The Stability Box for Minimizing Total Flow Time under Uncertain Data

Chapter · February 2013

DOI: 10.1007/978-3-642-34336-0_3

3 authors:

Yuri N. Sotskov
United Institute Of Informatics Problems
135 PUBLICATIONS **1,482** CITATIONS
SEE PROFILE

Tsung-Chyan Lai
Harbin Engineering University
34 PUBLICATIONS **396** CITATIONS
SEE PROFILE

Frank Werner
Otto-von-Guericke-Universität Magdeburg
346 PUBLICATIONS **2,588** CITATIONS
SEE PROFILE

The Stability Box for Minimizing Total Weighted Flow Time under Uncertain Data

Yuri N. Sotskov[1], Tsung-Chyan Lai[2], and Frank Werner[3]

[1] United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova Str 6, Minsk, Belarus
sotskov@newman.bas-net.by

[2] Department of Business Administration, National Taiwan University, Roosevelt Rd 85, Taipei, Taiwan
tclai@ntu.edu.tw

[3] Faculty of Mathematics, Otto-von-Guericke-University, Magdeburg, Germany
frank.werner@ovgu.de

Abstract. We consider an uncertain single-machine scheduling problem, in which the processing time of a job can take any real value from a given closed interval. The criterion is to minimize the sum of weighted completion times of the n jobs, a weight being associated with each job. For a job permutation, we study the stability box, which is a subset of the stability region. We derive an $O(n \log n)$ algorithm for constructing a job permutation with the largest dimension and volume of a stability box. The efficiency of such a permutation is demonstrated via a simulation on a set of randomly generated instances with $1000 \leq n \leq 2000$. If several permutations have the largest dimension and volume of a stability box, the developed algorithm selects one of them due to a mid-point heuristic.

Keywords: Single-machine scheduling, Uncertain data, Total weighted flow time, Stability analysis.

1 Introduction

In real-life scheduling, the numerical data are usually uncertain. A stochastic [6] or a fuzzy method [8] are used when the job processing times may be defined as random variables or as fuzzy numbers. If these times may be defined neither as random variables with known probability distributions nor as fuzzy numbers, other methods are needed to solve a scheduling problem under uncertainty [1, 7, 13]. The robust method [1–3] assumes that the decision-maker prefers a schedule hedging against the worst-case scenario. The stability method [4, 5, 10–13] combines a stability analysis, a multi-stage decision framework and the solution concept of a minimal dominant set of semi-active schedules.

In this paper, we implement the stability method for a single-machine problem with interval processing times of the n jobs (Section 2). In Section 3, we derive an $O(n \log n)$ algorithm for constructing a job permutation with the largest dimension and volume of a stability box. Computational results are presented in Section 4. We conclude with Section 5.

2 Problem Setting

The jobs $\mathcal{J} = \{J_1, J_2, ..., J_n\}$, $n \geq 2$, have to be processed on a single machine, a positive weight w_i being given for any job $J_i \in \mathcal{J}$. The processing time p_i of a job J_i can take any real value from a given segment $[p_i^L, p_i^U]$, where $0 \leq p_i^L \leq p_i^U$. The exact value $p_i \in [p_i^L, p_i^U]$ may remain unknown until the completion of the job $J_i \in \mathcal{J}$. Let $T = \{p \in R_+^n \mid p_i^L \leq p_i \leq p_i^U,\ i \in \{1, 2, \ldots, n\}\}$ denote the set of vectors $p = (p_1, p_2, \ldots, p_n)$ (scenarios) of the possible job processing times. $S = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ denotes the set of permutations $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n})$ of the jobs $\mathcal{J}$. Problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ is to find an optimal permutation $\pi_t \in S$:

$$\sum_{J_i \in \mathcal{J}} w_i C_i(\pi_t, p) = \gamma_p^t = \min_{\pi_k \in S} \left\{ \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p) \right\}. \quad (1)$$

Hereafter, $C_i(\pi_k, p) = C_i$ is the completion time of job $J_i \in \mathcal{J}$ in a semi-active schedule [6, 13] defined by the permutation π_k.

Since a factual scenario $p \in T$ is unknown before scheduling, the completion time C_i of a job $J_i \in \mathcal{J}$ can be determined after the schedule execution. Therefore, one cannot calculate the value γ_p^k of the objective function

$$\gamma = \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p)$$

for a permutation $\pi_k \in S$ before the schedule realization.

However, one must somehow define a schedule before to realize it. So, the problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ of finding an optimal permutation $\pi_t \in S$ defined in (1) is not correct. In general, one can find only a heuristic solution (a job permutation) to problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ the efficiency of which may be estimated either analytically or via a simulation.

In the deterministic case, when a scenario $p \in T$ is fixed before scheduling (i.e., equalities $p_i^L = p_i^U = p_i$ hold for each job $J_i \in \mathcal{J}$), problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ reduces to the classical problem $1|| \sum w_i C_i$. In contrast to the uncertain problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$, problem $1|| \sum w_i C_i$ is called deterministic. The deterministic problem $1|| \sum w_i C_i$ is correct and can be solved exactly in $O(n \log n)$ time [9] due to the necessary and sufficient condition (2) for the optimality of a permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$:

$$\frac{w_{k_1}}{p_{k_1}} \geq \frac{w_{k_2}}{p_{k_2}} \geq \ldots \geq \frac{w_{k_n}}{p_{k_n}}, \quad (2)$$

where the strict inequality $p_{k_i} > 0$ holds for each job $J_{k_i} \in \mathcal{J}$. Using the sufficiency of condition (2), problem $1|| \sum w_i C_i$ can be solved to optimality by the weighted shortest processing time rule: process the jobs $\mathcal{J}$ in non-increasing order of their weight-to-process ratios $\frac{w_{k_i}}{p_{k_i}}$, $J_{k_i} \in \mathcal{J}$.

3 The Stability Box

In [12], the stability box $\mathcal{SB}(\pi_k, T)$ within a set of scenarios T has been defined for a permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$. To present the definition of the stability box $\mathcal{SB}(\pi_k, T)$, we need the following notations.

We denote $\mathcal{J}(k_i) = \{J_{k_1}, J_{k_2}, \ldots, J_{k_{i-1}}\}$ and $\mathcal{J}[k_i] = \{J_{k_{i+1}}, J_{k_{i+2}}, \ldots, J_{k_n}\}$. Let S_{k_i} denote the set of permutations $(\pi(\mathcal{J}(k_i)), J_{k_i}, \pi(\mathcal{J}[k_i])) \in S$ of the jobs $\mathcal{J}$, $\pi(\mathcal{J}')$ being a permutation of the jobs $\mathcal{J}' \subset \mathcal{J}$. Let N_k denote a subset of set $N = \{1, 2, \ldots, n\}$. The notation $1|p|\sum w_i C_i$ will be used for indicating an instance with a fixed scenario $p \in T$ of the deterministic problem $1||\sum w_i C_i$.

