

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/225451117>

# Minimizing Total Weighted Completion Time with Uncertain Data: A Stability Approach

Article in Automation and Remote Control · October 2010

DOI: 10.1134/S0005117910100048

---

CITATIONS

9

---

READS

54

3 authors, including:



**Yuri N. Sotskov**

United Institute Of Informatics Problems

135 PUBLICATIONS 1,482 CITATIONS

SEE PROFILE



**Frank Werner**

Otto-von-Guericke-Universität Magdeburg

346 PUBLICATIONS 2,588 CITATIONS

SEE PROFILE

# Minimizing Total Weighted Completion Time with Uncertain Data: A Stability Approach<sup>1</sup>

Yu. N. Sotskov,\* N. G. Egorova,\* and F. Werner\*\*

\**United Institute of Informatics Problems, Belarussian National Academy of Sciences, Minsk, Belarus*

\*\**Otto-von-Guericke-University, Magdeburg, Germany*

Received January 12, 2010

**Abstract**—A single-machine scheduling problem is investigated provided that the input data are uncertain: The processing time of a job can take any real value from the given segment. The criterion is to minimize the total weighted completion time for the  $n$  jobs. As a solution concept to such a scheduling problem with an uncertain input data, it is reasonable to consider a minimal dominant set of job permutations containing an optimal permutation for each possible realization of the job processing times. To find an optimal or approximate permutation to be realized, we look for a permutation with the largest stability box being a subset of the stability region. We develop a branch-and-bound algorithm to construct a permutation with the largest volume of a stability box. If several permutations have the same volume of a stability box, we select one of them due to one of two simple heuristics. The efficiency of the constructed permutations (how close they are to a factually optimal permutation) and the efficiency of the developed software (average CPU-time used for an instance) are demonstrated on a wide set of randomly generated instances with  $5 \leq n \leq 100$ .

**DOI:** 10.1134/S0005117910100048

## 1. INTRODUCTION

Real-life scheduling problems may involve different forms of uncertainty and several approaches are available in the OR literature to deal with uncertain scheduling problems. In the well-developed stochastic approach [1], an uncertain processing time is assumed to be a random variable with a probability distribution which is known a priori. If there is no sufficient information to characterize a priori the probability distribution of each random processing time, other approaches are needed [2, 3]. In particular, in a robust approach [4–6], the decision-maker prefers a schedule that hedges against the worst-case scenario among all the possible realizations of the uncertain processing times. The stability approach developed in [7–12] combines a stability analysis, a two-stage scheduling decision framework, and the solution concept of a minimal dominant set of schedules. A minimal dominant set of schedules being constructed in an off-line fashion optimally covers all the possible scenarios: For any possible scenario such a set contains at least one schedule which is optimal [9–11]. A minimal dominant set of schedules (which may be constructed off-line) allows a scheduler to make an on-line decision whenever additional information on the realization of the processing times becomes available [9, 10].

In this paper, we consider a single-machine scheduling problem with interval processing times of  $n$  jobs which have to be processed. To solve this problem optimally (or approximately), we use the notion of a stability box of a job permutation, which is similar to the well-known stability ball [9, 12–14] used in post-optimality analysis. We use an exact formula for characterizing the

<sup>1</sup> The research of the first author was supported by Belorussian Republican Foundation for Fundamental Research.

stability box of any concrete job permutation in  $O(n \log n)$  time and develop a fast branch-and-bound algorithm to find a permutation with the largest volume of a stability box. We report computational results on finding a permutation with the largest volume of a stability box and satisfying one of two heuristic rules.

2. PROBLEM SETTING, NOTATIONS, STATE-OF-THE-ART

A set of  $n$  jobs  $\mathcal{J} = \{J_1, J_2, \dots, J_n\}$ ,  $n \geq 2$ , has to be processed on a single machine, a weight  $w_i > 0$  being given for job  $J_i \in \mathcal{J}$ . The processing time  $p_i$  of a job  $J_i \in \mathcal{J}$  can take any real value between a lower bound  $p_i^L > 0$  and an upper bound  $p_i^U \geq p_i^L$ , both bounds being known before scheduling. The processing time  $p_i$  may remain unknown until the completion of job  $J_i$  (such a condition is realistic for most real-life scheduling problems).

