The Stability Box in Interval Data for Minimizing the Sum of Weighted Completion Times.

Conference Paper · January 2011

Source: DBLP

4 authors, including:

Yuri N. Sotskov
United Institute Of Informatics Problems

Tsung-Chyan Lai
Harbin Engineering University

Frank Werner
Otto-von-Guericke-Universität Magdeburg

THE STABILITY BOX IN INTERVAL DATA FOR MINIMIZING THE SUM OF WEIGHTED COMPLETION TIMES

Yuri N. Sotskov
United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova Str 6, Minsk, Belarus

Natalja G. Egorova
United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova Str 6, Minsk, Belarus

Tsung-Chyan Lai
Department of Business Administration, National Taiwan University, Roosevelt Rd 85, Taipei, Taiwan

Frank Werner
Faculty of Mathematics, Otto-von-Guericke-University, Magdeburg, Germany

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

Abstract: We consider a single-machine scheduling problem, in which the processing time of a job can take any value from a given segment. The criterion is to minimize the sum of weighted completion times of the n jobs, a positive weight being associated with a 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 volume and dimension of a stability box. The efficiency of a permutation with the largest dimension and volume of a stability box is demonstrated via a simulation on a set of randomly generated instances with $1000 \leq n \leq 2000$. If several permutations have the largest volume of a stability box, the developed algorithm selects one of them due to one of three simple heuristics: a lower-point heuristic, an upper-point heuristic or a mid-point heuristic.

April 8, 2011

1 INTRODUCTION

In a real-life scheduling problem, the numerical data are often uncertain. The stochastic method (Pinedo, 2002) or the fuzzy method (Slowinski and Hapke, 1999) are used when the job processing times may be defined as random variables or as fuzzy numbers. If the processing times can be defined neither as random variables with known probability distributions nor as fuzzy numbers with known membership functions, other methods are needed to solve a scheduling problem under uncertainty (Daniels and Kouvelis, 1995; Sabuncuoglu and Goren, 2009; Sotskov et al., 2010). In particular, the robust method (Daniels and Kouvelis, 1995) assumes that the decision-maker prefers a schedule hedging against the worst-case scenario among possible realizations of the job processing times. The stability method (Lai and Sotskov, 1999; Lai et al., 1997; Sotskov et al., 2010) combines a stability analysis with a multi-stage scheduling decision framework on the basis of the data obtained while some jobs have been completed.

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 volume of a stability box. An example is considered in Subsection 3.1. Computational results are presented in Section 4. We conclude with Section 5.

2 PROBLEM SETTING

A set of jobs $\mathcal{J} = \{J_1, ..., J_n\}$, $n \geq 2$, has to be processed on a single machine, a positive weight w_i being given for a 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 $[p_i^L, p_i^U]$, $0 \leq p_i^L \leq p_i^U$. The exact value $p_i \in [p_i^L, p_i^U]$ may remain unknown until the completion of job J_i.

Let $T = \{p \in R_+^n \mid p_i^L \leq p_i \leq p_i^U,\ i \in \{1, \ldots, n\}\}$ denote the set of vectors $p = (p_1, \ldots, p_n)$ (scenarios) of the possible job processing times. $S = \{\pi_1, \ldots, \pi_{n!}\}$ denotes the set of permutations $\pi_k = (J_{k_1}, \ldots, J_{k_n})$ 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 $\pi_t \in S$:

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

Hereafter, $C_i(\pi_k, p) = C_i$ is the completion time of job $J_i \in \mathcal{J}$ in a semi-active schedule defined by 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 only after the execution of the schedule. 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 $\pi_t \in S$ before the realization of the schedule. However, one must somehow define a schedule before to realize it. So, problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ of finding an optimal permutation $\pi_k \in S$ defined in (1) is not correct. In general, one can find only a heuristic solution (a 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 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. Problem $1||\sum w_i C_i$ is correct and can be solved exactly in $O(n \log n)$ time (Smith, 1956) due to the necessary and sufficient condition (2) for the optimality of a permutation $\pi_k = (J_{k_1}, \ldots, J_{k_n}) \in S$:

$$\frac{w_{k_1}}{p_{k_1}} \geq \ldots \geq \frac{w_{k_n}}{p_{k_n}}, \tag{2}$$

where $p_{k_i} > 0$ 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 (Sotskov and Lai, 2011), the stability box $\mathcal{SB}(\pi_k, T)$ within a set of scenarios T has been defined for a permutation $\pi_k = (J_{k_1}, \dots, J_{k_n}) \in S$. To present the definition of $\mathcal{SB}(\pi_k, T)$, we need the following notations. Let $\mathcal{J}(k_i) = \{J_{k_1}, \dots, J_{k_{i-1}}\}$, $\mathcal{J}[k_i] = \{J_{k_{i+1}}, \dots, J_{k_n}\}$. S_{k_i} is the set of permutations $(\pi(\mathcal{J}(k_i)), J_{k_i}, \pi(\mathcal{J}[k_i])) \in S$, $\pi(\mathcal{J}')$ denoting a permutation of the jobs $\mathcal{J}' \subset \mathcal{J}$. N_k is a subset of $N = \{1, \dots, n\}$. The notation $1|p|\sum w_iC_i$ is used for indicating an instance with a fixed scenario $p \in T$ of the deterministic problem $1||\sum w_iC_i$.

Definition 1 (Sotskov and Lai, 2011) *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 $\pi_k = (J_{k_1}, \dots, J_{k_n}) \in S$, if permutation $\pi_e = (J_{e_1}, \dots, J_{e_n}) \in S_{k_i}$ being optimal for the instance $1|p|\sum w_iC_i$ with a scenario $p = (p_1, \dots, p_n) \in T$ remains optimal for the instance $1|p'|\sum w_iC_i$ with a scenario

$$p' \in \{\times_{j=1, j\neq i}^{n} [p_{k_j}, p_{k_j}]\} \times [l_{k_i}, u_{k_i}]$$

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 $1|p|\sum w_iC_i$, then $\mathcal{SB}(\pi_k, T) = \emptyset$.

For any scheduling instance, the stability box is a subset of the stability region (Sotskov et al., 1998). However, we substitute the stability region by the stability box, since the latter is easy to compute.

3.1 Illustrative example

For the sake of simplicity of the calculation, we consider the special case $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$ of problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$ 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 problem $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$ 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 (Sotskov and Lai, 2011), formula (8) 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 permutation $\pi_k = (J_{k_1}, \dots, J_{k_n}) \in S$ is calculated as follows:

$$d^-_{k_i} = \max\left\{\frac{w_{k_i}}{p^U_{k_i}},\ \max_{i<j\le n}\left\{\frac{w_{k_j}}{p^L_{k_j}}\right\}\right\}, i \in \{1,\dots,n-1\}, \tag{3}$$

$$d^-_{k_n} = \frac{w_{k_n}}{p^U_{k_n}}. \tag{4}$$

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 π_k is calculated as

$$d^+_{k_i} = \min\left\{\frac{w_{k_i}}{p^L_{k_i}},\ \min_{1\le j<i}\left\{\frac{w_{k_j}}{p^U_{k_j}}\right\}\right\}, i \in \{2,\dots,n\}, \tag{5}$$

$$d^+_{k_1} = \frac{w_{k_1}}{p^L_{k_1}}. \tag{6}$$

For Example 1, the values $d^-_{k_i}, i \in \{1,\dots,8\}$, defined in (3) and (4) are given in column 5 of Table 1. The values $d^+_{k_i}$ defined in (5) and (6) are given in column 6.

Theorem 1 (Sotskov and Lai, 2011) *If there is no job J_{k_i}, $i \in \{1,\dots,n-1\}$, in permutation $\pi_k = (J_{k_1},\dots,J_{k_n}) \in S$ such that inequality*

$$\frac{w_{k_i}}{p^L_{k_i}} < \frac{w_{k_j}}{p^U_{k_j}} \tag{7}$$

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

$$\mathcal{SB}(\pi_k, T) = \times_{d^-_{k_i} \le d^+_{k_i}} \left[\frac{w_{k_i}}{d^+_{k_i}}, \frac{w_{k_i}}{d^-_{k_i}}\right]. \tag{8}$$

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

Using Theorem 1, we can calculate the stability box $\mathcal{SB}(\pi_1, T)$ of permutation $\pi_1 = (J_1,\dots,J_8)$ in Example 1. We convince that there is no job J_{k_i}, $i \in \{1,\dots,n-1\}$, with inequality (7). Due to Theorem 1, $\mathcal{SB}(\pi_1, T) \ne \emptyset$. 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 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 $\mathcal{J}$.