Definition 1. [12] *The maximal closed rectangular box*

$$\mathcal{SB}(\pi_k, T) = \times_{k_i \in N_k}[l_{k_i}, u_{k_i}] \subseteq T$$

is a stability box of permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$, if permutation $\pi_e = (J_{e_1}, J_{e_2}, \ldots, J_{e_n}) \in S_{k_i}$ being optimal for the instance $1|p|\sum w_i C_i$ with a scenario $p = (p_1, p_2, \ldots, p_n) \in T$ remains optimal for the instance $1|p'|\sum w_i C_i$ with a scenario $p' \in \{\times_{j=1}^{i-1}[p_{k_j}, p_{k_j}]\} \times [l_{k_i}, u_{k_i}] \times \{\times_{j=i+1}^{n}[p_{k_j}, p_{k_j}]\}$ for each $k_i \in N_k$. If there does not exist a scenario $p \in T$ such that permutation π_k is optimal for the instance $1|p|\sum w_i C_i$, then $\mathcal{SB}(\pi_k, T) = \emptyset$.

The maximality of the closed rectangular box $\mathcal{SB}(\pi_k, T) = \times_{k_i \in N_k}[l_{k_i}, u_{k_i}]$ in Definition 1 means that the box $\mathcal{SB}(\pi_k, T) \subseteq T$ has both a maximal possible dimension $|N_k|$ and a maximal possible volume.

For any scheduling instance, the stability box is a subset of the stability region [13, 14]. However, we substitute the stability region by the stability box, since the latter is easy to compute [11, 12]. In [11], a branch-and-bound algorithm has been developed to select a permutation in the set S with the largest volume of a stability box. If several permutations have the same volume of the stability box, the algorithm from [11] selects one of them due to simple heuristics. The efficiency of the constructed permutations has been demonstrated on a set of randomly generated instances with $5 \leq n \leq 100$.

In [12], an $O(n \log n)$ algorithm has been developed for calculating a stability box $\mathcal{SB}(\pi_k, T)$ for the fixed permutation $\pi_k \in S$ and an $O(n^2)$ algorithm has been developed for selecting a permutation in the set S with the largest dimension and volume of a stability box. The efficiency of these algorithms was demonstrated on a set of randomly generated instances with $10 \leq n \leq 1000$. All algorithms developed in [11, 12] use the precedence-dominance relation on the set of jobs $\mathcal{J}$ and the solution concept of a minimal dominant set $S(T) \subseteq S$ defined as follows.

Definition 2. [10] *The set of permutations $S(T) \subseteq S$ is a minimal dominant set for a problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$, if for any fixed scenario $p \in T$, the set $S(T)$ contains at least one optimal permutation for the instance $1|p|\sum w_i C_i$, provided that any proper subset of set $S(T)$ loses such a property.*

Definition 3. [10] *Job J_u dominates job J_v, if there exists a minimal dominant set $S(T)$ for the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ such that job J_u precedes job J_v in every permutation of the set $S(T)$.*

Theorem 1. [10] *For the problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$, *job* J_u *dominates job* J_v *if and only if inequality (3) holds:*

$$\frac{w_u}{p_u^U} \ge \frac{w_v}{p_v^L}. \tag{3}$$

Due to Theorem 1 proven in [10], we can obtain a compact presentation of a minimal dominant set $S(T)$ in the form of a digraph $(\mathcal{J}, \mathcal{A})$ with the vertex set $\mathcal{J}$ and the arc set $\mathcal{A}$. To this end, we can check inequality (3) for each pair of jobs from the set $\mathcal{J}$ and construct a dominance digraph $(\mathcal{J}, \mathcal{A})$ of the precedence-dominance relation on the set of jobs $\mathcal{J}$ as follows. The arc (J_u, J_v) belongs to the set $\mathcal{A}$ if and only if inequality (3) holds. The construction of the digraph $(\mathcal{J}, \mathcal{A})$ takes $O(n^2)$ time.

3.1 Illustrative Example

For the sake of simplicity of the calculation, we consider a special case $1|p_i^L \le p_i \le p_i^U|\sum C_i$ of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ when each job $J_i \in \mathcal{J}$ has a weight w_i equal to one. From condition (2), it follows that the deterministic problem $1||\sum C_i$ can be solved to optimality by the shortest processing time rule: process the jobs in non-decreasing order of their processing times p_{k_i}, $J_{k_i} \in \mathcal{J}$.

A set of scenarios T for Example 1 of the uncertain problem $1|p_i^L \le p_i \le p_i^U|\sum C_i$ is defined in columns 1 and 2 in Table 1.

Table 1. Data for calculating $\mathcal{SB}(\pi_1, T)$ for Example 1.

	1	2	3	4	5	6	7	8
i	p_i^L	p_i^U	$\frac{w_i}{p_i^U}$	$\frac{w_i}{p_i^L}$	d_i^-	d_i^+	$\frac{w_i}{d_i^+}$	$\frac{w_i}{d_i^-}$
1	2	3	$\frac{1}{3}$	0.5	1	0.5	2	1
2	1	9	$\frac{1}{9}$	1	$\frac{1}{6}$	$\frac{1}{3}$	3	6
3	8	8	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{6}$	$\frac{1}{9}$	9	6
4	6	10	0.1	$\frac{1}{6}$	0.1	$\frac{1}{9}$	9	10
5	11	12	$\frac{1}{12}$	$\frac{1}{11}$	0.1	$\frac{1}{11}$	11	10
6	10	19	$\frac{1}{19}$	0.1	$\frac{1}{15}$	$\frac{1}{12}$	12	15
7	17	19	$\frac{1}{19}$	$\frac{1}{17}$	$\frac{1}{15}$	$\frac{1}{19}$	19	15
8	15	20	$\frac{1}{20}$	$\frac{1}{15}$	$\frac{1}{20}$	$\frac{1}{19}$	19	20

In [12], formula (9) has been proven. To use it for calculating the stability box $\mathcal{SB}(\pi_k, T)$, one has to define for each job $J_{k_i} \in \mathcal{J}$ the maximal range $[l_{k_i}, u_{k_i}]$ of possible variations of the processing time p_{k_i} preserving the optimality of permutation π_k (see Definition 1). Due to the additivity of the objective function $\gamma = \sum_{J_i \in \mathcal{J}} w_iC_i(\pi_k, p)$, the lower bound $d_{k_i}^-$ on the maximal range of possible variations of the weight-to-process ratio $\frac{w_{k_i}}{p_{k_i}}$ preserving the optimality of the

permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ is calculated as follows:

$$d^-_{k_i} = \max\left\{\frac{w_{k_i}}{p^U_{k_i}}, \max_{i<j\le n}\left\{\frac{w_{k_j}}{p^L_{k_j}}\right\}\right\}, i \in \{1, 2, \ldots, n-1\}, \tag{4}$$

$$d^-_{k_n} = \frac{w_{k_n}}{p^U_{k_n}}. \tag{5}$$

The upper bound $d^+_{k_i}, J_{k_i} \in \mathcal{J}$, on the maximal range of possible variations of the weight-to-process ratio $\frac{w_{k_i}}{p_{k_i}}$ preserving the optimality of the permutation π_k is calculated as follows:

$$d^+_{k_i} = \min\left\{\frac{w_{k_i}}{p^L_{k_i}}, \min_{1\le j<i}\left\{\frac{w_{k_j}}{p^U_{k_j}}\right\}\right\}, i \in \{2, 3, \ldots, n\}, \tag{6}$$

$$d^+_{k_1} = \frac{w_{k_1}}{p^L_{k_1}}. \tag{7}$$

For Example 1, the values $d^-_{k_i}, i \in \{1, 2, \ldots, 8\}$, defined in (4) and (5) are given in column 5 of Table 1. The values $d^+_{k_i}$ defined in (6) and (7) are given in column 6. In [12], the following claim has been proven.

