25.96**	1.54	1.96	**0.00**	**1.26**	1.37	875.25
	150	33.27	33.32	**32.68**	1.81	1.95	**0.00**	**1.78**	2.56	1 299.53
8	200	**40.85**	41.03	42.66	**0.00**	0.44	4.43	**2.53**	3.21	1 800.00
	250	**53.19**	53.67	57.24	**0.00**	0.90	7.61	**3.24**	4.09	1 800.00
	300	**64.35**	65.25	73.75	**0.00**	1.39	14.61	**3.67**	5.27	1 800.00
	400	**75.53**	76.87	–	**0.00**	1.77	–	**4.15**	6.93	1 800.00
	500	**87.79**	89.91	–	**0.00**	2.41	–	**4.92**	8.14	1 800.00

The results show that for any c, CPLEX can calculate the accurate optimal solutions for small-scale problems ($n \leqslant 150$) within an acceptable period, while the RDH and VNS can give approximate solutions with small gaps. In these cases, CPLEX outperforms RDH in terms of the solution quality. However, for large-scale problem instances ($150 < n < 400$), CPLEX cannot find exact solutions, but can give "best so far" solutions. When $n \geqslant 400$, CPLEX fails to give a feasible solution. At these cases, RDH and VNS can give better solutions than CPLEX within 10 s. And the solution quality of RDH is better than VNS.

As for the computational efficiency of the tested solution approaches, in Table 1, we can see that the RDH and VNS require fewer computation efforts to calculate solutions. For the same scale problems, RDH has a shorter computation time than VNS. The computation time of CPLEX increases dramatically as problem scales increase. In particular, its time consumption exceeds 1 800 s when $n \geqslant 150$. In view of this, the RDH holds absolute advantages compared to CPLEX and VNS.

4.2.2 Evaluation of the robust model and solutions

To compare the performance of robust sequence $\boldsymbol{x}^*$ with nominal sequence $\hat{\boldsymbol{x}}$ which is obtained by FCFS based on $\hat{\boldsymbol{r}}$. Results in Table 2 show that robust sequence $\boldsymbol{x}^*$, which is theoretically the optimal solution of the MILP problem, certainly possesses smaller W-$T_{\max}$. These results verify the robustness of the proposed robust scheduling model.

Table 2 Comparison between robust and nominal sequences

n	W-$T_{\max}(\hat{\boldsymbol{x}})$	W-$T_{\max}(\boldsymbol{x}^*)$	DEV	R-DEV/%
10	25.42	**19.36**	6.06	31.30
50	32.73	**24.49**	8.24	33.65
100	41.59	**31.64**	9.95	31.45
150	49.82	**39.17**	10.65	27.19
200	59.47	**46.23**	13.24	28.64
250	76.31	**59.66**	16.65	27.91
300	85.94	**70.24**	15.52	22.04
400	98.25	**81.61**	16.64	20.39
500	111.31	**93.54**	17.77	19.00

Table 3 shows the robustness of solutions with different probability distributions. It reveals that solutions based on different distributions are generally in line with solutions based on average values and deviations obtained by the RDH, even though higher mean values and lower standard deviations of the AWT can be obtained by using robust sequence $\boldsymbol{x}^*$. This means the performance of the robust sequence is more stable. The results here again confirm the distributional robustness of the proposed heuristic approach.

Table 3 Simulation results with different distributions ($n = 10$)

Distribution	W-$T_{\max}(\hat{\boldsymbol{x}})$	W-$T_{\max}(\boldsymbol{x}^*)$	SD($\hat{\boldsymbol{x}}$)	SD($\boldsymbol{x}^*$)	R-DEV$_1$/%	R-DEV$_2$/%
Uniform	21.37	**20.52**	3.47	**2.52**	4.09	27.38
Normal	21.44	**19.32**	3.59	**2.28**	6.26	36.49
Poisson	20.91	**19.96**	3.25	**2.24**	4.76	31.08
Geometric	25.26	**23.77**	4.32	**2.77**	5.97	35.87

5. Conclusions

The problem with uncertain job due dates is one of the most important issues in single machine scheduling, which is a realistic difficult problem in iron-making. To deal with the uncertainty, we establish a robust model by minimizing the maximum tardiness in the worst case scenario over all jobs. Unlike the traditional stochastic programming model

which requires exact distributions, our model needs only the information of due date intervals. The optimization objective is designed as minimizing the maximum tardiness in the worst case scenario. Based on the proven property in which the number of the worst possible scenarios is finite for any sequence, the R-TP can be reformed as a solvable MILP. Then, the RDH is proposed to solve the linear programming problem. In this algorithm, we firstly generate an initial sequence by the FCFS rule based on the average release times of all jobs. Then, the initial sequence is further improved by a variable neighborhood search according to the robust dominance rule. With three extensive experiments, we validate and verify that the RDH has clear advantages over CPLEX in terms of solution quality, robustness and computation time especially for large-scale problems.

