Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times

Chung-Cheng Lu a, Kuo-Ching Ying b, Shih-Wei Lin c,

a Institute of Information Management, National Taipei University of Technology, Taipei, Taiwan
b Department of Industrial Engineering and Management, National Taipei University of Technology, Taipei, Taiwan

c Department of Information Management, Chang Gung University, Taoyuan, Taiwan

Abstract

This research deals with the single machine scheduling problem (SMSP) with uncertain job processing times. The single machine robust scheduling problem (SMRSP) aims to obtain robust job sequences with minimum worst-case total flow time. We describe uncertain processing times using intervals, and adopt an uncertainty set that incorporates a budget parameter to control the degree of conservatism. A revision of the uncertainty set is also proposed to address correlated uncertain processing times due to a number of common sources of uncertainty. A mixed integer linear program is developed for the SMRSP, where a linear program for determining the worst-case total flow time is integrated within the conventional integer program of the SMSP. To efficiently solve the SMRSP, a simple iterative improvement (SII) heuristic and a simulated annealing (SA) heuristic are developed. Experimental results demonstrate that the proposed SII and SA heuristics are effective and efficient in solving SMRSP with practical problem sizes.

1. Introduction

The single machine scheduling problem (SMSP) is one of the most studied production scheduling problems due to its complexity and practical importance. This problem aims at obtaining the best sequence for a set of jobs in a manufacturing system with a single machine. The total flow time (TFT), makespan, total tardiness, or their weighted combinations are performance measures frequently used in such a system. In an industrial setting, manufacturing systems usually operate in highly uncertain environments in which interruptions (mostly random in nature) prevent the execution of production schedules exactly as they were developed. Particularly, variation in processing times and other stochastic events (e.g., machine breakdowns, rush orders, order cancellations, and raw material shortages) lead to the variability in manufacturing systems (Sabuncuoglu & Goren, 2009).

This research addresses the SMSP with uncertain job processing times. Recognizing the need to take data uncertainty into account, researchers have adopted different methodologies to obtain robust schedules in the single machine context. Stochastic programming (SP) is a classical approach to tackling job data uncertainty in SMSPs (e.g., Agrawala, Coffman, Carey, & Tripathi, 1984; Birge & Louveaux, 1997; Soroush, 2007; Trietsch & Baker, 2008; van den Akker & Hoogeveen, 2008; Wu & Zhou, 2008). Although SP provides a theoretically sound foundation for SMSPs with random data, the application of SP to large size stochastic SMSPs is hindered due to the following reasons: (i) probabilistic distribution knowledge of input data is required, (ii) optimization of expectations may be suitable for (long term) planning purposes but not of practical interest from the operational viewpoint, and (iii) the problem size will increase exponentially with the number of uncertain parameters (see e.g., Ben-Tal & Nemirovski, 1999; Bertsimas & Sim, 2003; Daniels & Kouvelis, 1995).

Robust optimization (RO; e.g., Mulvey & Vanderbei, 1995; Kouvelis & Yu, 1997), which optimizes against the worst instances using a min–max objective, is another approach that has been employed to solve SMSPs with uncertain data. Daniels and Kouvelis (1995) generated robust schedules to minimize the TFT for the SMSP with uncertain processing times. They defined schedule robustness according to absolute or relative deviations from optimal TFT in the worst-case scenario, and showed that the discrete-scenario version of the problem is NP-hard. A branch-and-bound exact algorithm and two surrogate relaxation heuristics were also developed in their research to obtain robust schedules. Yang and Yu (2002) dealt with the SMSP with uncertain processing times represented as a discrete set of scenarios, and presented an exact dynamic programming algorithm and two heuristics to obtain robust schedules. Kasperski (2005) considered uncertain
due dates and processing times represented as interval data in an SMSP with precedence constraints, and developed a polynomial time algorithm for constructing robust schedules with minimum worst-case deviation from optimal performance. Lebedev and Averbakh (2006) addressed a robust SMSP, in which the processing time of each job is represented by a prescribed interval or range. They proved that the problem is NP-hard when the min–max regret criterion is used, and showed that the problem can be solved in \( O(n \log n) \) time if the number of jobs is even and all intervals of uncertainty have the same center. Montemanni (2007) presented a mixed integer linear programming (MILP) formulation for the robust SMSP, introduced by Lebedev and Averbakh (2006), and discussed some preprocessing rules which can be utilized to efficiently solve the MILP formulation. Kasperski and Zielinski (2008) considered the TFT-minimization SMSP with processing times represented as intervals and job precedence constraints. They showed that this problem is approximable within 2 if its deterministic counterpart is polynomially solvable under a midpoint scenario. The above excellent references significantly reduce the gap between theoretical progresses and industrial practices for the SMSP with uncertain data.

With increasing awareness of the importance of solution robustness, this past decade witnessed continuous development of RO-related techniques and renewed interest in robust counterpart optimization (RCO) approaches to dealing with data uncertainty in mathematical programming. Early in the 1970s, Soyster (1973) considered coefficient uncertainty in linear programming (LP) formulations and proposed an equivalent LP model that admits the highest protection by designating all uncertain parameters to take their boundary values simultaneously. As a result, this approach may be too conservative in practice and the robust solution is much worse than the optimal solution of the nominal problem. To address the problem of over-conservativeness, El-Ghaoui, Oustry, and Lebret (1998) and Ben-Tal and Nemirovski (1999, 2000) developed the RCO formulations of LP and convex programming problems, which assume that uncertainty sets of data are ellipsoids. These formulations have the flexibility to control the degree of solution conservatism through safety parameters and constraint violation probabilities. Bertsimas and Sim (2004) have recently proposed a different RCO approach that reformulates nonlinear robust models as linear models according to the duality theory of LP. Bertsimas and Sim (2003) demonstrated that this RCO approach can be directly applied to discrete optimization models and network flow problems.

This research addressed the TFT-minimization SMSP with uncertain job processing times represented by intervals. This single machine robust scheduling problem (SMRSP) aims to obtain robust job sequences with minimum worst-case TFT. This min–max criterion is different from the min–max regret criterion used by Lebedev and Averbakh (2006). In this research the uncertainty set, proposed by Bertsimas and Sim (2003, 2004), is adopted for interval-represented processing times. Not only does it include a budget parameter for controlling the number of coefficients that can simultaneously take their largest variations, it also provides a way of incorporating different attitudes toward risk (e.g., risk-averse, risk neutral, or risk-seeking). Hence, the decision maker can select the job schedule which achieves a balance between robustness and optimality.