Using formula (8), we obtain the stability box for permutation π_1:

$$\mathcal{SB}(\pi_1, T) = \left[\frac{w_2}{d^+_2}, \frac{w_2}{d^-_2}\right] \times \left[\frac{w_4}{d^+_4}, \frac{w_4}{d^-_4}\right] \times \left[\frac{w_6}{d^+_6}, \frac{w_6}{d^-_6}\right] \times \left[\frac{w_8}{d^+_8}, \frac{w_8}{d^-_8}\right] = [3,6] \times [9,10] \times [12,15] \times [19,20].$$

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 $9 = 3 \cdot 1 \cdot 3 \cdot 1$.

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

$$\left(\frac{w_i}{d^-_i} - \frac{w_i}{d^+_i}\right) : \left(p^U_i - p^L_i\right) \tag{9}$$

for the jobs $J_i \in \mathcal{J}$ having non-empty ranges $[l_i, u_i]$ of possible variations of the time p_i (inequality $d^-_i \le d^+_i$ 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.

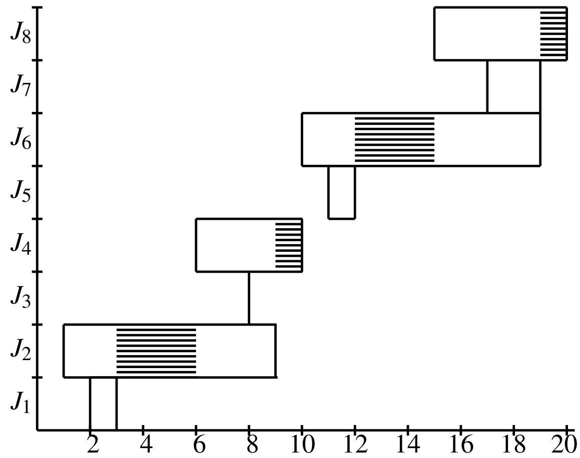

Figure 1: The maximal ranges $[l_i, u_i]$ of possible variations of the processing times $p_i, i \in \{2,4,6,8\}$, within the stability box $\mathcal{SB}(\pi_1, T)$ are dashed.

3.2 Properties of a stability box

We investigate some properties of a stability box, which allow us to derive an $O(n \log n)$ algorithm for finding a permutation $\pi_t \in S$ with the largest volume of a stability box

$$\mathcal{SB}(\pi_t, T) = \times_{t_i \in N_t} [l_{t_i}, u_{t_i}] \subseteq T.$$

Definition 1 implies the following claim.

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

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

Let S^{max} be the set of all permutations in S with the largest volume and dimension of a stability box. 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 $\pi_t = (J_{t_1}, \ldots, J_{t_n}) \in S^{max}$. 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 $\left[\frac{w_v}{p_v^U}, \frac{w_v}{p_v^L}\right]$.

Case (I) is defined by the inequalities

$$\frac{w_v}{p_v^U} \leq \frac{w_i}{p_i^U}, \frac{w_v}{p_v^L} \leq \frac{w_i}{p_i^L} \tag{10}$$

provided that at least one of inequalities (10) is strict.

In case (I), the order of the jobs J_v and J_i in permutation $\pi_t \in S^{max}$ may be defined by a strict inequality from (10): 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 (3) – (6) and (8)). The following property has been 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

$$\frac{w_v}{p_v^U} = \frac{w_i}{p_i^U}, \frac{w_v}{p_v^L} = \frac{w_i}{p_i^L}. \tag{11}$$

Property 3 *For case (II), there exists a permutation $\pi_t \in S^{max}$, in which 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 jobs J_i and J_v are located adjacently, then the maximal range $[l_u, u_u]$ of possible variation of the processing time p_u for any job $J_u \in \mathcal{J} \setminus \{J_i, J_v\}$ preserving the optimality of 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 jobs J_i and J_v. ■

If equalities (11) hold, one can restrict the search for a permutation $\pi_t \in S^{max}$ by a subset of permutations in S with 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 (11) as one job (either job J_i or J_v).*

Case (III) is defined by the strict inequalities

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

For 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 (12) holds.