Theorem 2. [12] *If there is no job J_{k_i}, $i \in \{1, 2, \ldots, n-1\}$, in permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ such that inequality*

$$\frac{w_{k_i}}{p^L_{k_i}} < \frac{w_{k_j}}{p^U_{k_j}} \tag{8}$$

holds for at least one job J_{k_j}, $j \in \{i+1, i+2, \ldots, n\}$, then the stability box $\mathcal{SB}(\pi_k, T)$ is calculated as follows:

$$\mathcal{SB}(\pi_k, T) = \times_{d^-_{k_i} \le d^+_{k_i}}\left[\frac{w_{k_i}}{d^+_{k_i}}, \frac{w_{k_i}}{d^-_{k_i}}\right]. \tag{9}$$

Otherwise, $\mathcal{SB}(\pi_k, T) = \emptyset$.

Using Theorem 2, we can calculate the stability box $\mathcal{SB}(\pi_1, T)$ of the permutation $\pi_1 = (J_1, J_2, \ldots, J_8)$ in Example 1. First, we convince that there is no job J_{k_i}, $i \in \{1, 2, \ldots, n-1\}$, with inequality (8). Due to Theorem 2, $\mathcal{SB}(\pi_1, T) \neq \emptyset$.

The bounds $\frac{w_{k_i}}{d^+_{k_i}}$ and $\frac{w_{k_i}}{d^-_{k_i}}$ on the maximal possible variations of the processing times p_{k_i} preserving the optimality of the permutation π_1 are given in columns 7 and 8 of Table 1. The maximal ranges (segments) of possible variations of the job processing times within the stability box $\mathcal{SB}(\pi_1, T)$ are dashed in a coordinate system in Fig. 1, where the abscissa axis is used for indicating the job processing times and the ordinate axis for the jobs from set $\mathcal{J}$.

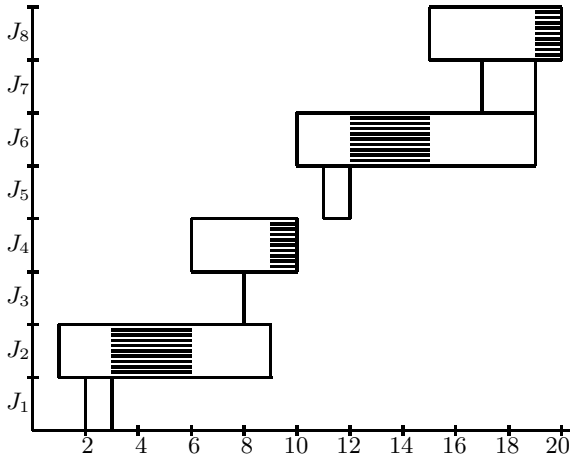

Fig. 1. The maximal ranges $[l_i, u_i]$ of possible variations of the processing times $p_i, i \in \{2, 4, 6, 8\}$, within the stability box $\mathcal{SB}(\pi_1, T)$ are dashed.

Using formula (9), we obtain the stability box for permutation π_1 as follows:

$$\begin{aligned}\mathcal{SB}(\pi_1, T) &= \left[\frac{w_2}{d_2^+}, \frac{w_2}{d_2^-}\right] \times \left[\frac{w_4}{d_4^+}, \frac{w_4}{d_4^-}\right] \times \left[\frac{w_6}{d_6^+}, \frac{w_6}{d_6^-}\right] \times \left[\frac{w_8}{d_8^+}, \frac{w_8}{d_8^-}\right] \\ &= [3, 6] \times [9, 10] \times [12, 15] \times [19, 20].\end{aligned}$$

Each job J_i, $i \in \{1, 3, 5, 7\}$, has an empty range of possible variations of the time p_i preserving the optimality of permutation π_1 since $d_i^- > d_i^+$ (see columns 5 and 6 in Table 1). The dimension of the stability box $\mathcal{SB}(\pi_1, T)$ is equal to $4 = 8 - 4$. The volume of this stability box is equal to $9 = 3 \cdot 1 \cdot 3 \cdot 1$.

In [12], an $O(n \log n)$ algorithm STABOX has been developed for calculating the stability box $\mathcal{SB}(\pi_k, T)$ for a fixed permutation $\pi_k \in S$.

For practice, the value of the relative volume of a stability box is more useful than its absolute value. The relative volume of a stability box is defined as the product of the fractions

$$\left(\frac{w_i}{d_i^-} - \frac{w_i}{d_i^+}\right) : \left(p_i^U - p_i^L\right) \tag{10}$$

for the jobs $J_i \in \mathcal{J}$ having non-empty ranges $[l_i, u_i]$ of possible variations of the processing time p_i (inequality $d_i^- \leq d_i^+$ must hold for such a job $J_i \in \mathcal{J}$).

The relative volume of the stability box for permutation π_1 in Example 1 is calculated as follows: $\frac{3}{8} \cdot \frac{1}{4} \cdot \frac{3}{9} \cdot \frac{1}{5} = \frac{1}{160}$. The absolute volume of the whole box of the scenarios T is equal to $2880 = 1 \cdot 8 \cdot 4 \cdot 1 \cdot 9 \cdot 2 \cdot 5$. The relative volume of the rectangular box T is defined as 1.

3.2 Properties of a Stability Box

A job permutation in the set S with a larger dimension and a larger volume of the stability box seems to be more efficient than one with a smaller dimension and (or) a smaller volume of stability box.

We investigate properties of a stability box, which allow us to derive an $O(n \log n)$ algorithm for choosing a permutation $\pi_t \in S$ which has

(a) the largest dimension $|N_t|$ of the stability box $\mathcal{SB}(\pi_t, T) = \times_{t_i \in N_t}[l_{t_i}, u_{t_i}] \subseteq T$ among all permutations $\pi_k \in S$ and

(b) the largest volume of the stability box $\mathcal{SB}(\pi_t, T)$ among all permutations $\pi_k \in S$ having the largest dimension $|N_k| = |N_t|$ of their stability boxes $\mathcal{SB}(\pi_k, T)$.

Definition 1 implies the following claim.

Property 1. For any jobs $J_i \in \mathcal{J}$ and $J_v \in \mathcal{J}$, $v \neq i$,

$$(w_i/u_i, w_i/l_i) \bigcap \left[w_v/p_v^U, w_v/p_v^L\right] = \varnothing.$$

Let S^{max} denote the subset of all permutations π_t in the set S possessing properties (a) and (b). Using Property 1, we shall show how to define the relative order of a job $J_i \in \mathcal{J}$ with respect to a job $J_v \in \mathcal{J}$ for any $v \neq i$ in a permutation $\pi_t = (J_{t_1}, J_{t_2}, \ldots, J_{t_n}) \in S^{max}$. To this end, we have to treat all three possible cases (I)–(III) for the intersection of the open interval $\left(\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}\right)$ and the closed interval $\left[\frac{w_v}{p_v^U}, \frac{w_v}{p_v^L}\right]$. The order of the jobs J_i and J_v in the desired permutation $\pi_t \in S^{max}$ may be defined in the cases (I)–(III) using the following rules.