While most previous RO/RCO models assume independently distributed uncertain parameters, uncertain job processing times may be attributed to several common sources of uncertainty (e.g., shortage of raw materials, machine breakdowns, and employee sick leave), and hence correlated, in some manufacturing systems. A revision of the uncertainty set is proposed to address correlated uncertain processing times due to a number of common sources of uncertainty. The case in which an uncertainty source has distinct levels of impact on different groups of uncertain data is also addressed. For instance, jobs are often classified into a number of groups (e.g., Webster & Baker, 1995), each of which may require additional resources (e.g., raw materials, equipments, or labor skills), and hence the unavailability of one particular resource may only have impact on the job group that is highly dependent on that resource but no (or very little) impact on other job groups.

This research develops a mixed integer linear programming (MILP) model for the SMRSP. The basic idea is to integrate a linear program developed to determine the worst-case TFT within the integer program of the (deterministic) TFT-minimization SMSP. To efficiently solve the SMRSP, a simple iterative improvement (SI) heuristic and a simulated annealing (SA) heuristic are developed. Numerical experiments are conducted to demonstrate the effectiveness and efficiency of SI and SA heuristics. The tradeoff between robustness and optimality is examined and the impact of the degree of uncertainty on the performance measures is also explored.

The remainder of this paper is organized as follows. Section 2 defines the SMRSP of interest and presents the MILP model for the SMRSP. The SMRSP with correlated data uncertainty is also addressed and the revised uncertainty set for this problem is defined. Section 3 describes the proposed SI and SA heuristics. Numerical experimental results are provided in Section 4, followed by concluding remarks given in Section 5.

2. Problem statement and models

2.1. Single machine robust scheduling problem (SMRSP)

Consider a set \( N \) of \( n \) jobs to be processed on a single machine with the processing time of job \( j \) being \( p_j, j = 1, \ldots, n \). All jobs are released at the beginning of the scheduling period. A classical SMSP in the literature is to find an optimal processing sequence of the jobs which minimizes the sum of completion times (or TFT) of all jobs. Let \( x_{ij} = 1 \) if job \( j \) is assigned to the \( i \)th position of the sequence and 0 otherwise. \( x = \{x_{ij}, i = 1, \ldots, n\} \). The integer programming (IP) formulation of the SMSP is as follows.

\[
\text{(SMSP) Minimize } C
\]

\[
\text{Subject to } C \geq \sum_{j=1}^{n} \sum_{i=1}^{n} (n - i + 1)p_jx_{ij}, \quad (1)
\]

\[
\sum_{j=1}^{n} x_{ij} = 1, \quad j = 1, \ldots, n, \quad (2)
\]

\[
\sum_{i=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, n, \quad (3)
\]

\[
x_{ij} \in \{0, 1\}, i = 1, \ldots, n, \quad j = 1, \ldots, n. \quad (4)
\]

It is well known that an optimal solution \( x^* \) to the above SMSP can be obtained using the classical shortest processing time (SPT) rule. However, particularly when job processing times are increased due to unexpected interruptions, this classical approach, which determines job schedules according to nominal processing times, may lead to substantially unsatisfactory scheduling performance. Thus, a robust scheduling approach has to explicitly take into account variations in job processing times that result in significant delays to production schedules.

In this research, uncertain processing times (i.e., the increase of processing times due to interruptions) are represented using the following uncertainty set:

\[
U_i = \left\{ \{p_j, \bar{p}_j + \tilde{p}_j]\}, \forall j \in N: \sum_{j \in N} \bar{p}_j - \tilde{p}_j \leq \Gamma \right\}.
\]
where \( p = \{p_1, p_2, \ldots, p_n\} \). In this representation, the uncertain processing time \( p_j \) of job \( j \) has a nominal value \( p_j \) and a maximum variation \( \beta_j (p_j) \geq 0 \). This is equivalent to regarding that each uncertain processing time \( p_j \) as the random variable \( \eta_j = (p_j - p_j) / p_j \), which follows an unknown but bounded distribution taking values in \([0, 1]\). Further, by adopting the idea of Bertsimas and Sim (2004), the budget parameter \( \Gamma \) is introduced to control the degree of solution conservatism. Specifically, this uncertainty representation postulates that, in a scenario \( r \), a processing time \( p_j (r) \in [p_j, p_j + \beta_j] \) is assigned to each job \( j \) and the total variation in job processing times is less than or equal to \( \Gamma \) (i.e., \( \sum_{j=1}^{n} \frac{p_j (r) - p_j}{p_j} \leq \Gamma \)).

Let \( \Omega \) be the set of all possible scenarios satisfying this postulate. Define \( C(x, r) \) as the total flow time of sequence \( x \) in scenario \( r \). The robustness cost \( RC \) of \( x \) is defined as its maximum TFT among all possible scenarios in the set \( \Omega \); that is,

\[
RC(x, \Gamma) = \max_{r \in \Omega} C(x, r) = \max_{r \in \Omega} \sum_{j=1}^{n} (n - i + 1) p_j (r) x_{ij},
\]

The SMRSP will find the robust sequence (or schedule) \( x^* \) with the smallest (among all feasible sequences) maximum RC; that is

\[
(\text{SMRSP}) \quad x^* = \arg \min_{x \in X} RC(x, \Gamma) = \arg \min_{x \in X} \max_{r \in \Omega} \sum_{j=1}^{n} (n - i + 1) p_j (r) x_{ij},
\]

where \( X \) is the set of feasible sequences. In the above definition, the robust sequence \( x^* \) can be considered as a sequence that should guarantee reasonably good performance under all possible realizations of job processing times defined in the set \( \Omega \).

### 2.2. Mixed integer linear program for SMRSP

This subsection presents the MILP formulation for the SMRSP. First, the linear program for determining the worst-case TFT for a sequence \( x \) (or \( RC(x) \)) is introduced. According to the definition of \( U_1 \) given in Section 2.1, \( RC(x) \) can be considered as the sum of the nominal TFT (i.e., deterministic part) and the maximum possible variation of the TFT (i.e., random part) of \( x \). Furthermore, since the nominal TFT is fixed for a given job sequence \( x \), the problem of determining the RC of \( x \) (i.e., Robustness Cost Problem, RCP) is equivalent to obtaining the maximum possible variation of the TFT of sequence \( x \). Specifically, we consider the RCP with respect to the uncertainty set \( U_1 \) as follows:

\[
(\text{RCP1}) \quad \beta_1 (x, \Gamma) = \max_{m \in \{1, \ldots, n\} \mid x \in \Omega} \max_{m \in \{1, \ldots, n\}} \sum_{j=1}^{n} (n - i + 1) p_j x_{ij} + (\Gamma - |\Gamma|) \sum_{j=1}^{n} (n - i + 1) p_j x_{ij},
\]

In Eq. (8), \( m \) is a subset of \( N \) that contains \( |\Gamma| \) jobs with uncertain processing times. The idea of introducing the budget parameter, \( \Gamma \), is that it is often unlikely that the worst-case value will simultaneously happen for all the uncertain processing times. Thus, it may be adequate to allow \( |\Gamma| \) of the \( n \) uncertain processing times to obtain their worst-case value and one processing time \( p_j \) to change by \( (\Gamma - |\Gamma|) p_j \). Note that by varying the protection level, \( \Gamma \in [0, n] \), the decision-maker is able to adjust solution robustness against the level of conservatism of the solution, thus incorporating different attitudes toward the risk of uncertainty. When \( \Gamma = 0 \), \( \beta_1 (x, \Gamma) = 0 \), the SMRSP problem is equivalent to the nominal problem. On the other hand, if \( \Gamma = n \), the most conservative version of the problem, which is equal to the approach developed by Sosyter (1973), is obtained.