Let  $T$  denote the set of all vectors  $p = (p_1, p_2, \dots, p_n)$  of the possible processing times. The set  $T$  is a closed rectangular box in the space  $R_+^n$  of non-negative  $n$ -dimensional real vectors and may be represented as the Cartesian product of the  $n$  segments  $[p_i^L, p_i^U]$ ,  $i \in \{1, 2, \dots, n\}$ :

$$T = \left\{ p \mid p \in R_+^n, p_i^L \leq p_i \leq p_i^U, i \in \{1, 2, \dots, n\} \right\} = \times_{i=1}^n [p_i^L, p_i^U]. \tag{1}$$

A vector  $p \in T$  of the possible processing times is called a scenario.

Let  $S = \{\pi_1, \pi_2, \dots, \pi_n!\}$  be the set of all permutations  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n})$  of the jobs  $\mathcal{J} = \{J_{k_1}, J_{k_2}, \dots, J_{k_n}\}$ . If both the permutation  $\pi_k$  of the job set  $\mathcal{J}$  and the scenario  $p \in T$  are fixed, then  $C_i = C_i(\pi_k, p)$  is the completion time of job  $J_i \in \mathcal{J}$  in a semi-active schedule defined by permutation  $\pi_k$ . As usual, a schedule is called semi-active if no job  $J_i \in \mathcal{J}$  can start earlier without delaying the completion time of another job from set  $\mathcal{J}$  and without altering the processing permutation of the jobs  $\mathcal{J}$ . The criterion under consideration is  $\sum w_i C_i$ , which is the minimization of the sum of the weighted job completion times:

$$\sum_{J_i \in \mathcal{J}} w_i C_i(\pi_t, p) = \min_{\pi_k \in S} \left\{ \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p) \right\},$$

where permutation  $\pi_t = (J_{t_1}, J_{t_2}, \dots, J_{t_n}) \in S$  is optimal. By adopting the three-field notation  $\alpha|\beta|\gamma$  introduced in [15], the above scheduling problem is denoted by  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ .

Since the scenario  $p \in T$  may remain unknown before the completion of the jobs  $\mathcal{J}$ , the completion time  $C_i$  of each job  $J_i \in \mathcal{J}$  cannot be calculated at the stage of scheduling. Therefore, problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is not mathematically correct in the sense that the values of the objective function  $\gamma = \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p)$  for different permutations  $\pi_k \in S$  remain uncertain before the completion of the job set  $\mathcal{J}$ .

If a scenario  $p \in T$  is fixed before scheduling (i.e., equalities  $p_i^L = p_i^U = p_i$  hold and segment  $[p_i^L, p_i^U]$  is degenerated into one point  $p_i = [p_i, p_i]$  for each job  $J_i$ ,  $i \in \{1, 2, \dots, n\}$ ), then problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  reduces to problem  $1||\sum w_i C_i$  with deterministic processing times, which is mathematically correct and can be solved in  $O(n \log n)$  time due to Smith [16].

In what follows, the problem  $\alpha|p_i^L \leq p_i \leq p_i^U|\gamma$  with the objective function  $\gamma = f(C_1, C_2, \dots, C_n)$  is called uncertain in contrast to its counterpart, problem  $\alpha||\gamma$ , which is called deterministic. While an optimal sequencing rule for the deterministic problem  $1||\sum w_i C_i$  has been known since 1956 [16], its uncertain counterpart  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  continues to attract the attention of the researchers who develop different approaches for correcting and solving the uncertain problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  (see [4–7, 11, 17, 18] among others). Next, we survey some recent results for scheduling problems with uncertain processing times.

### 2.1. Robust Approach

For problem  $1|p_i^L \leq p_i \leq p_i^U|\gamma$ , there usually does not exist a permutation of the jobs  $\mathcal{J}$  that remains optimal for all scenarios from  $T$ . So, an additional criterion is often introduced for the problem  $1|p_i^L \leq p_i \leq p_i^U|\gamma$  with uncertain data, e.g., a robust schedule minimizing the worst-case deviation from optimality was introduced in [4, 5] to hedge against data uncertainty.