Case (I) is defined by the inequalities

$$\frac{w_v}{p_v^U} \leq \frac{w_i}{p_i^U}, \quad \frac{w_v}{p_v^L} \leq \frac{w_i}{p_i^L} \tag{11}$$

provided that at least one of inequalities (11) is strict.

In case (I), the desired order of the jobs J_v and J_i in permutation $\pi_t \in S^{max}$ may be defined by a strict inequality from (11): job J_v proceeds job J_i in permutation π_t.

Indeed, if job J_i proceeds job J_v, then the maximal ranges $[l_i, u_i]$ and $[l_v, u_v]$ of possible variations of the processing times p_i and p_v preserving the optimality of $\pi_k \in S$ are both empty (it follows from equalities (4) – (7) and (9)). Thus, the following property is proven.

Property 2. For case (I), there exists a permutation $\pi_t \in S^{max}$, in which job J_v proceeds job J_i.

Case (II) is defined by the equalities

$$\frac{w_v}{p_v^U} = \frac{w_i}{p_i^U}, \quad \frac{w_v}{p_v^L} = \frac{w_i}{p_i^L}. \tag{12}$$

Property 3. For case (II), there exists a permutation $\pi_t \in S^{max}$, in which the jobs J_i and J_v are located adjacently: $i = t_r$ and $v = t_{r+1}$.

Proof. The maximal ranges $[l_i, u_i]$ and $[l_v, u_v]$ of possible variations of the processing times p_i and p_v preserving the optimality of $\pi_k \in S$ are both empty.

If job J_i and job J_v are located adjacently, then the maximal range $[l_u, u_u]$ of possible variations of the processing time p_u for any job $J_u \in \mathcal{J} \setminus \{J_i, J_v\}$ preserving the optimality of the permutation π_k is no less than that if at least one job $J_w \in \mathcal{J} \setminus \{J_i, J_v\}$ is located between job J_i and job J_v. ■

If equalities (12) hold, one can restrict the search for a permutation $\pi_t \in S^{max}$ by a subset of permutations in set S with the adjacently located jobs J_i and J_v (Property 3). Moreover, the order of such jobs $\{J_i, J_v\}$ does not influence the volume of the stability box and its dimension.

Remark 1. Due to Property 3, while looking for a permutation $\pi_t \in S^{max}$, we shall treat a pair of jobs $\{J_i, J_v\}$ satisfying (12) as one job (either job J_i or J_v).

Case (III) is defined by the strict inequalities

$$\frac{w_v}{p_v^U} > \frac{w_i}{p_i^U}, \quad \frac{w_v}{p_v^L} < \frac{w_i}{p_i^L}. \tag{13}$$

For a job $J_i \in \mathcal{J}$ satisfying case (III), let $\mathcal{J}(i)$ denote the set of all jobs $J_v \in \mathcal{J}$, for which the strict inequalities (13) hold.

Property 4. (i) For a fixed permutation $\pi_k \in S$, job $J_i \in \mathcal{J}$ may have at most one maximal segment $[l_i, u_i]$ of possible variations of the processing time $p_i \in [p_i^L, p_i^U]$ preserving the optimality of permutation π_k.

(ii) For the whole set of permutations S, only in case (III), a job $J_i \in \mathcal{J}$ may have more than one (namely: $|\mathcal{J}(i)| + 1 > 1$) maximal segments $[l_i, u_i]$ of possible variations of the time $p_i \in [p_i^L, p_i^U]$ preserving the optimality of this or that particular permutation from the set S.

Proof. Part *(i)* of Property 4 follows from the fact that a non-empty maximal segment $[l_i, u_i]$ (if any) is uniquely determined by the subset $\mathcal{J}^-(i)$ of jobs located before job J_i in permutation π_k and the subset $\mathcal{J}^+(i)$ of jobs located after job J_i. The subsets $\mathcal{J}^-(i)$ and $\mathcal{J}^+(i)$ are uniquely determined for a fixed permutation $\pi_k \in S$ and a fixed job $J_i \in \mathcal{J}$.

Part *(ii)* of Property 4 follows from the following observations. If the open interval $\left(\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}\right)$ does not intersect with the closed interval $\left[\frac{w_v}{p_v^U}, \frac{w_v}{p_v^L}\right]$ for each job $J_v \in \mathcal{J}$, then there exists a permutation $\pi_t \in S^{max}$ with a maximal segment $[l_i, u_i] = \left[w_i/p_i^U, w_i/p_i^L\right]$ preserving the optimality of permutation π_t.

Each job $J_v \in \mathcal{J}$ with a non-empty intersection $\left(\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}\right) \bigcap \left[\frac{w_v}{p_v^U}, \frac{w_v}{p_v^L}\right] \neq \emptyset$ satisfying inequalities (11) (case (I)) or equalities (12) (case (II)) may shorten the above maximal segment $[l_i, u_i]$ and cannot generate a new possible maximal segment. In case (III), a job J_v satisfying inequalities (13) may generate a

new possible maximal segment $[l_i, u_i]$ just for job J_i satisfying the same strict inequalities (13) as job J_v does. So, the cardinality $|\mathcal{L}(i)|$ of the whole set $\mathcal{L}(i)$ of such segments $[l_i, u_i]$ is not greater than $|\mathcal{J}(i)| + 1$. ■

Let $\mathcal{L}$ denote the set of all maximal segments $[l_i, u_i]$ of possible variations of the processing times p_i for all jobs $J_i \in \mathcal{J}$ preserving the optimality of a permutation $\pi_t \in S^{max}$. Using Property 4 and induction on the cardinality $|\mathcal{J}(i)|$, we proved

Property 5. $|\mathcal{L}| \leq n$.

3.3 A Job Permutation with the Largest Volume of a Stability Box

The above properties allows us to derive an $O(n \log n)$ algorithm for calculating a permutation $\pi_t \in S^{max}$ with the largest dimension $|N_t|$ and the largest volume of a stability box $\mathcal{SB}(\pi_t, T)$.

Algorithm MAX-STABOX

Input: Segments $[p_i^L, p_i^U]$, weights w_i, $J_i \in \mathcal{J}$.
Output: Permutation $\pi_t \in S^{max}$, stability box $\mathcal{SB}(\pi_t, T)$.

Step 1: Construct the list $\mathcal{M}(U) = (J_{u_1}, J_{u_2}, \ldots, J_{u_n})$ and the list $\mathcal{W}(U) = (\frac{w_{u_1}}{p_{u_1}^U}, \frac{w_{u_2}}{p_{u_2}^U}, \ldots, \frac{w_{u_n}}{p_{u_n}^U})$ in non-increasing order of $\frac{w_{u_r}}{p_{u_r}^U}$. Ties are broken via decreasing $\frac{w_{u_r}}{p_{u_r}^L}$.

Step 2: Construct the list $\mathcal{M}(L) = (J_{l_1}, J_{l_2}, \ldots, J_{l_n})$ and the list $\mathcal{W}(L) = (\frac{w_{l_1}}{p_{l_1}^L}, \frac{w_{l_2}}{p_{l_2}^L}, \ldots, \frac{w_{l_n}}{p_{l_n}^L})$ in non-increasing order of $\frac{w_{l_r}}{p_{l_r}^L}$. Ties are broken via decreasing $\frac{w_{l_r}}{p_{l_r}^U}$.