According to the uncertainty set \( U_1 \) and the RCP defined by Eq. (8), the RC of a job sequence \( x \) for a given protection level \( \Gamma \) (i.e., \( RC(x, \Gamma) \)) is determined as follows.

\[
RC(x, \Gamma) = \sum_{j=1}^{n} (n - i + 1) p_j x_{ij} + \beta_1 (x, \Gamma).
\]

Next, an LP reformulation of the RCP, Eq. (8), is presented in the following proposition.

**Proposition 1.** The RCP can be reformulated as the following linear program (DLP1):

\[
(\text{DLP1}) \quad \begin{align*}
\text{Minimize} & \quad \omega + \sum_{j=1}^{n} \pi_j x_{ij} y_j \\
\text{subject to} & \quad \omega + \pi_j \geq \sum_{i=1}^{n} (n - i + 1) p_j x_{ij}, \quad \forall j \in N, \\
& \quad \pi_j \geq 0, \quad \forall j \in N, \\
& \quad \omega \geq 0.
\end{align*}
\]

**Proof of Proposition 1.** Given a decision variable vector \( x \), the RCP defined in Eq. (8) can be reformulated as the following LP problem (LP1), because its optimal solution \( y^* = \{y_j, j = 1, \ldots, n\} \) consists of \( |\Gamma| \) variables equal to 1 and one variable equal to \( \Gamma - |\Gamma| \).

\[
(\text{LP1}) \quad \begin{align*}
\text{Maximize} & \quad \sum_{j=1}^{n} \sum_{i=1}^{n} (n - i + 1) p_j x_{ij} y_j \\
\text{subject to} & \quad \sum_{j=1}^{n} y_j \leq \Gamma, \\
& \quad 0 \leq y_j \leq 1, \quad \forall j \in N.
\end{align*}
\]

This is equivalent to selecting a subset \( \{m \cup r \} \mid m \subseteq \Omega, |m| = |\Gamma|, r \in N, m \} \) with the corresponding objective function \( \sum_{i=1}^{n} \sum_{j=1}^{n} (n - i + 1) p_j x_{ij} + (\Gamma - |\Gamma|) \sum_{i=1}^{n} (n - i + 1) p_j x_{ij} \). Let \( \omega \) and \( \pi_j (\forall j \in N) \) be the dual variables associated with Constraints (15) and (16), respectively. Then, the dual problem of the LP1 can be obtained as the DLP1 defined by Eqs. (10)–(13). Since Problem LP1 is feasible and bounded for \( \Gamma \in [0, n] \), by strong duality, Problem DLP1 is also feasible and bounded and their objective values coincide. Thus, the RCP defined in Eq. (8) is equivalent to Problem DLP1. This completes the proof.

By integrating the LP reformulation (DLP1) within the IP formulation of the SMSP presented in Section 2.1, the MILP for the SMRSP with a given protection level \( \Gamma \) is formulated as follows.

\[
(\text{MILP}) \quad \begin{align*}
\text{Minimize} & \quad \omega + \sum_{j=1}^{n} \sum_{i=1}^{n} (n - i + 1) p_j x_{ij} y_j + \Gamma \omega + \sum_{j=1}^{n} \pi_j \delta x_{ij} y_j
\end{align*}
\]

Subject to Eqs. (3), (4), (5), (11), (12) and (13).

### 2.3. SMRSP with correlated data uncertainty

The uncertain job processing times defined in the uncertainty set \( U_1 \) are assumed to be independently distributed within their respective interval. Nevertheless, as mentioned previously it is possible that the uncertain data are correlated. To deal with possible correlations between uncertain job processing times, we assume that data uncertainty is due to a number of common sources \( s = 1, \ldots, S \). The effect of uncertainty source \( s \) on the processing time of jobs can be represented using the random variable \( \eta_s \) distributed in \([0, 1]\). Specifically, \( p_{js} \) is denoted as the largest deviation from the nominal value \( p_j \) due to the impact of the uncertainty source \( s \) on the processing time of job \( j \). Then the uncertain processing time of job \( j \) can be expressed as:

\[
p_j = p_j + \sum_{s=1}^{S} \eta_s p_{js}.
\]

As a result, the uncertainty set for correlated uncertain processing times can be written as follows:
\( U_2 = \left\{ p_j y_j \in [y_j, p_j + \bar{p}_j], \forall y_j \in N: \sum_{j=1}^{n} p_j - p_j \leq \Gamma \right\} \),

where \( \bar{p}_j = \sum_{i=1}^{n} \bar{p}_{j_i} \). Essentially, the correlations between uncertain job processing times are modeled through the random variables \( \eta_s, s = 1, 2, \ldots, S \), corresponding to different sources of uncertainty.

The RCP with respect to the uncertainty set \( U_2 \) is defined as follows:

\[
\begin{align*}
(\text{RCP2}) \quad \beta_2(x, \Gamma) &= \max_{\{y_j(t) | y_j \in N, t \in (\Gamma), \forall k \}} \left\{ \sum_{j=1}^{n} (n - i + 1) \bar{p}_j x_{kj} + (\Gamma - [\Gamma]) \sum_{i=1}^{n} (n - i + 1) \bar{p}_k x_{ki} \right\} \\
& \quad + (\Gamma - [\Gamma]) \sum_{i=1}^{n} (n - i + 1) \bar{p}_k x_{ki} \right\} \\
\end{align*}
\]

According to Proposition 1, the RCP2 can be reformulated as the LP defined in Eqs. (10)–(13), but with Constraint (11) replaced by Constraint (19).

\[
\omega + \pi_j \geq (n - i + 1) \bar{p}_j x_{kj}, \forall y_j \in N, \forall k \\
\]

Note that the definition of uncertainty set \( U_2 \) connotes that an uncertainty source \( s \) has the same effect on all jobs (through random variable \( \eta_s \)). However, in some instances, particularly when jobs are classified into a number of groups, \( k = 1, 2, \ldots, K \), according to some group-specific resources required in the production, a source of uncertainty may affect only the processing times of a portion of the \( n \) jobs or a number of groups. In other cases, different uncertainty sources have distinct degrees of impact on different job groups. To take into account the group-specific sources of uncertainty, the random variables \( \eta_s \in [0,1] \), \( \forall s, k \), which describe the effect of uncertainty source \( s \) on job group \( k \) are introduced. Let \( j(k) \) be a job of group \( k \) that has \( n(k) \) jobs, and \( \bar{p}_{j(k)} \) be the largest deviation from the nominal value \( p_{j(k)} \) due to the impact of uncertainty source \( s \) on the processing time of job \( j(k) \). Then the uncertain processing time of job \( j(k) \) can be expressed as: \( p_{j(k)} = p_{j(k)} + \sum_{s=1}^{S} \eta_s \bar{p}_{j(k)} \). This expression leads to the following uncertainty set for group-correlated uncertain processing times.

\[
U_3 = \left\{ p_j y_j \in [y_j, p_j + \bar{p}_j], \forall y_j \in N(k), \forall k: \sum_{j \in N(k)} p_j - p_j \leq \Gamma \right\}
\]

where \( \bar{p}_j = \sum_{k=1}^{K} \bar{p}_{j(k)} \) and \( N(k) \) is the index set of the jobs in group \( k; \cup_{k=1}^{K} N(k) = N \).