In a robust approach, set  $T$  could contain a continuum of scenarios (i.e.,  $T$  is the Cartesian product of segments as defined in (1)) or set  $T$  could contain a finite number of scenarios:

$$T = \{p^j = (p_1^j, p_2^j, \dots, p_n^j) \mid p^j \in R_+^n, j \in \{1, 2, \dots, m\}\}.$$

For a scenario  $p \in T$ , let  $\gamma_p^t$  denote the optimal value of the objective function  $\gamma = f(C_1, C_2, \dots, C_n)$  for problem  $1||\gamma$  with the fixed scenario  $p$ . Permutation  $\pi_t \in S$  is optimal, if

$$f(C_1(\pi_t, p), \dots, C_n(\pi_t, p)) = \gamma_p^t = \min_{\pi_k \in S} \gamma_p^k = \min_{\pi_k \in S} f(C_1(\pi_k, p), \dots, C_n(\pi_k, p)).$$

For any permutation  $\pi_k \in S$  and any scenario  $p \in T$ , the difference  $\gamma_p^k - \gamma_p^t = r(\pi_k, p)$  is called the regret for permutation  $\pi_k$  with the objective function value equal to  $\gamma_p^k$  under scenario  $p$ . For permutation  $\pi_k \in S$ , value

$$Z(\pi_k) = \max\{r(\pi_k, p) \mid p \in T\}$$

is called the worst-case absolute regret. The worst-case relative regret is defined as follows:

$$Z'(\pi_k) = \max \left\{ \frac{r(\pi_k, p)}{\gamma_p^t} \mid p \in T \right\}$$

provided that  $\gamma_p^t \neq 0$ . In [4, 6], the problem  $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$  of minimizing the total completion time (when all weights are equal:  $w_i = 1$  for each job  $J_i \in \mathcal{J}$ ) has been considered. For a given specific scenario  $p^j \in T$ , the deterministic problem  $1||\sum C_i$  arises which can be solved using the shortest processing times (SPT) rule [16]: Process the jobs of set  $\mathcal{J}$  in non-decreasing order of their processing times  $p_i^j$ ,  $J_i \in \mathcal{J}$ . While the deterministic problem  $1||\sum C_i$  is solvable in  $O(n \log n)$ , finding a permutation  $\pi_t \in S$  of minimizing either the worst-case absolute regret  $Z(\pi_t)$  or the worst-case relative regret  $Z'(\pi_k)$  for the uncertain counterpart  $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$  is binary NP-hard even for two possible scenarios (the corresponding proofs are published in [4, 6], respectively). Only a few very special cases are known to be polynomially solvable for minimizing the worst-case regret for the problems  $1|p_i^L \leq p_i \leq p_i^U|\gamma$ .

In [17], a 2-approximation algorithm has been developed to minimize the worst-case regret for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$ . In [4, 6, 18], exact and heuristic algorithms have been developed and tested to minimize the worst-case regret for the same problem.

### 2.2. Stability Approach

In what follows, we adopt the stability approach [7–9] for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  considering set  $T$  as the continuum of possible scenarios defined by (1). This stability approach combines a stability analysis [9, 13, 14, 19–21], a two-stage scheduling decision framework (the off-line planning stage and the on-line scheduling stage) [9, 10, 22], and the solution concept of a minimal dominant set of semi-active schedules [7–9, 11, 12] (a minimal dominant set of permutations in the case of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ ).

As a solution concept to an uncertain scheduling problem, it is reasonable to consider a minimal dominant set of job permutations (semi-active schedules) defined as follows.

**Definition 1.** A set of permutations (semi-active schedules)  $S(T) \subseteq S$  is a *minimal dominant set* for problem  $\alpha|p_i^L \leq p_i \leq p_i^U|\gamma$ , if

- (a) for any fixed scenario  $p \in T$ , set  $S(T)$  contains at least one permutation (semi-active schedule), which is optimal for the deterministic counterpart  $\alpha||\gamma$  associated with scenario  $p$ ,
- (b) property (a) is lost for any proper subset of set  $S(T)$ .