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 a particular permutation from 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 fixed $\pi_k \in S$ and $J_i \in \mathcal{J}$.

Part *(ii)*. 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 $\left[\frac{w_v}{p_v^U}, \frac{w_v}{p_v^L}\right]$, $J_v \in \mathcal{J}$, 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 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 (10) (case (I)) or equalities (11) (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 (12) may generate a new possible maximal segment $[l_i, u_i]$ just for job J_i satisfying the same inequalities (12) 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 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

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

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 lists $\mathcal{M}(U) = (J_{u_1}, \dots, J_{u_n})$ and $\mathcal{W}(U) = \left(\frac{w_{u_1}}{p_{u_1}^U}, \dots, \frac{w_{u_n}}{p_{u_n}^U}\right)$ in non-decreasing order of $\frac{w_{u_r}}{p_{u_r}^U}$. Ties are broken via increasing $\frac{w_{u_r}}{p_{u_r}^L}$.

Step 2: Construct the lists $\mathcal{M}(L) = (J_{l_1}, \dots, J_{l_n})$ and $\mathcal{W}(L) = \left(\frac{w_{l_1}}{p_{l_1}^L}, \dots, \frac{w_{l_n}}{p_{l_n}^L}\right)$ in non-decreasing order of

$\frac{w_{l_r}}{p^L_{l_r}}$. Ties are broken via increasing $\frac{w_{l_r}}{p^U_{l_r}}$.

Step 3: **for** $j = 1$ to $j = n$ **do** compare J_{u_j} and 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 $\pi_t \in S^{max}$ **go to** step 8.

Step 5: **else** job $J_{u_j} = J_i$ satisfies (12). Construct the set $\mathcal{J}(i) = \{J_{u_{r+1}}, \dots, J_{l_{k+1}}\}$ of all jobs J_v satisfying (12), 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 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 $j = k + 1$ **go to** step 4.

Step 8: Set $j := j + 1$ **go to** step 4.
end for

Step 9: Construct 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** from (Sotskov and Lai, 2011).
Stop.

Steps 1 and 2 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.

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 permutation $\pi_t \in S^{max}$. The positions of the remaining jobs $\mathcal{J} \setminus \mathcal{J}^t$ in permutation π_t are determined in steps 5 – 7. The fixed order of the jobs $\mathcal{J}^t$ may shorten the original segment $[p^L_i, p^U_i]$ of a job $J_i \in \mathcal{J} \setminus \mathcal{J}^t$ as follows: $[\widehat{p}^L_i, \widehat{p}^U_i]$. So, in steps 5 – 7, the reduced segment $[\widehat{p}^L_i, \widehat{p}^U_i]$ has to be considered instead of segment $[p^L_i, p^U_i]$ for a job $J_i \in \mathcal{J} \setminus \mathcal{J}^t$. Let I' denote the maximal subset of set I including exactly one element from each set $I(i)$, for which job $J_i \in \mathcal{J}$ satisfies (12).

Property 6 *There exists a permutation $\pi_t \in S$ with the set $I' \subseteq I$ 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 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 (10) preserve the optimality of permutation $\pi_t \in S^{max}$.

Let $\mathcal{J}^*$ denote the set of all jobs J_i satisfying (12). It is easy to see that

$$\bigcap_{J_i \in \mathcal{J}} (\widehat{p}^L_i, \widehat{p}^U_i] = \varnothing.$$

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. ■

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 (Sotskov and Lai, 2011) has the same complexity. Thus, the whole algorithm MAX-STABOX takes $O(n \log n)$ time.

Using Algorithm MAX-STABOX, one can show that the permutation $\pi_1 = (J_1, \dots, J_8)$ has the largest volume of a stability box in Example 1.

Next, we compare $\mathcal{SB}(\pi_1, T)$ with the stability boxes calculated for the permutations obtained by 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

$$p^L = (p^L_1, \dots, p^L_n) \in T.$$

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

$$p^U = (p^U_1, \dots, p^U_n) \in T.$$

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

$$p^M = \left(\frac{p^U_1 - p^L_1}{2}, \dots, \frac{p^U_n - p^L_n}{2} \right) \in T.$$

For Example 1, we obtain $\pi_l = (J_2, J_1, J_4, J_3, J_6, J_5, J_8, J_7)$ and

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

The volume of the stability box $\mathcal{SB}(\pi_l, T)$ is equal to 1.