The RCP with respect to the uncertainty set \( U_3 \) is defined as follows.

\[
\begin{align*}
(\text{RCP3}) \quad \beta_3(x, \Gamma) &= \max_{\{y_j(t) | y_j \in N, t \in (\Gamma), \forall k \}} \left\{ \sum_{k=1}^{K} \sum_{j \in N(k)} (n - i + 1) \bar{p}_j x_{kj} \right\} \\
& \quad + (\Gamma - [\Gamma]) \sum_{i=1}^{n} (n - i + 1) \bar{p}_k x_{ki} \right\} \\
\end{align*}
\]

\textbf{Proposition 2.} \textbf{Solving the RCP3 is equivalent to solving the following linear program.}

\[
(\text{DLP2}) \quad \text{Minimize} \quad \Gamma \omega + \sum_{k=1}^{K} \sum_{j \in N(k)} \pi_{j(k)} \\
\text{Subject to} \quad \omega + \pi_j \geq (n - i + 1) \bar{p}_j x_{kj}, \forall y_j \in N(k), \forall k, \\
\omega \geq 0, \\
\pi_{j(k)} \geq 0, \forall y_j \in N(k), \forall k.
\]

\textbf{Proof of Proposition 2.} Given a decision variable vector \( x \), the RCP3 defined in Eq. (20) is equal to the following LP (LP2), because the optimal solution \( y^* = (y_{j(k)}, \forall y_j \in N(k), \forall k) \) of the LP program consists of \( [\Gamma] \) variables equal to 1 and one variable equal to \( \Gamma - [\Gamma] \).

\[
(\text{LP2}) \quad \text{Minimize} \quad \sum_{k=1}^{K} \sum_{j \in N(k)} (n - i + 1) \bar{p}_j x_{kj} \sum_{j \in N(k)} y_{j(k)} \\
\text{Subject to} \quad \sum_{k=1}^{K} \sum_{j \in N(k)} y_{j(k)} \leq \Gamma, \\
0 \leq y_{j(k)} \leq 1, \forall y_j \in N(k), \forall k.
\]

Let \( \omega \) and \( \pi_{j(k)} \) \( (\forall y_j \in N(k), \forall k) \) be the dual variables associated with Constraints (26) and (27), respectively. Then, the dual program of LP2 can be obtained as the DLP2 defined by Eqs. (21)–(24). Since Problem L2 is feasible and bounded for \( \Gamma \in [0, \Gamma] \), by strong duality, the DLP2 is also feasible and bounded and their objective values coincide. Thus, the RCP3 defined in Eq. (20) is equivalent to the DLP2. This completes the proof.

\section{3. Heuristics for solving the SMRSP}

The MILP presented in Section 2.1 can be readily solved by commercial solvers, such as GUROBI (GUROBI Optimization Inc., 2013). However, when the size of problem instances is large, solving the MILP of the SMRSP is computationally hard. To efficiently solve the SMRSP, this research develops two heuristics, namely SII and SA, which are presented in Sections 3.1 and 3.2, respectively.

A common step of these two heuristics is to evaluate the objective function value – the robustness cost \( (RC) \) defined in Eq. (10), which is equivalent to determining the maximum possible variation of the TFT (i.e., the objective value of the RCP problem in Eq. (9)) for a given sequence \( x \). Proposition 1 presents the LP reformulation, DLP1, for the RCP problem. This LP reformulation (i.e., LP1) is the continuous Knapsack problem, which can be solved in polynomial time. Thus, evaluating the robustness cost of each sequence, which is crucial in the proposed heuristics, is easy.

\subsection*{3.1. Simple iterative improvement heuristic}

The proposed SII heuristic is a simple approach that consists of three major steps: SPT rule (e.g., Conway, Maxwell, & Miller, 1967), modified NEH algorithm (Nawaz, Enscore, & Ham, 1983), and Iterative Local Search (ILS; e.g., Toumi & Ouis, 2008). In the first step (initialization), the SPT rule minimizes the nominal or expected total flow time of the jobs by sequencing in non-decreasing order of nominal processing times. In the second step (construction), the modified NEH algorithm performs the iterative insertion of all jobs into a partial sequence according to the initial order determined by the SPT rule in Step 1. In the third step (improvement), ILS is employed to improve the sequence obtained by the modified NEH algorithm in Step 2. The algorithmic steps of the proposed SII heuristic are described as follows.

\textbf{Step 1: (SPT)}

Use the SPT rule, which sorts the nominal job processing times \( p_j, j = 1, \ldots, n \) in non-decreasing order, to obtain a job sequence \( x_{\text{SPT}} = (j_1, j_2, \ldots, j_n) \). In case of a tie, break it arbitrarily.

\textbf{Step 2: (Modified NEH Algorithm)}

2.1. Take the first two jobs from \( x_{\text{SPT}} \), that is \( j_1 \) and \( j_2 \), and schedule them in order to minimize \( RC \) as if they were only these two jobs. Note that the \( RC \) of a partial sequence with
the two jobs is the sum of the nominal TFT and the maximum possible variation obtained by the aforementioned simple greedy rule.

2.2. Do for each of the jobs \( J_1, \ldots, J_n \), successively: Insert the next job into the (partial) sequence at the position that results in the smallest \( RC \) among all possible insertion positions. Denote the final sequence of Step 2 as \( x_{\text{NEH}} = (J_1, J_2, J_3, \ldots, J_n) \).

Step 3: (ILS)

Let \( x_{\text{best}} = x_{\text{NEH}} \). Do for each of the jobs: \( J_i, \text{Best} \); \( i = 1, 2, \ldots, n - 1 \), successively: Swap its position with that of each subsequent job, i.e., \( J_i, \text{Best} \); \( i = 1 + 1, 2, \ldots, n \) in the sequence \( x_{\text{Best}} \). If the sequence after swapping has a smaller \( RC \), then set it as a new sequence of \( x_{\text{Best}} \) from which the iterative improvement procedure is continued.