Due to condition (b), the above set  $S(T)$  is a minimal dominant set with respect to inclusion. Set  $S(T)$  has been investigated in [7–9, 23–25] for the makespan criterion, and in [9, 12, 21, 24, 26] for the total completion time criterion. Article [26] addresses the total completion time in a two-machine flow-shop problem  $F2|p_i^L \leq p_i \leq p_i^U|\sum C_i$ . A geometrical algorithm has been developed for solving the flow-shop problem  $Fm|p_i^L \leq p_i \leq p_i^U, n = 2|\sum C_i$  with  $m$  machines and two jobs. For an uncertain flow-shop scheduling problem with two or three machines, sufficient conditions were identified when a transposition of two jobs minimizes the total completion time. The work of [24] deals either with the criterion  $C_{\max}$  or with  $\sum C_i$ , where the processing times are fixed while the setup times belong to the given segments. Dominance relations were identified for an uncertain flow-shop scheduling problem with two machines. In [23], for a two-machine flow-shop problem  $F2|p_i^L \leq p_i \leq p_i^U|C_{\max}$  sufficient conditions were identified when a transposition of two jobs minimizes  $C_{\max}$ . In [12], for a job-shop problem  $Jm|p_i^L \leq p_i \leq p_i^U|\sum C_i$  with  $m$  machines, several exact and heuristic algorithms were developed by using the disjunctive graph model and computational results have been reported.

Before presenting heuristic algorithms for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , we remind some known results for the uncertain scheduling problem and for its deterministic counterpart.

In [16], it was proven that problem  $1||\sum w_i C_i$  can be solved in  $O(n \log n)$  time due to the following sufficient condition for the optimality of permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$ :

$$\frac{w_{k_1}}{p_{k_1}} \geq \frac{w_{k_2}}{p_{k_2}} \geq \dots \geq \frac{w_{k_n}}{p_{k_n}}, \tag{2}$$

where inequality  $p_{k_i} > 0$  holds for each job  $J_{k_i} \in \mathcal{J}$ . Thus, problem  $1||\sum w_i C_i$  can be solved to optimality by the weighted shortest processing times (WSPT) rule: Process the jobs in non-increasing order of their weight-to-process ratio  $\frac{w_{k_i}}{p_{k_i}}$ . Inequalities (2) provide also a necessary condition for the optimality of permutation  $\pi_k \in S$ , see [27].

**Theorem 1** [16, 27]. *Permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$  is optimal for the deterministic problem  $1||\sum w_i C_i$  if and only if inequalities (2) hold.*

A minimal dominant set  $S(T)$  for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  may be determined by using the following dominance relation on the set of jobs  $\mathcal{J}$ .

**Definition 2.** Job  $J_u$  dominates job  $J_v$  with respect to  $T$  (this will be denoted by  $J_u \mapsto J_v$ ) if there exists a minimal dominant set  $S(T)$  for 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 set  $S(T)$ .

Definition 2 implies that a minimal dominant set constructed for the deterministic problem  $1||\sum w_i C_i$  associated with a scenario  $p \in T$  is a singleton,  $S(T) = \{\pi_k\}$ , where  $T = \{p\}$ . Hence, relations  $J_{k_1} \mapsto J_{k_2} \mapsto J_{k_3} \mapsto \dots \mapsto J_{k_{n-1}} \mapsto J_{k_n}$  with respect to  $T = \{p\}$  hold for the deterministic problem  $1||\sum w_i C_i$ . The following claim has been proven in [11].

**Theorem 2** [11]. *For problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , job  $J_u$  dominates job  $J_v$  with respect to  $T$  if and only if*

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

The cardinality  $|S(T)|$  of a minimal dominant set  $S(T)$  may be considered as a measure of uncertainty for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ . In the least uncertain case, a minimal dominant set for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is a singleton,  $\{\pi_k\} = S(T)$ , which is also a solution to the deterministic counterpart  $1|\sum w_i C_i$  associated with any scenario  $p \in T$ . This case was characterized in [11] as follows.

**Theorem 3** [11]. *For the existence of a dominant singleton  $S(T) = \{\pi_k\} = \{(J_{k_1}, J_{k_2}, \dots, J_{k_n})\}$  for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , inequalities (4) are necessary and sufficient:*

$$\frac{w_{k_1}}{p_{k_1}^U} \geq \frac{w_{k_2}}{p_{k_2}^L}; \quad \frac{w_{k_2}}{p_{k_2}^U} \geq \frac{w_{k_3}}{p_{k_3}^L}; \quad \dots; \quad \frac{w_{k_{n-1}}}{p_{k_{n-1}}^U} \geq \frac{w_{k_n}}{p_{k_n}^L}. \quad (4)$$

The most uncertain case of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is that with  $|S(T)| = n!$ . This case was also characterized in [11].