We obtain $\pi_u = (J_1, J_3, J_2, J_4, J_5, J_7, J_6, J_8)$ and $\pi_m = (J_1, J_2, J_4, J_3, J_5, J_6, J_8, J_7)$. The volume of the stability box

$$\mathcal{SB}(\pi_u, T) = \left[\frac{w_8}{d_8^+}, \frac{w_8}{d_8^-}\right] = [19,20] \times [10,11]$$

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 that the dimension of the stability box $\mathcal{SB}(\pi_m, T)$ is equal to 2, while the dimension of the stability box $\mathcal{SB}(\pi_1, T)$ is equal to 4.

4 COMPUTATIONAL RESULTS

There might be several permutations with the largest dimension and (or) relative volume of a stability box, e.g., since several consecutive jobs in a permutation $\pi_k \in S^{max}$ may have an empty range of possible variations of their weight-to-process ratios. We break ties in ordering such jobs by adopting one of the three obvious heuristics. Algorithm MAX-STABOX combined with the lower-point heuristic, the upper-point heuristic and the mid-point heuristic (see Subsection 3.3) is called Algorithm SL, Algorithm SU and Algorithm SM, respectively.

Note that in the experiments for $10 \le n \le 1000$ conducted in (Sotskov and Lai, 2011), the mid-point heuristic outperformed the lower-point heuristic and the upper-point heuristic.

Algorithms SL, Algorithm SU and Algorithm SM were coded in C++ and 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 Table 2 for randomly generated instances of problem $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$ with

$$n \in \{1000, 1200\ldots, 2000\}.$$

Each series, which is presented in Table 2, contains 10 instances with the same number n and the same maximal error δ of the random processing times. The number n is given in parenthesis in column 1 in Table 2 before the corresponding set of instances tested.

An 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 was defined as follows:

$$p_i^L = C \cdot (1 - \frac{\delta}{100}),$$

and the upper bound p_i^U was defined as follows:

$$p_i^U = C \cdot (1 + \frac{\delta}{100}).$$

The maximal possible error of the random processing time (in percentages) is equal to $\delta\%$ given in column 1 in Table 2.

We tested instances of problem $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$ with

$$\delta\% \in \{0.25\%, 0.5\%, 0.75\%, 1\%, 2.5\%, 5\%, 15\%, 25\%\}.$$

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 $U = 100$.

For each job $J_i \in \mathcal{J}$, the weight $w_i \in R_+^1$ was uniformly distributed in the range $[1, 50]$.

In the experiments, we answered the question of how large the relative error Δ of the value $\gamma^t_{p^*}$ of the objective function

$$\gamma = \sum_{i=1}^{n} w_i C_i$$

was obtained for the permutation π_t with the largest dimension and relative volume of a stability box $\mathcal{SB}(\pi_t, T)$ with respect to the actually optimal objective function value γ_{p^*} calculated for the actual processing times $p^* = (p_1^*, \dots, p_n^*) \in T$:

$$\Delta = \frac{\gamma_{p^*} - \gamma^t_{p^*}}{\gamma_{p^*}}.$$

In contrast to the weights w_i, the actual processing times $p_i^*, J_i \in \mathcal{J}$, were assumed to be unknown before scheduling. Column 2 represents the average largest relative volume of the stability box $\mathcal{SB}(\pi_t, T)$ (see (9) for the definition of a relative stability box).

The average (maximum) error Δ of the value $\gamma^k_{p^*}$ of the objective function $\gamma = \sum_{i=1}^{n} w_i C_i$ obtained for the permutation π_k with the largest relative volume of a stability box are presented in columns 3 – 5 (columns 6 – 8). Columns 3, 4 and 5 (columns 6, 7 and 8) present the average (maximal) error Δ for the corresponding series of instances obtained by Algorithm SL, Algorithm SU and Algorithm SM, respectively.

For all series presented in Table 2, the average error Δ of the value $\gamma^k_{p^*}$ of the objective function $\gamma = \sum_{i=1}^{n} w_i C_i$ obtained for the permutation π_k with the largest relative volume of a stability box combined with the lower-point heuristic, the upper-point heuristic and the mid-point heuristic was not greater than 0.012201, 0.012171 and 0.012187, respectively (see series of instances with $n = 1400$ and $\delta\% = 25\%$).