3.2. SA heuristic

Given the input of job information (i.e., nominal values and maximum variations of job processing times) and parameters: \( \Gamma, T_0, T_f, \text{Iter}_{\text{max}}, \text{Num}_{\text{max}}, \) and \( \beta \), the proposed SA-based heuristic starts by generating an initial job sequence \( x_0 \) (using the SPT rule to obtain the optimal nominal sequence in this study) and initializing the current temperature \( \text{Temp} \), the number of iterations \( \text{Iter} \), the incumbent solution \( x_{\text{best}} \), and its objective value (or robustness cost) \( RC \). In each iteration, first a new solution \( x_{\text{new}} \) is generated based on the current solution \( x \). Let \( \Delta E = RC(x_{\text{new}}) - RC(x) \). If \( \Delta E \leq 0 \), then the current solution \( x \) is replaced by the new solution \( x_{\text{new}} \); otherwise, the probability of replacing \( x \) with \( x_{\text{new}} \) is \( \text{Temp} / (\text{Temp}^2 + \Delta E^2) \). This is done by generating a random number, \( r \) \((0 < r < 1)\) and replacing \( x \) with \( x_{\text{new}} \) if \( r < (\text{Temp} / (\text{Temp}^2 + \Delta E^2)) \). The current temperature \( \text{Temp} \) is decreased after running \( \text{Iter}_{\text{max}} \) iterations from the previous decrease, according to the formula \( \text{Temp} = \text{Temp} \times \beta \), where \( 0 < \beta < 1 \). If \( \text{Temp} \) is lower than \( T_f \), then the algorithm is terminated. If the incumbent solution \( x_{\text{best}} \) is not improved within \( \text{Num}_{\text{max}} \), then the algorithm is terminated. Upon termination, the optimal or near-optimal solution is \( x_{\text{Best}} \) with robustness cost \( RC(x_{\text{Best}}) \).

The procedure of the SA heuristic is presented below (Lin, Gupta, Ying, & Lee, 2009; Ying, Gupta, Lin, & Lee, 2010).

**Step 0:** Input job information and parameters: \( \Gamma, T_0, T_f, \text{Iter}_{\text{max}}, \text{Num}_{\text{max}}, \) and \( \beta \).

**Step 1:** Initialization

1.1. Generate the initial solution \( x_0 \) using the SPT rule: \( x = x_0 \). 
1.2. Initialize \( \text{Temp} = T_0, \text{Iter} = 0, x_{\text{best}} = x, RC = RC(x) \).

**Step 2:** Generate a new solution \( x_{\text{new}} \) based on \( x \) and evaluate its robustness cost \( RC(x_{\text{new}}) \).

2.1. Generate \( x_{\text{new}} \) by randomly choosing two jobs in \( x \) and swapping their positions or by randomly selecting one job and inserting it before another randomly selected job in \( x \).

2.2. Obtain the maximum possible variation of the TFT of \( x_{\text{new}} \) using the simple greedy rule, and compute the robustness cost \( RC(x_{\text{new}}) \) of \( x_{\text{new}} \).

**Step 3:** \( \text{Iter} = \text{Iter} + 1; \text{Num} = 0 \).

**Step 4:** If \( \Delta E = RC(x_{\text{new}}) - RC(x) \leq 0 \), then go to Step 4.1; otherwise go to Step 4.2.

4.1. Let \( x = x_{\text{new}} \), \( \text{Iter} = \text{Iter} + 1; \text{Num} = 0 \).

4.2. Generate a random number \( r \) \~\( U(0, 1) \).

If \( r < (\text{Temp} / (\text{Temp}^2 + \Delta E^2)) \), then \( x = x_{\text{new}} \).

**Step 5:** If \( RC(x) < RC \), then \( x_{\text{best}} = x, RC = RC(x), \text{Num} = 0 \).

**Step 6:** If \( \text{Iter} = \text{Iter}_{\text{max}} \), then \( \text{Temp} = \text{Temp} \times \beta, \text{Iter} = 0, \text{Num} = \text{Num} + 1 \).

**Step 7:** If \( \text{Temp} < T_f \) or \( \text{Num} = \text{Num}_{\text{max}} \), then stop; otherwise go to Step 2.

4. Numerical experiments

4.1. Experimental design

Numerical experiments on randomly generated datasets were conducted to examine the computational performance of the proposed heuristics, solution robustness, tradeoff between robustness and optimality, and impact of data uncertainty on the solutions. The MILP was solved using the commercial solver, Gurobi Optimizer 5.5, and the two algorithms were coded in C computer language. Gurobi and the codes of the algorithms were executed on a personal computer with a Pentium Core 2 Duo 2.67 GHz CPU and 2 GB RAM.

The uncertain job processing times were generated as follows. First, the nominal processing time of job \( j, p_j \), was generated as a random integer from the uniform distribution in [10, 50]. Then, the maximum variation of the processing time of job \( j, p_j \), was determined as \( (s_j/100)p_j \), where \( s_j \) is a random integer generated from the uniform distribution in [10, \%]. The sizes of randomly generated problem instances are \( n = 50, 100 \), and 200. The degree of data uncertainty \( x = 30, 40, 50, \) and 60. The parameter values used in the SA-based heuristic are as follows: \( T_0 = 2000, T_f = 10, \text{Iter}_{\text{max}} = n \times 400, \text{Num}_{\text{max}} = 20 \), and \( \beta = 0.90 \). These parameter values were set according to the results of a preliminary experiment that tests a large number of combinations of parameter values.

To evaluate the robustness of the solutions, the distribution of the TFT of robust sequences was simulated by subjecting the processing times to random perturbations. Specifically, in each simulation run \( i \), the (actual) processing time \( p_j \) of each job \( j \) was independently drawn from the uniform distribution \([p_j - p_j/2, p_j + p_j/2]\). The TFTs of the robust solutions \( \text{x}(\Gamma) \) with the protection (or robustness) level \( \Gamma \) and of the optimal nominal solution \( \text{x} \) (corresponding to \( \Gamma = 0 \)) were compared using the SPT rule. The TFTs obtained in run \( i \) are denoted as \( \text{C}(\text{x}(\Gamma) \), \( \text{C}(\text{x}) \), \( \text{C}(\text{x}, i) \), respectively. A total of 40,000 runs of random processing times were generated to compare the averages, \( \text{C}((\text{x}(\cdot)), \) and standard deviations, \( \sigma(\text{x}(\cdot)), \) of the simulated TFTs of the robust solutions for different protection levels.