There are many possible directions in future research. For example, we can consider other uncertain parameters, such as the processing time and delivery time. We can also design more efficient meta-heuristics to improve the effectiveness and efficiency of the proposed problem.

References

[1] PHILIP K. Due date quotation models and algorithms. Scheduling Algorithms Methods & Models, 2005, 11(3): 187 – 204.

[2] TANG L, WANG G, LIU J. A branch-and-price algorithm to solve the molten iron allocation problem in iron and steel industry. Elsevier, Computers & Operations Research, 2007, 34(10): 3001 – 3015.

[3] CLAUDE D, DAWID S, BRETT H. Molten metal treatment improvements at JW aluminum used as a method to guarantee metal quality. Light Metals, 2015, 31(6): 979 – 982.

[4] MICHAEL P. Single machine models with release dates (stochastic). Scheduling, 2016, 42(3): 293 – 317.

[5] CHANG Z, SONG S, ZHANG Y, et al. Distributionally robust single machine scheduling with risk aversion. European Journal of Operational Research, 2017, 256(1): 261 – 274.

[6] YU S, FRANK W. Sequencing and scheduling with inaccuratedata. New York: Nova Science Publishers, 2014.

[7] ROMAN S, MACIEJ H. Scheduling under fuzziness. Physica-Verlag, 2000, 5(1): 98 – 99.

[8] ADAM K, PAWE Z. Robust single machine scheduling problem with weighted number of late jobs criterion. Operations Research Proceedings, 2014: 279 – 284.

[9] MICHAEL D. Deterministic and stochastic scheduling. Proc. of the NATO Advanced Study and Research Institute on Theoretical Approaches to Scheduling Problems, 2012, 6(2): 84 – 92.

[10] ZHANG L, LIN Y, XIAO Y, et al. International journal of machine learning and cybernetics. New York: Springer, 2017.

[11] ANDREW D, CHRISTOPHE G, CHRIS B. Slack-based techniques for robust schedules. Proc. of the 6th European Conference on Planning, 2014: 115 – 128.

[12] XIONG J, XING L, CHEN Y. Robust scheduling for multiobjective flexible job-shop problems with random machine breakdowns. International Journal of Production Economics, 2013, 141(1): 112 – 126.

[13] YUE F, SONG S, ZHANG Y, et al. Single machine scheduling with uncertain release times. Proc. of the 36th Control Conference, 2017: 2729 – 2734.

[14] RICHARD D, PANAGIOTIS K. Robust scheduling to hedge against processing time uncertainty in single-stage production. Management Science, 1995, 41(2): 363 – 376.

[15] SOTSKOV Y N, LAI T C. Minimizing total weighted flow time under uncertainty using dominance and a stability box. Computers & Operations Research, 2012, 39(6): 1271 – 1289.

[16] LAI D, CHYAN T, YU S, et al. The optimality box in uncertain data for minimising the sum of the weighted job completion times. International Journal of Production Research, 2017, 7(5): 529 – 557.

[17] JORDI P. The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective. Computers & Operations Research, 2016, 66(8): 141 – 152.

[18] MONALDO M, NIKOLAUS M, OLA S. Approximating single machine scheduling with scenarios, 2008, 40(3): 153 – 164.

[19] MOGHADDAM S, KAVEH A, SEYED S, et al. A scenario-based robust optimization approach for batch processing scheduling. Proc. of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2016, 230(12): 2286 – 2295.

[20] CHRISTIAN B, JENS K. Extended GRASP for the job shop scheduling problem with total weighted tardiness objective. European Journal of Operational Research, 2017, 261(3): 835 – 848.

[21] XU X, CUI W, LIN J, et al. Robust makespan minimisation in identical parallel machine scheduling problem with interval data. International Journal of Production Research, 2013, 51(12): 3532 – 3548.

[22] NING C, YOU F. Adaptive robust optimization with minimax regret criterion: multiobjective optimization framework and computational algorithm for planning and scheduling under uncertainty. Computers & Chemical Engineering, 2017, 108(5): 425 – 447.