The maximum error Δ of the value $\gamma^k_{p^*}$ of the objective function $\gamma = \sum_{i=1}^{n} w_i C_i$ obtained for the permutation π_k with the largest relative volume of a stability box combined with the lower-point heuristic, the upper-point heuristic and the mid-point heuristic was not greater than 0.013244, 0.013143 and 0.013169, respectively (see the series of instances with $n = 1400$, $\delta\% = 25\%$ and that with $n = 1600$, $\delta\% = 25\%$). In most series tested in our experiments with algorithm MAX-STABOX, Algorithm SM outperformed both Algorithm SL and Algorithm SU.

The CPU-time (column 9) grows slowly with n, and it was not greater than 52.4 s for each instance tested.

5 CONCLUDING REMARKS

In (Sotskov and Lai, 2011), an $O(n^2)$ algorithm has been developed for calculating a permutation $\pi_t \in S$ with the largest volume of a stability box $\mathcal{SB}(\pi_t, T)$ with respect to interval data T.

In Section 3, we proved Properties 1 – 6 of a stability box $\mathcal{SB}(\pi_t, T)$ allowing us to derive an $O(n \log n)$ algorithm for calculating a permutation π_t. The volume of a stability box is an efficient invariant of uncertain interval data, as it is shown in computational experiments on a PC (see Section 4 and Table 2).

ACKNOWLEDGEMENTS

The first and third authors were supported in this research by the National Science Council of Taiwan.

REFERENCES

Daniels, R. and Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single stage production. *Management Science*, V. 41(2):363–376.

Lai, T.-C. and Sotskov, Y. (1999). Sequencing with uncertain numerical data for makespan minimization. *Journal of the Operations Research Society*, V. 50:230–243.

Lai, T.-C., Sotskov, Y., Sotskova, N., and Werner, F. (1997). Optimal makespan scheduling with given bounds of processing times. *Mathematical and Computer Modelling*, V. 26(3):67–86.

Pinedo, M. (2002). *Scheduling: Theory, Algorithms, and Systems*. Prentice-Hall, Englewood Cliffs, NJ, USA.

Sabuncuoglu, I. and Goren, S. (2009). Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research. *International Journal of Computer Integrated Manufacturing*, V. 22(2):138–157.

Slowinski, R. and Hapke, M. (1999). *Scheduling under Fuzziness*. Physica-Verlag, Heidelberg, Germany, New York, USA.

Smith, W. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, V. 3(1):59–66.

Sotskov, Y. and Lai, T.-C. (2011). Minimizing total weighted flow time under uncertainty using dominance and a stability box. *Computers* & *Operations Research*. doi:10.1016/j.cor.2011.02.001.

Sotskov, Y., Sotskova, N., Lai, T.-C., and Werner, F. (2010). *Scheduling under Uncertainty. Theory and Algorithms*. Belorusskaya nauka, Minsk, Belarus.

Sotskov, Y., Wagelmans, A., and Werner, F. (1998). On the calculation of the stability radius of an optimal or an approximate schedule. *Annals of Operations Research*, V. 83:213–252.

Table 2: Computational results for randomly generated instances with $[L,U] = [1,100]$, $w_i \in [1,50]$ and $n \in \{1000,\dots,2000\}$.