In addition to the objective value \( RC \), the other two performance measures of interest are robust price, \( \eta(\Gamma) \), and hedge value, \( H(\Gamma) \), defined as follows.

\[
\eta(\Gamma) = C(\text{x}(\Gamma)) - C(\text{x}),
\]

where \( C(\text{x}(\Gamma)) = \sum_{i=1}^{n} \sum_{j=1}^{n} (n - i + 1) \times p_j x_{ij}(\Gamma) \) is the TFT corresponding to \( \text{x}(\Gamma) \) and \( C(\text{x}) = \sum_{i=1}^{n} \sum_{j=1}^{n} (n - i + 1) p_j x_{ij} \) is the TFT corresponding to \( \text{x} \).

\[
H(\Gamma) = RC(\text{x}) - RC(\text{x}(\cdot)),
\]

In the above definitions, \( \eta(\Gamma) \) is equal to the price that the decision-maker needs to pay for employing the robust sequence \( \text{x}(\Gamma) \) instead of the optimal nominal sequence \( \text{x} \) in the scenario of nominal processing times (i.e., the absence of processing time deviations). \( H(\Gamma) \) equals the difference between the RC of \( \text{x} \) and that of the robust solution \( \text{x}(\Gamma) \) for a given \( \Gamma \). Moreover, \( \eta(\Gamma) \) represents the tradeoff between robustness and optimality, while \( H(\Gamma) \) can be considered as the regret of employing the sequence \( \text{x} \) in the worst-case scenario.

In the case of correlated uncertain processing times, for each job \( j \), the largest deviation from the nominal value \( p_j \) due to the impact of the uncertainty source \( s, p_j \), was determined as \( (s \times \beta)x \), where \( \beta \) is a constant parameter. For instance, if \( s = 2 \) (i.e., the second uncertainty source) and \( \beta = 10 \), then \( p_j = 20\% \) of \( p_j \). This setting implies that a common source \( s \) has the same degree of impact on all jobs. Then, the total variation was generated as
\[ p_j = (1 + r_j) \sum_{i=1}^5 p_{ij} \], where \( r_j \in [-0.1, 0.1] \). Consider the example where \( s = 1, . . . , 5 \), \( p_1 = 50 \), and \( \beta = 10 \). \( p_1 = (1 + 10\%)p_1 = (2 \times 10\%)p_1 \), \( p_2 = (3 \times 10\%)p_2 = (4 \times 10\%)p_2 = (5 \times 10\%)p_2 \). Finally, \( p_5 = (1 + r_5)(150\%)p_5 = 75(1 + r_5) \).

As for the case with group-specific correlated processing time uncertainties, for job \( j(k) \) of group \( k \), the largest deviation from the nominal value \( p_j(k) \) due to the impact of uncertainty source \( s \), \( p_j(k) \), was determined as \((s \times k \times \beta\%)p_j(k)\). For instance, if \( s = 2, k = 2 \) (i.e., the second group), and \( \beta = 5 \), then \( p_{j(2)}(2) = 20\% \) of \( p_j(k) \). In this case, a common source \( s \) has the same degree of impact on all jobs in group \( k \). Then, the total variation was generated as \( p_j(k) = (1 + r_j) \sum_{i=1}^5 p_{ij} \), where \( r_j(k) \in [-0.1, 0.1] \). Consider the example where \( s = 1, . . . , 5, k = 1, . . . , 5 \), and \( \beta = 5 \). For jobs of group \( 1 \) (i.e., \( k = 1 \)),

\[ p_{j(1)} = (1 + 10\%)p_{j(1)}; \quad p_{j(1)} = (2 \times 10\%)p_{j(1)}; \]
\[ p_{j(1)} = (3 \times 10\%)p_{j(1)}; \quad p_{j(1)} = (4 \times 10\%)p_{j(1)}; \quad p_{j(1)} = (5 \times 10\%)p_{j(1)}; \]
\[ p_{j(1)(1)} = (1 + r_{j(1)}) (150\%); \quad p_{j(1)(1)} = (1 + r_{j(1)}) (150\%); \]

Similarly, for jobs of group \( 5 \) (i.e., \( k = 5 \)),

\[ p_{j(5)} = (1 + 10\%)p_{j(5)}; \quad p_{j(5)} = (2 \times 10\%)p_{j(5)}; \]
\[ p_{j(5)} = (3 \times 10\%)p_{j(5)}; \quad p_{j(5)} = (4 \times 10\%)p_{j(5)}; \quad p_{j(5)} = (5 \times 10\%)p_{j(5)}; \]
\[ p_{j(5)} = (1 + r_{j(5)})(50 \times 100 + 150 + 200 + 250\%); \]
\[ p_{j(5)} = (1 + r_{j(5)})(150\%); \]

4.2. Experimental results

4.2.1. Effectiveness and efficiency of the proposed heuristics

To examine the effectiveness (or solution quality) of the proposed MILP, this research compared the objective values obtained by the SII and SA heuristics and those obtained by solving the MILP. The maximum computing time for Gurobi to solve the MILP was set to 14,400 s (4 h). The left part of Table 1 shows the RCs obtained by the SII and SA heuristics and the MILP for various protection levels, when \( x = 40 \). As expected, the RC increases with the protection level. The bold numbers indicate the optimal or best solutions. For the instances of \( n = 50 \) (small size), SA obtains optimal solution for all the eleven instances, while SII finds optimal solution for six out of the eleven instances. For the instances of \( n = 100 \) (medium size), nine out of the eleven instances are optimally solved by SA, but only three out of the eleven instances are solved by SII. When the problem size increases to \( n = 200 \) (large size), Gurobi gets optimal solution for five instances within the four-hour computation time, while SA obtains best solution for nine out of the eleven instances and outperforms SII, which finds optimal solution for only two instances. The above results show that SA is more effective than SII in solving the SMRSP of interest.

This research evaluated the computational efficiency of the heuristics in terms of CPU time (in seconds). The CPU times for different problem sizes and protection levels are presented on the right side of Table 1. As can be seen, SA and SII require very light computational effort, while the computation time of solving the MILP by Gurobi increases dramatically when solving large size instances (i.e., \( n = 200 \)). Particularly for the instances with

Table 1

<table>
<thead>
<tr>
<th>( \Gamma \times n )</th>
<th>Objective value (RC)</th>
<th>Computation time (s)</th>
<th>MILP (Gurobi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SII</td>
<td>SA</td>
<td>SA</td>
<td>SII</td>
</tr>
<tr>
<td>50</td>
<td>100</td>
<td>200</td>
<td>50</td>
</tr>
<tr>
<td>0</td>
<td>27,813</td>
<td>441,167</td>
<td>112,810</td>
</tr>
<tr>
<td>10</td>
<td>29,252</td>
<td>464,254</td>
<td>118,408</td>
</tr>
<tr>
<td>20</td>
<td>28,300</td>
<td>482,656</td>
<td>123,111</td>
</tr>
<tr>
<td>30</td>
<td>31,428</td>
<td>517,216</td>
<td>127,318</td>
</tr>
<tr>
<td>40</td>
<td>32,320</td>
<td>571,826</td>
<td>130,915</td>
</tr>
<tr>
<td>50</td>
<td>33,032</td>
<td>652,183</td>
<td>134,288</td>
</tr>
<tr>
<td>60</td>
<td>33,661</td>
<td>652,183</td>
<td>134,288</td>
</tr>
<tr>
<td>70</td>
<td>34,187</td>
<td>531,015</td>
<td>134,288</td>
</tr>
<tr>
<td>80</td>
<td>34,624</td>
<td>596,492</td>
<td>134,801</td>
</tr>
<tr>
<td>90</td>
<td>34,938</td>
<td>596,492</td>
<td>134,801</td>
</tr>
</tbody>
</table>

The best solution is bold face.