Step 3: FOR $j = 1$ to $j = n$ DO
compare job J_{u_j} and job J_{l_j}.

Step 4: IF $J_{u_j} = J_{l_j}$ THEN job J_{u_j} has to be located in position j in permutation $\pi_t \in S^{max}$ GOTO step 8.

Step 5: ELSE job $J_{u_j} = J_i$ satisfies inequalities (13). Construct the set $\mathcal{J}(i) = \{J_{u_{r+1}}, J_{u_{r+2}}, \ldots, J_{l_{k+1}}\}$ of all jobs J_v satisfying inequalities (13), where $J_i = J_{u_j} = J_{l_k}$.

Step 6: Choose the largest range $[l_{u_j}, u_{u_j}]$ among those generated for the job $J_{u_j} = J_i$.

Step 7: Partition the set $\mathcal{J}(i)$ into the subsets $\mathcal{J}^-(i)$ and $\mathcal{J}^+(i)$ generating the largest range $[l_{u_j}, u_{u_j}]$. Set $j = k + 1$ GOTO step 4.

Step 8: Set $j := j + 1$ GOTO step 4.
END FOR

Step 9: Construct the permutation $\pi_t \in S^{max}$ via putting the jobs $\mathcal{J}$ in the positions defined in steps 3 – 8.

Step 10: Construct the stability box $\mathcal{SB}(\pi_t, T)$ using algorithm STABOX derived in [12]. STOP.

Steps 1 and 2 of algorithm MAX-STABOX are based on Property 3 and Remark 1. Step 4 is based on Property 2. Steps 5 – 7 are based on Property 4, part *(ii)*. Step 9 is based on Property 6 which follows.

To prove Property 6, we have to analyze algorithm MAX-STABOX. In steps 1, 2 and 4, all jobs $\mathcal{J}^t = \{J_i \mid J_{u_j} = J_i = J_{l_j}\}$ having the same position in both lists $\mathcal{M}(U)$ and $\mathcal{M}(L)$ obtain fixed positions in the permutation $\pi_t \in S^{max}$.

The positions of the remaining jobs $\mathcal{J} \setminus \mathcal{J}^t$ in the permutation π_t are determined in steps 5 – 7. The fixed order of the jobs $\mathcal{J}^t$ may shorten the original segment $[p_i^L, p_i^U]$ of a job $J_i \in \mathcal{J} \setminus \mathcal{J}^t$. We denote such a reduced segment as $[\widehat{p}_i^L, \widehat{p}_i^U]$. So, in steps 5 – 7, the reduced segment $[\widehat{p}_i^L, \widehat{p}_i^U]$ has to be considered instead of original segment $[p_i^L, p_i^U]$ for a job $J_i \in \mathcal{J} \setminus \mathcal{J}^t$.

Let $\mathcal{L}'$ denote the maximal subset of set $\mathcal{L}$ (see Property 5) including exactly one element from each set $\mathcal{L}(i)$, for which job $J_i \in \mathcal{J}$ satisfies the strict inequalities (13).

Property 6. There exists a permutation $\pi_t \in S$ with the set $\mathcal{L}' \subseteq \mathcal{L}$ of maximal segments $[l_i, u_i]$ of possible variations of the processing time $p_i, J_i \in \mathcal{J}$, preserving the optimality of the permutation π_t.

Proof. Due to Property 2 and steps 1 – 4 of algorithm MAX-STABOX, the maximal segments $[l_i, u_i]$ and $[l_v, u_v]$ (if any) of jobs J_i and J_v satisfying (11) preserve the optimality of the permutation $\pi_t \in S^{max}$.

Let $\mathcal{J}^*$ denote the set of all jobs J_i satisfying (13). It is easy to see that

$$\bigcap_{J_i \in \mathcal{J}} (\widehat{p}_i^L, \widehat{p}_i^U] = \varnothing.$$

Therefore,

$$\bigcap_{J_i \in \mathcal{J}} \mathcal{J}(i) = \varnothing.$$

Hence, step 9 is correct: putting the set of jobs $\mathcal{J}$ in the positions defined in steps 3 – 8 does not cause any contradiction of the job orders. ■

Obviously, steps 1 and 2 take $O(n \log n)$ time. Due to Properties 4 and 5, steps 6, 7 and 9 take $O(n)$ time. Step 10 takes $O(n \log n)$ time since algorithm STABOX derived in [12] has the same complexity. Thus, the whole algorithm MAX-STABOX takes $O(n \log n)$ time. It is easy to convince that, due to steps 1 – 5, the permutation π_t constructed by algorithm MAX-STABOX possesses property (a) and, due to steps 6, 7 and 9, this permutation possesses property (b).

Remark 2. Algorithm MAX-STABOX constructs a permutation $\pi_t \in S$ such that the dimension $|N_t|$ of the stability box $\mathcal{SB}(\pi_t, T) = \times_{t_i \in N_t}[l_{t_i}, u_{t_i}] \subseteq T$ is the largest one for all permutations S, and the volume of the stability box $\mathcal{SB}(\pi_t, T)$ is the largest one for all permutations $\pi_k \in S$ having the largest dimension $|N_k| = |N_t|$ of their stability boxes $\mathcal{SB}(\pi_k, T)$.

Returning to Example 1, one can show (using Algorithm MAX-STABOX) that permutation $\pi_1 = (J_1, J_2, \ldots, J_8)$ has the largest dimension and volume of a stability box. Next, we compare $\mathcal{SB}(\pi_1, T)$ with the stability boxes calculated for the permutations obtained by the three heuristics defined as follows.

The lower-point heuristic generates an optimal permutation $\pi_l \in S$ for the instance $1|p^L|\sum w_i C_i$ with

$$p^L = (p_1^L, p_2^L, \ldots, p_n^L) \in T. \tag{14}$$

The upper-point heuristic generates an optimal permutation $\pi_u \in S$ for the instance $1|p^U|\sum w_i C_i$ with

$$p^U = (p_1^U, p_2^U, \ldots, p_n^U) \in T. \tag{15}$$

The mid-point heuristic generates an optimal permutation $\pi_m \in S$ for the instance $1|p^M|\sum w_i C_i$ with

$$p^M = \left(\frac{p_1^U - p_1^L}{2}, \frac{p_2^U - p_2^L}{2}, \ldots, \frac{p_n^U - p_n^L}{2}\right) \in T. \tag{16}$$

We obtain the permutation $\pi_l = (J_2, J_1, J_4, J_3, J_6, J_5, J_8, J_7)$ with the stability box

$$\mathcal{SB}(\pi_l, T) = \left[\frac{w_2}{d_2^+}, \frac{w_2}{d_2^-}\right] \times \left[\frac{w_6}{d_6^+}, \frac{w_6}{d_6^-}\right] = [1, 2] \times [10, 11].$$