**Theorem 4** [11]. *Let  $p_i^L < p_i^U$ ,  $J_i \in \mathcal{J}$ . For the existence of a minimal dominant set  $S(T)$  for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  with a maximum cardinality  $|S(T)| = n!$ , inequality (5) is necessary and sufficient:*

$$\max \left\{ \frac{w_i}{p_i^U} \mid J_i \in \mathcal{J} \right\} < \min \left\{ \frac{w_i}{p_i^L} \mid J_i \in \mathcal{J} \right\}. \quad (5)$$

In [28], the uniqueness of a minimal dominant set  $S(T)$  was investigated. Let notation  $1|p|\sum w_i C_i$  be used for indicating an individual problem (an instance) of the mass problem  $1|\sum w_i C_i$  associated with a specific scenario  $p$ . We shall use the notation  $a = \min \left\{ \frac{w_i}{p_i^U} \mid J_i \in \mathcal{J} \right\}$  and the notation  $b = \max \left\{ \frac{w_i}{p_i^L} \mid J_i \in \mathcal{J} \right\}$ . The criterion of the uniqueness of a minimal dominant set  $S(T)$  uses the following subsets  $\mathcal{J}_r$ ,  $r \in [a, b]$ , of the job set  $\mathcal{J}$ :

$$\mathcal{J}_r = \left\{ J_i \in \mathcal{J} \mid r = \frac{w_i}{p_i^U} = \frac{w_i}{p_i^L} \right\}. \quad (6)$$

**Theorem 5** [28]. *For problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , a minimal dominant set  $S(T)$  is uniquely determined if and only if there is no real  $r \in [a, b]$  such that  $|\mathcal{J}_r| \geq 2$ .*

**Theorem 6** [28]. *Let  $S(T)$  be a minimal dominant set for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ . For any permutation  $\pi_k \in S(T)$ , there exists a scenario  $p \in T$  such that  $\pi_k$  is the unique optimal permutation for the instance  $1|p|\sum w_i C_i$  if and only if there is no  $r \in [a, b]$  such that  $|\mathcal{J}_r| \geq 2$ .*

Theorem 4 is generalized in [28] as follows.

**Theorem 7** [28]. *Let there exist no real  $r \in [a, b]$  such that  $|\mathcal{J}_r| \geq 2$ . For the existence of a minimal dominant set  $S(T)$  for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  attaining the maximum cardinality  $|S(T)| = n!$ , inequality (5) is necessary and sufficient.*

Since the cardinality of a minimal dominant set could range from 1 (Theorem 3) to  $n!$  (Theorems 4 and 7), it is impossible to generate in polynomial time all the elements of set  $S(T)$  and so there is no polynomial algorithm for enumerating all permutations of a set  $S(T)$ . Fortunately, due to Theorem 2, one can obtain a compact presentation of a minimal dominant set  $S(T)$  for a problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  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, one can check condition (3) for each pair of jobs  $J_u$  and  $J_v$ , from set  $\mathcal{J}$  and construct a digraph  $(\mathcal{J}, \mathcal{A})$  of the dominance relation on the set  $\mathcal{J}$  as follows: Arc  $(J_u, J_v)$  belongs to set  $\mathcal{A}$  if and only if  $J_u \mapsto J_v$ . To construct the digraph  $(\mathcal{J}, \mathcal{A})$  takes  $O(n^2)$  time.

**Theorem 8** [28]. *Digraph  $(\mathcal{J}, \mathcal{A})$  constructed for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  defines a strict order relation on the set  $\mathcal{J}$  if and only if there is no a real  $r \in [a, b]$  such that  $|\mathcal{J}_r| \geq 2$ .*

If there exists a real  $r \in [a, b]$  with inequality  $|\mathcal{J}_r| \geq 2$ , then due to Theorem 5, there exist at least two minimal dominant sets for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ . The whole number of minimal dominant sets for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  was calculated in [28] as follows.