Table 2

<p>| SMRSP solutions (( n = 200 ).) |
|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|</p>
<table>
<thead>
<tr>
<th>( \Gamma )</th>
<th>( p_b )</th>
<th>( x = 30 )</th>
<th>( x = 40 )</th>
<th>( x = 50 )</th>
<th>( x = 60 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 0 )</td>
<td>0.5025</td>
<td>0.482</td>
<td>0.592</td>
<td>0.705</td>
<td>0.807</td>
</tr>
<tr>
<td>( 1 )</td>
<td>0.3520</td>
<td>0.226</td>
<td>0.274</td>
<td>0.315</td>
<td>0.337</td>
</tr>
<tr>
<td>( 10 )</td>
<td>0.1845</td>
<td>0.230</td>
<td>0.274</td>
<td>0.315</td>
<td>0.337</td>
</tr>
<tr>
<td>( 15 )</td>
<td>0.0818</td>
<td>0.143</td>
<td>0.178</td>
<td>0.211</td>
<td>0.222</td>
</tr>
<tr>
<td>( 20 )</td>
<td>0.2283</td>
<td>0.378</td>
<td>0.418</td>
<td>0.452</td>
<td>0.473</td>
</tr>
<tr>
<td>( 25 )</td>
<td>0.0827</td>
<td>0.602</td>
<td>0.634</td>
<td>0.666</td>
<td>0.688</td>
</tr>
<tr>
<td>( 30 )</td>
<td>0.0763</td>
<td>1.153</td>
<td>1.186</td>
<td>1.219</td>
<td>1.242</td>
</tr>
<tr>
<td>( 35 )</td>
<td>0.0023</td>
<td>1.250</td>
<td>1.283</td>
<td>1.316</td>
<td>1.348</td>
</tr>
<tr>
<td>( 40 )</td>
<td>0.0039</td>
<td>1.393</td>
<td>1.426</td>
<td>1.458</td>
<td>1.490</td>
</tr>
<tr>
<td>( 50 )</td>
<td>0.0043</td>
<td>1.599</td>
<td>1.632</td>
<td>1.665</td>
<td>1.698</td>
</tr>
<tr>
<td>( 55 )</td>
<td>0.0000</td>
<td>1.796</td>
<td>1.829</td>
<td>1.862</td>
<td>1.895</td>
</tr>
<tr>
<td>( 60 )</td>
<td>0.0000</td>
<td>1.993</td>
<td>2.026</td>
<td>2.059</td>
<td>2.092</td>
</tr>
</tbody>
</table>
n = 200, solving the MILP by GUROBI requires usually more than four hours, but SA spends less than 3 min for one instance. Owing to the enormous computation time for solving the MILP by GUROBI, it is very difficult to examine the solution quality of the two heuristics for \( n > 200 \). In summary, the numerical results demonstrate that the proposed SA is effective and efficient in solving SMRSP with practical problem sizes.

### 4.2.2. Solution robustness, robust price, and hedge value

Table 2 presents the probability bounds (PBs) of actual TFTs exceeding the optimal objective values for different protection levels in problem instances of \( n = 200 \), and \( x = 30, 40, 50, \) and 60. The PBs are computed using Theorem 3 presented in Bertsimas and Sim (2004). The corresponding PB is fairly small when \( \Gamma > 0 \) (PB \( \leq 4 \times 10^{-2} \)), which represents a sufficient protection level to hedge against uncertain job processing times. That is, the job sequences obtained corresponding to \( \Gamma = 40 \) should be robust enough in that the probability of obtaining worse TFTs is practically negligible. Moreover, a marginal decrease in PB resulting from raising the protection level over 40 may come at the expense of a large increase in objective value. Also shown in Table 2 are the RCs and the change in percentages in RC, \( \Delta RC \), which is defined as \( (RC(\Gamma) - RC(\Gamma = 0))/RC(\Gamma = 0) \times 100\% \). Recall that RC\((\Gamma = 0)\) is the optimal nominal TFT obtained using the SPT rule. To achieve the same level of robustness (or protection), one has to allow a larger increase in RC when the processing times are subject to a higher degree of uncertainty (i.e., a larger \( \Gamma \)).

![Distributions of simulated TFTs (n = 200 and x = 60)](image)

For instance, when \( \Gamma = 30 \) (or PB is \( 1.7 \times 10^{-3} \)), \( \Delta RC \) increases from 11.153\% to 17.044\% as \( x \) increases from 30 to 60. In addition, to increase the protection level, say from 10 to 15, more must be sacrificed in RC when the degree of uncertainty is increased from \( x = 30 \) to \( x = 60 \); that is, when \( x = 30 \), the difference in RC is 8394 (=468266–459982), which increases to 12,455 (=482,834 – 470,379) when \( x = 60 \).

The simulation results are reported in Table 3. First, because of the smaller averages and standard deviations of the simulated TFTs of the robust solutions (i.e., \( \Gamma > 0 \)), they are found to outperform the optimal nominal solutions (i.e., \( \Gamma = 0 \)). Moreover, when the solution is more robust (i.e., \( \Gamma \) is larger), the standard deviation of simulated TFTs is decreased, implying lower performance variability (Bertsimas & Sim, 2004). Fig. 1 depicts the distributions of the simulated TFTs for the solutions corresponding to \( \Gamma = 0, 20, 40, \) and 60, when \( n = 100 \) and \( x = 60 \). These distributions also demonstrate that as \( \Gamma \) increases, the central tendency of simulated TFTs is more significant (i.e., less variability). The results did not seem to depend on the specific instances generated and tested.