$\delta\%$ (Number of jobs)	Volume of $\mathcal{SB}(\pi_t, T)$	Average error			Maximal error			CPU time
		SL	SU	SM	SL	SU	SM	
($n = \mathbf{1000}$)								
0.25%	1	0	0	0	0	0	0	6.1
0.5%	1	0	0	0	0	0	0	6.1
0.75%	≈ 1	0.000038	0.000039	0.000023	0.000044	0.00005	0.000028	6.2
1%	≈ 1	0.00006	0.000057	0.000042	0.000068	0.000061	0.000046	6.1
2.5%	0.2043605	0.000183	0.000176	0.000159	0.000214	0.000207	0.000183	6.2
5%	0.0001695	0.000545	0.000541	0.000524	0.000596	0.000589	0.000578	6.2
15%	≈ 0	0.004294	0.004288	0.004268	0.004805	0.004794	0.004763	6.4
25%	≈ 0	0.01174	0.011709	0.011704	0.012772	0.012727	0.012802	6.7
($n = \mathbf{1200}$)								
0.25%	1	0	0	0	0	0	0	10.5
0.5%	1	0	0	0	0	0	0	10.5
0.75%	≈ 1	0.000038	0.000038	0.000022	0.000042	0.000041	0.000023	10.7
1%	≈ 1	0.000059	0.000057	0.000041	0.000065	0.000062	0.000043	10.4
2.5%	0.0294795	0.000177	0.000172	0.000154	0.000184	0.000181	0.000163	10.8
5%	0.000009	0.000538	0.000538	0.000518	0.000571	0.000564	0.000554	10.7
15%	≈ 0	0.004200	0.004199	0.004178	0.004641	0.004654	0.004618	10.9
25%	≈ 0	0.012056	0.012065	0.012065	0.012637	0.012698	0.01262	11.3
($n = \mathbf{1400}$)								
0.25%	1	0	0	0	0	0	0	16.8
0.5%	≈ 1	0	0	0	0.000001	0.000001	0.000001	16.9
0.75%	≈ 1	0.000041	0.000039	0.000023	0.000044	0.000044	0.000029	17.7
1%	≈ 1	0.00006	0.000059	0.000042	0.000067	0.000063	0.000047	16.8
2.5%	0.0231771	0.000176	0.000175	0.000154	0.000188	0.000185	0.000162	17.1
5%	0.0000288	0.00055	0.000537	0.000523	0.000583	0.000573	0.000563	17.1
15%	≈ 0	0.004417	0.004409	0.004394	0.00459	0.004557	0.004559	17.4
25%	≈ 0	0.012201	0.012171	0.012187	0.013244	0.013143	0.013155	17.6
($n = \mathbf{1600}$)								
0.25%	1	0	0	0	0	0	0	25
0.5%	≈ 1	0	0	0	0.000001	0	0.000001	25.1
0.75%	≈ 1	0.000039	0.000038	0.000022	0.00004	0.000039	0.000024	25
1%	≈ 1	0.00006	0.000058	0.000041	0.000067	0.000062	0.000047	25.1
2.5%	0.0291372	0.00018	0.000177	0.000158	0.000194	0.000188	0.000168	25.2
5%	0.0000068	0.000559	0.000556	0.000539	0.000608	0.000598	0.000574	25.4
15%	≈ 0	0.004348	0.004343	0.00432	0.004646	0.004646	0.004633	25.9
25%	≈ 0	0.012032	0.012009	0.01203	0.013109	0.013135	0.013169	26.7
($n = \mathbf{1800}$)								
0.25%	1	0	0	0	0	0	0	36.2
0.5%	≈ 1	0	0	0	0.000001	0	0.000001	35.6
0.75%	≈ 1	0.00004	0.00004	0.000023	0.000046	0.000045	0.000026	35.6
1%	≈ 1	0.00006	0.000059	0.000042	0.000064	0.000065	0.000048	35.7
2.5%	0.0103524	0.000177	0.000175	0.000156	0.000189	0.000186	0.000165	35.8
5%	0.0000001	0.000549	0.000541	0.000526	0.000564	0.000575	0.000549	36.1
15%	≈ 0	0.004359	0.004345	0.004323	0.004515	0.004536	0.00451	35.6
25%	≈ 0	0.011928	0.011928	0.011936	0.012493	0.012495	0.012492	38.3
($n = \mathbf{2000}$)								
0.25%	1	0	0	0	0	0	0	48.9
0.5%	1	0	0	0	0	0	0	49.8
0.75%	≈ 1	0.000039	0.000038	0.000023	0.000042	0.000041	0.000026	48.5
1%	≈ 1	0.000061	0.00006	0.000043	0.000065	0.000065	0.000046	48.9
2.5%	0.0175329	0.000178	0.000178	0.000157	0.000191	0.000188	0.00017	49.3
5%	0.0000001	0.000546	0.000541	0.000523	0.000571	0.000559	0.00054	49.1
15%	≈ 0	0.004444	0.004431	0.004423	0.004744	0.004718	0.004705	49.6
25%	≈ 0	0.012043	0.012026	0.012029	0.012324	0.012296	0.012294	52.4