The volume of the stability box $\mathcal{SB}(\pi_l, T)$ is equal to 1. We obtain the permutation $\pi_u = (J_1, J_3, J_2, J_4, J_5, J_7, J_6, J_8)$ and the permutation $\pi_m = (J_1, J_2, J_4, J_3, J_5, J_6, J_8, J_7)$. The volume of the stability box

$$\mathcal{SB}(\pi_u, T) = \left[\frac{w_4}{d_4^+}, \frac{w_4}{d_4^-}\right] \times \left[\frac{w_8}{d_8^+}, \frac{w_8}{d_8^-}\right] = [9, 10] \times [19, 20]$$

is equal to 1. The volume of the stability box

$$\mathcal{SB}(\pi_m, T) = \left[\frac{w_2}{d_2^+}, \frac{w_2}{d_2^-}\right] \times \left[\frac{w_6}{d_6^+}, \frac{w_6}{d_6^-}\right] = [3, 6] \times [12, 15]$$

is equal to $9 = 3 \cdot 3$. It is the same volume of the stability box as that of permutation π_1. Note, however, that the dimension $|N_m|$ of the stability box $\mathcal{SB}(\pi_m, T)$ is equal to 2, while the dimension $|N_1|$ of the stability box $\mathcal{SB}(\pi_1, T)$ of the permutation $\pi_1 \in S^{max}$ is equal to 4. Thus, $\pi_m \notin S^{max}$ since permutation π_m does not possess property (a).

4 Computational Results

There might be several permutations with the largest dimension and relative volume of a stability box $\mathcal{SB}(\pi_t, T)$ since several consecutive jobs in a permutation $\pi_t \in S^{max}$ may have an empty range of possible variations of their processing

times preserving the optimality of the permutation π_t. In the computational experiments, we break ties in ordering such jobs by adopting the mid-point heuristic which generates a subsequence of these jobs as a part of an optimal permutation $\pi_m \in S$ for the instance $1|p^M|\sum w_iC_i$ with the scenario $p^M \in T$ defined by (16).

Our choice of the mid-point heuristic is based on the computational results of the experiments conducted in [12] for the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ with $10 \le n \le 1000$. In those computational results, the subsequence of a permutation $\pi_m \in S$ outperformed both the corresponding subsequence of the permutation $\pi_l \in S$ and that of the permutation $\pi_u \in S$ defined by (14) and (15), respectively.

We coded the algorithm MAX-STABOX combined with the mid-point heuristic for ordering consecutive jobs having an empty range of their processing times preserving the optimality of the permutation $\pi_t \in S^{max}$ in C++. This algorithm was tested on a PC with AMD Athlon (tm) 64 Processor 3200+, 2.00 GHz, 1.96 GB of RAM. We solved (exactly or approximately) a lot of randomly generated instances. Some of the computational results obtained are presented in Tables 2 – 4 for randomly generated instances of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ with the number $n \in \{1000, 1100, \ldots, 2000\}$ of jobs.

Each series presented in Tables 2 – 4 contains 100 solved instances with the same combination of the number n of jobs and the same maximal possible error $\delta\%$ of the random processing times $p_i \in [p_i^L, p_i^U]$. The integer center C of a segment $[p_i^L, p_i^U]$ was generated using the uniform distribution in the range $[L, U]$: $L \le C \le U$. The lower bound p_i^L for the possible job processing time $p_i \in [p_i^L, p_i^U]$ was defined as $p_i^L = C \cdot (1 - \frac{\delta}{100})$, the upper bound p_i^U of $p_i \in [p_i^L, p_i^U]$ was defined as $p_i^U = C \cdot (1 + \frac{\delta}{100})$. The same range $[L, U]$ for the varying center C of the segment $[p_i^L, p_i^U]$ was used for all jobs $J_i \in \mathcal{J}$, namely: $L = 1$ and $U = 100$. In Tables 2 – 4, we report computational results for the series of instances of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ with the maximal possible errors $\delta\%$ of the job processing times from the set $\{0.25\%, 0.4\%, 0.5\%, 0.75\%, 1\%, 2.5\%, 5\%, 15\%, 25\%\}$.

For each job $J_i \in \mathcal{J}$, the weight $w_i \in R_+^1$ was uniformly distributed in the range $[1, 50]$. It should be noted that the job weights w_i were assumed to be known before scheduling (in contrast to the actual processing times p_i^* of the jobs $J_i \in \mathcal{J}$, which were assumed to be unknown before scheduling).

The number n of jobs in each instance of a series is given in column 1 of Table 2, Table 3 and Table 4. The maximum possible error $\delta\%$ of the random processing times $p_i \in [p_i^L, p_i^U]$ is given in column 2. Column 3 represents the average relative number $|\mathcal{A}|$ of the arcs in the dominance digraph $(\mathcal{J}, \mathcal{A})$ constructed using condition (3) of Theorem 1. The relative number $|\mathcal{A}|$ is calculated in percentages of the number of arcs in the complete circuit-free digraph of order n as follows: $\left(|\mathcal{A}| : \frac{n(n-1)}{2}\right) \cdot 100\%$. Column 4 represents the average dimension $|N_t|$ of the stability box $\mathcal{SB}(\pi_t, T)$ of the permutation π_t with the largest relative volume of a stability box. $|N_k|$ is equal to the number of jobs with a non-zero maximal possible variation of the processing time preserving the optimality of permutation $\pi_t \in S^{max}$. Column 5 represents the average relative volume of the stability box $\mathcal{SB}(\pi_t, T)$ of the permutations π_t with the largest dimension and

relative volume of a stability box. If $\mathcal{SB}(\pi_t, T) = T$ for all instances in the series, then column 5 contains the number one.

In the experiments, we answered the question of how large the relative error Δ of the objective function $\gamma = \sum_{i=1}^{n} w_i C_i$ was for the permutation $\pi_t \in S^{max}$ with the largest dimension and relative volume of a stability box $\mathcal{SB}(\pi_t, T)$:

$$\Delta = \frac{\gamma_{p^*}^t - \gamma_{p^*}}{\gamma_{p^*}},$$

where p^* is the actual scenario (unknown before scheduling), γ_{p^*} is the optimal objective function value for the scenario $p^* \in T$ and $\gamma_{p^*}^t = \sum_{i=1}^{n} w_i C_i(\pi_t, p^*)$.

Column 6 represents the number of instances (among the 100 instances in a series) for which a permutation π_t with the largest dimension and relative volume of the stability box $\mathcal{SB}(\pi_t, T)$ provides an optimal solution for the instance $1|p^*|\sum w_i C_i$ with the actual processing times $p^* = (p_1^*, p_2^*, \ldots, p_n^*) \in T$.

¿From the experiments, it follows that, if the maximal possible error of the processing times is not greater than 0.4%, then the dominance digraph $(\mathcal{J}, \mathcal{A})$ is a complete circuit-free digraph. Therefore, the permutation $\pi_t \in S^{max}$ provides an optimal solution for such an instance $1|p^*|\sum w_i C_i$.

The average (maximum) relative error Δ of the objective function value $\gamma_{p^*}^t$ calculated for the permutation $\pi_t \in S^{max}$ constructed by the algorithm MAX-STABOX with respect to the optimal objective function value γ_{p^*} defined for the actual job processing times is given in column 7 (in column 8, respectively).

For all series presented in Tables 2 – 4, the average (maximum) error Δ of the value $\gamma_{p^*}^t$ of the objective function $\gamma = \sum_{i=1}^{n} w_i C_i$ obtained for the permutation $\pi_t \in S^{max}$ with the largest dimension and relative volume of a stability box was not greater than 0.012069 (not greater than 0.013812).