Table 4 displays the robust prices and hedge values corresponding to different protection levels in the problem instances of \( n = 200 \), and \( x = 30, 40, 50, \) and 60. As expected, the robust price increases with both the level of protection (\( \Gamma \)) and the degree of processing time uncertainty (\( x \)), indicating that when the solution is more robust and the degree of processing time uncertainty is higher, there is a need to tradeoff a larger solution quality for the

### Table 3

<table>
<thead>
<tr>
<th>( \Gamma )</th>
<th>( x = 30 )</th>
<th>( x = 40 )</th>
<th>( x = 50 )</th>
<th>( x = 60 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>896.329.30</td>
<td>77,781.01</td>
<td>1,207,733.06</td>
<td>1,517,428.81</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>92,567.12</td>
<td>97,178.25</td>
<td>92,601.39</td>
<td>111,150.45</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>93,980.84</td>
<td>75,056.49</td>
<td>92,601.39</td>
<td>140,675.18</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>951,541.64</td>
<td>74,301.93</td>
<td>90,901.99</td>
<td>113,150.45</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>947,496.02</td>
<td>70,506.99</td>
<td>89,891.61</td>
<td>148,067.59</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>943,809.28</td>
<td>70,769.82</td>
<td>88,537.13</td>
<td>147,311.49</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>942,072.07</td>
<td>71,653.61</td>
<td>87,988.92</td>
<td>144,269.10</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>958,596.12</td>
<td>70,569.99</td>
<td>86,975.65</td>
<td>143,286.24</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>934,464.40</td>
<td>70,261.83</td>
<td>85,770.97</td>
<td>142,504.94</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>933,095.17</td>
<td>69,746.24</td>
<td>85,770.97</td>
<td>139,504.94</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>927,274.30</td>
<td>69,282.72</td>
<td>84,223.88</td>
<td>138,070.01</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>926,451.70</td>
<td>68,021.74</td>
<td>83,753.71</td>
<td>135,737.59</td>
</tr>
<tr>
<td>( \xi(x'(\Gamma)) )</td>
<td>927,274.30</td>
<td>68,286.31</td>
<td>83,753.71</td>
<td>135,737.59</td>
</tr>
<tr>
<td>( \sigma(x'(\Gamma)) )</td>
<td>927,274.30</td>
<td>68,286.31</td>
<td>83,753.71</td>
<td>135,737.59</td>
</tr>
</tbody>
</table>

Fig. 1. Distributions of simulated TFTs (\( n = 200 \) and \( x = 60 \)).
Table 4
Robust price and hedge value (n = 200).

| Γ | x = 30 | | x = 40 | | x = 50 | | x = 60 |
|---|---|---|---|---|---|---|
| n(Γ) | H(Γ) | n(Γ) | H(Γ) | n(Γ) | H(Γ) | n(Γ) | H(Γ) |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 502 | 967 | 685 | 1308 | 1004 | 2107 | 1483 | 2881 |
| 10 | 1153 | 1910 | 1570 | 2970 | 2719 | 4355 | 3209 | 6029 |
| 15 | 1846 | 2676 | 2580 | 4240 | 4310 | 6305 | 4865 | 8494 |
| 20 | 2446 | 3267 | 3466 | 5222 | 5350 | 7633 | 5944 | 10258 |
| 25 | 3063 | 3850 | 4130 | 6089 | 6468 | 8730 | 6570 | 11693 |
| 30 | 3370 | 4388 | 4635 | 6809 | 6940 | 9880 | 7874 | 12746 |
| 35 | 3674 | 4803 | 5190 | 7375 | 7186 | 10594 | 8210 | 13149 |
| 40 | 4040 | 4962 | 5380 | 7648 | 7281 | 10594 | 8210 | 13149 |
| 45 | 3985 | 5105 | 5753 | 7760 | 7803 | 10852 | 9754 | 14563 |
| 50 | 3916 | 5226 | 5778 | 7785 | 7463 | 10414 | 9030 | 13109 |
| 55 | 3901 | 5194 | 5831 | 7736 | 7521 | 10551 | 9715 | 13536 |
| 60 | 3994 | 5074 | 5252 | 7501 | 7297 | 10428 | 9711 | 13592 |

Table 5
Results of SMRSP with correlated uncertainties (K = 5, S = 5, and n = 200).

| Γ | β = 5 | | β = 10 |
|---|---|---|
| n(Γ) | H(Γ) | n(Γ) | H(Γ) |
| 0 | 0 | 0 | 0 |
| 5 | 441,167 | 0 | 0 |
| 10 | 463,504 | 616 | 2379 |
| 15 | 484,895 | 753 | 3809 |
| 20 | 505,986 | 817 | 4602 |
| 25 | 526,833 | 924 | 4995 |
| 30 | 547,207 | 868 | 5152 |
| 35 | 566,721 | 871 | 5414 |
| 40 | 585,680 | 814 | 5258 |
| 45 | 604,215 | 850 | 5192 |
| 50 | 622,359 | 856 | 5040 |
| 55 | 640,030 | 882 | 4773 |
| 60 | 657,163 | 941 | 4404 |
| 65 | 673,678 | 884 | 3904 |

Table 6
Results of SMRSP with group-specific correlated uncertainties (K = 5, S = 5, and n = 200).

| Γ | β = 5 | | β = 10 |
|---|---|---|
| n(Γ) | H(Γ) | n(Γ) | H(Γ) |
| 0 | 0 | 0 | 0 |
| 5 | 441,167 | 0 | 0 |
| 10 | 552,060 | 819 | 15,251 |
| 15 | 660,731 | 889 | 23,901 |
| 20 | 768,447 | 978 | 27,800 |
| 25 | 874,906 | 966 | 29,984 |
| 30 | 978,770 | 1025 | 29,251 |
| 35 | 1,080,510 | 1127 | 28,988 |
| 40 | 1,176,940 | 1371 | 28,730 |
| 45 | 1,270,462 | 1480 | 28,214 |
| 50 | 1,363,064 | 1561 | 28,162 |
| 55 | 1,452,401 | 1788 | 28,607 |
| 60 | 1,540,406 | 1776 | 24,795 |

5. Concluding remarks

By integrating the linear program for determining the worst-case TFT within the integer program of the TFT-minimization SMSP, this research proposed the MILP for the SMRSP, where uncertain job processing times are represented as intervals. To efficiently solve the SMRSP, SI and SA heuristics were developed to obtain robust job schedules with minimum worst-case TFT. Experimental results demonstrate that the proposed heuristics are effective and efficient in solving SMRSP with practical problem sizes. Simulation results indicate that the robust sequences outperform the optimal nominal sequences under different protection levels. Also shown in the numerical example is that the proposed approach avoids the over-conservativeness in the conventional robust scheduling methods and allows a tradeoff between robustness and optimality.

This research contributes to the growing body of work on both theoretically and practically useful robust optimization approaches to production scheduling problems. A number of interesting lines of research may be pursued using the model and algorithm developed in this research. First, the formal proof of NP-hardness of the problem should be addressed in our subsequent research. Second, while the proposed heuristics are able to obtain optimal or near-optimal solutions on randomly generated test problems with practical sizes, their effectiveness and efficiency on large problem instances deserves further investigation. Moreover, the proposed methods may be extended to the SMSP with other objectives, such as makespan, number of completed jobs, total tardiness, or their weighted combinations. They can also be adapted to other production scheduling problems, for instance, parallel machine scheduling and flow shop scheduling problems.

References