[23] JAN L, RINNOOY K, PETER B. Complexity of machine scheduling problems. Journal of Scheduling, 1977, 1(4): 343 – 362.

[24] KENNETH B, ZAW S. Sequencing with due-dates and early start times to minimize maximum tardiness. Naval Research Logistics, 1974, 21(1): 171 – 176.

[25] HOOGEVEEN J A, VAN DE VELDE S L. A branch-and-bound algorithm for single-machine earliness-tardiness scheduling with idle time. Informs Journal on Computing, 1996, 8(4): 402 – 412.

[26] GHAITH R, MANSOOREH M, GEORGIOS A. A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence dependent setup time. Computers & Operations Research, 2004, 31(10): 1727 – 1751.

[27] ASIYE A, HARUN A, ALI A. Minimising maximum tardiness in assembly flowshops with setup times. International Journal of Production Research, 2017, 16(11): 25 – 37.

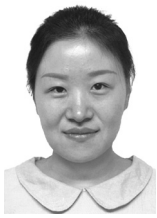
[28] PARISA A, BARZOKI R. Minimizing sum of the due date assignment costs, maximum tardiness and distribution costs in a supply chain scheduling problem. Applied Soft Computing, 2016, 47(6): 343 – 356.

[29] JAFARI A A, LOTFIM M. Single machine scheduling to minimize the maximum tardiness under piecewise linear deteriorating jobs. Scientia Iranica, 2018, 25(1): 370 – 385.

[30] MOHAMED A, CROCE D. Complexity of single machine scheduling problems under scenario-based uncertainty. Opera-

tions Research Letters, 2008, 36(3): 338 – 342.
[31] IGOR A. Minmax regret solutions for minimax optimization problems with uncertainty. Operations Research Letters, 2000, 27(2): 57 – 65.
[32] EHSAN A, MOSTAFA Z, MOJTABA F, et al. A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. Computers & Operations Research, 2016, 73(9): 56 – 66.
[33] WANG B, WANG X, LAN F, et al. A hybrid local search algorithm for robust job shop scheduling under scenarios. Applied Soft Computing, 2018, 62(4): 259 – 271.
[34] NITISH U, ALAN E, MICHEL B. The robust single machine scheduling problem with uncertain release and processing times. Journal of Scheduling, 2014, 34(15): 149 – 163.
[35] ARROYO JEC, OTTONI RDS, OLIVEIRAAD P. Multiobjective variable neighborhood search algorithms for a single machine scheduling problem with distinct due windows. Electronic Notes in Theoretical Computer Science, 2011, 281(1): 5 – 19.
[36] GUO P, CHEN W, WANG Y. A general variable neighborhood search for single-machine total tardiness scheduling problem with step-deteriorating jobs. Journal of Industrial & Management Optimization, 2017, 10(4): 1071 – 1090.
[37] JOAQU P, SANTIAGO P, SILVIA C, et al. Variable neighborhood search with memory for a singlemachine scheduling problem with periodic maintenance and sequence-dependent set-up times. Journal of Scheduling, 2013, 16(6): 661 – 673.

Biographies



YUE Fan was born in 1981. She received her Ph.D. degree in control science and engineering with the Department of Automation, Institute of System Integration in Tsinghua University, Beijing, China, in 2019. Her research interests include robust optimization, optimization algorithm, and system optimization.
E-mail: yyff1981@126.com

SONG Shiji was born in 1965. He is currently a professor with the Department of Automation, Tsinghua University, Beijing, China. His research interests include system modeling, optimization and control, computational intelligence, and pattern recognition.
E-mail: shijis@mail.tsinghua.edu.cn

JIA Peng was born in 1980. He received his B.S degree in opto-electronic engineering from Space Engineering University in 2003. He is currently a senior engineer at Jiuquan Satellite Launch Centre. His research interest inclucles application of unmanned aerial vehicle.
E-mail: jack_1699@sina.com

WU Guangping was born in 1986. He received his M.S. degree from the Academy of Armored Forces Engineering (AAFE) in 2011. He is currently an engineer at Institute of Systems Engineering. His research interests include signal processing, mechatronics engineering, and equipment demonstration.
E-mail: wgp20121314@163.com

ZHAO Han was born in 1982. He received his Ph.D. degree in control science and engineering with the Department of Automation, Institute of System Integration in Tsinghua University. His research interests include supply chain management and optimization approach.
E-mail: zhao-h15@mails.tsinghua.edu.cn