The CPU-time for an instance of a series is presented in column 5. This time includes the time for the realization of the $O(n^2)$ algorithm for constructing the dominance digraph $(\mathcal{J}, \mathcal{A})$ using condition (3) of Theorem 1 and the time for the realization of the $O(n \log n)$ algorithm MAX-STABOX for constructing the permutation $\pi_t \in S^{max}$ and the stability box $\mathcal{SB}(\pi_t, T)$. This CPU-time grows rather slowly with n, and it was not greater than 79.06 s for each instance.

5 Conclusion

In [12], an $O(n^2)$ algorithm has been developed for calculating a permutation $\pi_t \in S$ with the largest dimension and volume of a stability box $\mathcal{SB}(\pi_t, T)$. In Section 3, we proved Properties 1 – 6 of a stability box allowing us to derive an $O(n \log n)$ algorithm for calculating such a permutation $\pi_t \in S^{max}$. The dimension and volume of a stability box are efficient invariants of the uncertain data T, as it was shown in simulation experiments on a PC reported in Section 4.

The results that we presented may be generalized to the problem $1|prec, p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$, where the precedence constraints are given a priori on the set

Table 2. Randomly generated instances with $[L, U] = [1, 100]$, $w_i \in [1, 50]$ and $n \in \{1000, 1100, 1200, 1300\}$.

Number of jobs n	Maximal error of p_i $\delta\%$	Relative arc number $\lvert\mathcal{A}\rvert$ (in %)	Average dimension $\lvert N_t\rvert$	Relative volume of $\mathcal{SB}(\pi_t, T)$	Number of exact solutions	Average error Δ	Maximal error Δ	CPU time (in s)
1	2	3	4	5	6	7	8	9
1000	0.25%	100	1000	1	100	0	0	8.62
1000	0.4%	100	1000	1	100	0	0	8.56
1000	0.5%	100	989.61	0.227427	11	≈ 0	≈ 0	8.69
1000	0.75%	99.545177	451.29	≈ 0	0	0.000023	0.000031	8.98
1000	1%	99.192559	330.65	≈ 0	0	0.000042	0.000051	8.96
1000	2.5%	97.591726	124	0.000001	0	0.000157	0.000181	8.9
1000	5%	94.889794	54.86	0.001976	0	0.000526	0.000614	8.84
1000	15%	84.39185	12.29	0.011288	0	0.004309	0.004858	8.86
1000	25%	73.954372	4.71	0.09081	0	0.012045	0.013303	8.89
1100	0.25%	100	1100	1	100	0	0	11.51
1100	0.4%	100	1100	1	100	0	0	11.46
1100	0.5%	99.997839	1087.27	0.200252	11	≈ 0	≈ 0	11.51
1100	0.75%	99.539967	478.35	≈ 0	0	0.000023	0.00003	12.1
1100	1%	99.188722	349.3	≈ 0	0	0.000043	0.000049	12.05
1100	2.5%	97.611324	131.01	0.000001	0	0.000155	0.000175	11.8
1100	5%	94.862642	57.35	0.006242	0	0.000528	0.000593	11.79
1100	15%	84.288381	11.46	0.017924	0	0.004371	0.004899	11.76
1100	25%	74.076585	4.29	0.133804	0	0.01189	0.013289	11.8
1200	0.25%	100	1200	1	100	0	0	15.4
1200	0.4%	100	1200	1	100	0	0	15.12
1200	0.5%	99.998	1185.27	0.174959	5	≈ 0	0.000001	15.42
1200	0.75%	99.540619	515.8	≈ 0	0	0.000023	0.000029	16
1200	1%	99.190977	375.34	≈ 0	0	0.000042	0.000051	16.06
1200	2.5%	97.581479	138.75	0.000002	0	0.000156	0.000177	15.81
1200	5%	94.88253	62.06	0.006396	0	0.000534	0.000596	15.51
1200	15%	84.376763	12.88	0.042597	0	0.004332	0.004733	15.33
1200	25%	74.100395	5.01	0.08078	0	0.011872	0.01351	15.21
1300	0.25%	100	1300	1	100	0	0	19.75
1300	0.4%	100	1300	1	100	0	0	19.38
1300	0.5%	99.997583	1280.26	0.084004	2	≈ 0	≈ 0	19.54
1300	0.75%	99.549162	543.2	≈ 0	0	0.000023	0.000026	20.3
1300	1%	99.199789	400.41	≈ 0	0	0.000042	0.000053	20.32
1300	2.5%	97.602491	148.41	0.000004	0	0.000157	0.000186	20.01
1300	5%	94.877326	65.23	0.019927	0	0.000532	0.000588	19.95
1300	15%	84.388473	13.47	0.024207	0	0.004364	0.004758	19.52
1300	25%	73.975873	5.5	0.08254	0	0.011962	0.013812	19.52

Table 3. Randomly generated instances with $[L, U] = [1, 100]$, $w_i \in [1, 50]$ and $n \in \{1400, 1500, 1600, 1700\}$.

Number of jobs n	Maximal error of p_i $\delta\%$	Relative arc number $\lvert\mathcal{A}\rvert$ (in %)	Average dimension $\lvert N_t\rvert$	Relative volume of $\mathcal{SB}(\pi_t, T)$	Number of exact solutions	Average error Δ	Maximal error Δ	CPU time (in s)
1	2	3	4	5	6	7	8	9
1400	0.25%	100	1400	1	100	0	0	24.92
1400	0.4%	100	1400	1	100	0	0	24.8
1400	0.5%	99.997556	1377.21	0.078809	1	≈ 0	0.000001	24.97
1400	0.75%	99.539142	575.2	≈ 0	0	0.000023	0.000029	25.67
1400	1%	99.198461	422.65	≈ 0	0	0.000042	0.00005	25.63
1400	2.5%	97.594897	154.9	0.000001	0	0.000157	0.000178	25.1
1400	5%	94.869044	70.36	0.002356	0	0.000533	0.000615	25.29
1400	15%	84.364242	14.35	0.029338	0	0.004339	0.004841	24.72
1400	25%	74.096446	5.18	0.14077	0	0.011998	0.013041	24.27
1500	0.25%	100	1500	1	100	0	0	31.44
1500	0.4%	100	1500	1	100	0	0	31.08
1500	0.5%	99.997493	1474.09	0.070241	0	≈ 0	0.000001	31.64
1500	0.75%	99.544441	607.5	≈ 0	0	0.000042	0.000052	32.39
1500	1%	99.193199	444.29	≈ 0	0	0.000042	0.000052	32.39
1500	2.5%	97.61593	167.25	0.000005	0	0.000155	0.000171	31.43
1500	5%	94.861654	71.34	0.00282	0	0.000533	0.000582	31.36
1500	15%	84.409904	14.93	0.05372	0	0.004394	0.00492	30.46
1500	25%	74.281235	5.46	0.148403	0	0.011936	0.013685	30.33
1600	0.25%	100	1600	1	100	0	0	38.63
1600	0.4%	100	1600	1	100	0	0	38.67
1600	0.5%	99.997452	1569.35	0.046151	0	≈ 0	0.000001	38.8
1600	0.75%	99.54273	638.18	≈ 0	0	0.000023	0.00003	39.76
1600	1%	99.192323	464.89	≈ 0	0	0.000042	0.000048	40.04
1600	2.5%	97.601128	174.91	0.000004	0	0.000157	0.000177	38.71
1600	5%	94.861356	76.990000	0.003505	0	0.000532	0.000581	38.46
1600	15%	84.343239	14.75	0.036278	0	0.004341	0.004811	37.34
1600	25%	74.123830	5.75	0.087651	0	0.011899	0.013192	36.34
1700	0.25%	100	1700	1	100	0	0	47.29
1700	0.4%	100	1700	1	100	0	0	47.18
1700	0.5%	99.997432	1665.41	0.034556	1	≈ 0	0.000001	47.12
1700	0.75%	99.544993	671.09	≈ 0	0	0.000023	0.000027	48.25
1700	1%	99.203930	495.13	≈ 0	0	0.000041	0.000049	48.47
1700	2.5%	97.598734	180.99	0.000072	0	0.000156	0.000172	46.88
1700	5%	94.852439	80.53	0.001601	0	0.000533	0.000585	46.33
1700	15%	84.358524	17.27	0.028854	0	0.004379	0.0049	45.26
1700	25%	74.030579	6.03	0.082325	0	0.012069	0.013255	44.24

Table 4. Randomly generated instances with $[L, U] = [1, 100]$, $w_i \in [1, 50]$ and $n \in \{1800, 1900, 2000\}$.

Number of jobs n	Maximal error of p_i δ%	Relative arc number $\lvert\mathcal{A}\rvert$ (in %)	Average dimension $\lvert N_t\rvert$	Relative volume of $\mathcal{SB}(\pi_t, T)$	Number of exact solutions	Average error Δ	Maximal error Δ	CPU time (in s)
1	2	3	4	5	6	7	8	9
1800	0.25%	100	1800	1	100	0	0	56.18
1800	0.4%	100	1800	1	100	0	0	56.27
1800	0.5%	99.99761	1764.02	0.02624	0	≈ 0	0.000001	56.72
1800	0.75%	99.547537	706.21	≈ 0	0	0.000023	0.000028	57.38
1800	1%	99.193797	517.06	≈ 0	0	0.000042	0.000049	57.33
1800	2.5%	97.600247	190.97	0.000042	0	0.000156	0.000177	55.81
1800	5%	94.899074	84.82	0.007274	0	0.000529	0.000602	55.27
1800	15%	84.408342	17.67	0.040758	0	0.004348	0.004723	53.42
1800	25%	74.162869	6.38	0.126377	0	0.011981	0.013095	51.86
1900	0.25%	100	1900	1	100	0	0	65.65
1900	0.4%	100	1900	1	100	0	0	66.81
1900	0.5%	99.997533	1858.51	0.018832	0	≈ 0	0.000001	66.69
1900	0.75%	99.54191	733.81	≈ 0	0	0.000023	0.000028	67.75
1900	1%	99.189512	534.79	≈ 0	0	0.000042	0.000049	68.58
1900	2.5%	97.596318	199.82	0.000022	0	0.000156	0.000173	66.36
1900	5%	94.856400	89.93	0.002011	0	0.000534	0.000596	65.68
1900	15%	84.331351	17.61	0.048813	0	0.004372	0.004844	62.97
1900	25%	74.188836	6.82	0.092068	0	0.011965	0.013234	60.74
2000	0.25%	100	2000	1	100	0	0	78.41
2000	0.4%	100	2000	1	100	0	0	78.93
2000	0.5%	99.997489	1953.88	0.017798	2	≈ 0	≈ 0	79.06
2000	0.75%	99.542435	764.35	≈ 0	0	0.000023	0.000027	78.83
2000	1%	99.197383	565.09	≈ 0	0	0.000042	0.000048	78.1
2000	2.5%	97.605895	210.17	0.000035	0	0.000156	0.000173	75.8
2000	5%	94.867102	93.63	0.014015	0	0.000535	0.000606	75.02
2000	15%	84.412199	17.95	0.040101	0	0.004339	0.004751	74.08
2000	25%	73.977021	6.64	0.147426	0	0.01203	0.013046	71.22

of jobs. If the deterministic problem $1|prec|\sum w_iC_i$ for a particular type of precedence constraints is polynomially solvable, then the above results may be used for the uncertain counterpart $1|prec, p_i^L \le p_i \le p_i^U|\sum w_iC_i$. In the latter problem, the dominance digraph $(\mathcal{J}, \mathcal{A})$ contains the arc (J_u, J_v) only if this arc does not violate the precedence constraint given between the jobs J_u and J_v a priori.

Acknowledgements The first and second authors were supported in this research by National Science Council of Taiwan. The authors are grateful to Natalja G. Egorova for coding algorithm MAX-STABOX and other contributions.

References

1. Daniels, R., Kouvelis, P.: Robust scheduling to hedge against processing time uncertainty in single stage production. Management Science 41(2), 363 – 376 (1995)
2. Kasperski, A.: Minimizing maximal regret in the single machine sequencing problem with maximum lateness criterion. Operations Research Letters 33, 431 – 436 (2005)
3. Kasperski, A., Zelinski, P.: A 2-approximation algorithm for interval data minmax regret sequencing problem with total flow time criterion. Operations Research Letters 36, 343 – 344 (2008)
4. Lai, T.-C., Sotskov, Y.: Sequencing with uncertain numerical data for makespan minimization. Journal of the Operations Research Society 50, 230 – 243 (1999)
5. Lai, T.-C., Sotskov, Y., Sotskova, N., Werner, F.: Optimal makespan scheduling with given bounds of processing times. Mathematical and Computer Modelling 26(3), 67 – 86 (1997)
6. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems. Prentice-Hall, Englewood Cliffs, NJ, USA (2002)
7. Sabuncuoglu, I., Goren, S.: Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research. International Journal of Computer Integrated Manufacturing 22(2), 138 – 157 (2009)
8. Slowinski, R., Hapke, M.: Scheduling under Fuzziness. Physica-Verlag, Heidelberg, Germany, New York, USA (1999)
9. Smith, W.: Various optimizers for single-stage production. Naval Research Logistics Quarterly 3(1), 59 – 66 (1956)
10. Sotskov, Y., Egorova, N., Lai, T.-C.: Minimizing total weighted flow time of a set of jobs with interval processing times. Mathematical and Computer Modelling 50, 556 – 573 (2009)
11. Sotskov, Y., Egorova, N., Werner, F.: Minimizing total weighted completion time with uncertain data: A stability approach. Automation and Remote Control 71 (10), 2038–2057 (2010)
12. Sotskov, Y., Lai, T.-C.: Minimizing total weighted flow time under uncertainty using dominance and a stability box. Computers & Operations Research 39, 1271 – 1289 (2012)
13. Sotskov, Y., Sotskova, N., Lai, T.-C., Werner, F.: Scheduling under Uncertainty. Theory and Algorithms. Belorusskaya nauka, Minsk, Belarus (2010)
14. Sotskov, Y., Wagelmans, A., Werner, F.: On the calculation of the stability radius of an optimal or an approximate schedule. Annals of Operations Research 83, 213 – 252 (1998)