**Theorem 9** [28]. *If inequality  $|\mathcal{J}_{r_q}| \geq 2$  holds for each real  $r_q \in \{r_1, r_2, \dots, r_m\}$ , where integer  $m \geq 1$  is maximal and  $r_q \in [a, b]$ , then the number  $k$  of the minimal dominant sets existing for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is equal to  $\prod_{q=1}^m |\mathcal{J}_{r_q}|!$ . If  $m = 0$ , then  $k = 1$ .*

From Theorems 5–9, it follows that the existence of sets  $\mathcal{J}_{r_q}$  with  $|\mathcal{J}_{r_q}| \geq 2, r_q \in \{r_1, r_2, \dots, r_m\}$ , implies that a minimal dominant set  $S(T)$  loses the useful properties: If there exists at least one set  $\mathcal{J}_{r_q}$  which is not a singleton, then the binary relation  $\mathcal{A} \subseteq \mathcal{J} \times \mathcal{J}$  is not a strict order (Theorem 8); a minimal dominant set is not uniquely determined (Theorem 5); the number of minimal dominant sets existing for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  may be very large (Theorem 9). Next, we show how to overcome all these difficulties over set  $\mathcal{J}_{r_q}, |\mathcal{J}_{r_q}| \geq 2$ , and how to use these sets while solving a problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  (when no additional criterion is implied by the data uncertainty). The size  $n$  of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  will be reduced by the quantity  $|\mathcal{J}_{r_q}| - 1$  for each non-singleton  $\mathcal{J}_{r_q}$  via identifying a set of jobs  $\mathcal{J}_{r_q}$  in one job of them.

Due to Theorem 1, in any optimal permutation  $\pi_l \in S$  generated by the instance  $1|p|\sum w_i C_i$ , all the jobs of set  $\mathcal{J}_{r_q} \subseteq \mathcal{J}$  must be located adjacently one by one:  $\pi_l = (\dots, \pi(\mathcal{J}_{r_q}), \dots)$ , where  $\pi(\mathcal{J}_{r_q})$  is a permutation of the jobs  $\mathcal{J}_{r_q}$ . Moreover, the order of the jobs  $\{J_{q(1)}, J_{q(2)}, \dots, J_{q(|\mathcal{J}_{r_q}|)}\} = \mathcal{J}_{r_q}$  in permutation  $\pi(\mathcal{J}_{r_q})$  does not influence the value of the objective function  $\gamma = \sum_{i=1}^n w_i C_i$  calculated for any permutation  $\pi_k \in S$  of the form  $\pi_k = (\dots, \pi(\mathcal{J}_{r_q}), \dots)$  (since the processing time of each job  $J_{q(v)} \in \mathcal{J}_{r_q}$  is fixed and the weight-to-process ratios are the same for all the jobs of set  $\mathcal{J}_{r_q}$ ). Therefore, while looking for an optimal permutation for any instance  $1|p|\sum w_i C_i$  generated by the uncertain problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  via fixing a scenario  $p \in T$ , one can treat all the jobs  $\{J_{q(1)}, J_{q(2)}, \dots, J_{q(|\mathcal{J}_{r_q}|)}\} = \mathcal{J}_{r_q}$  as one job with the numerical parameters (weight and processing time) equal to those of any job of set  $\mathcal{J}_{r_q}$ . By choosing only one job from each of such sets  $\mathcal{J}_{r_q}, r_q \in \{r_1, r_2, \dots, r_m\}, |\mathcal{J}_{r_q}| \geq 2$ , the original instance of the uncertain problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  can be transformed into an equivalent instance (let this instance be denoted as  $1^*|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ ) with a smaller cardinality of the set of jobs to be scheduled (let this set be denoted as  $\mathcal{J}^*$ ):

$$|\mathcal{J}^*| = |\mathcal{J}| - \sum_{q=1}^m (|\mathcal{J}_{r_q}| - 1) = n + m - \sum_{q=1}^m |\mathcal{J}_{r_q}|.$$

In the following claim,  $1^*|p|\sum w_i C_i$  denotes the deterministic instance generated by the uncertain instance  $1^*|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  via fixing a scenario  $p \in T$ .

**Theorem 10** [28]. *An instance  $1^*|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is equivalent to the original instance of the uncertain problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  in the sense that for any fixed scenario  $p \in T$ , an optimal permutation  $\pi_k$  of the instance  $1^*|p|\sum w_i C_i$  is obtained from the corresponding optimal permutation  $\pi_t$  of the instance  $1|p|\sum w_i C_i$  of the original uncertain problem via deleting from permutation  $\pi_t$  all the jobs of set  $\mathcal{J} \setminus \mathcal{J}^*$ .*

Along with a smaller size, the equivalent instance  $1^*|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  has a unique minimal dominant set  $S(T)$  (due to Theorem 5). Consequently,  $S(T)$  is a minimal dominant set with respect to both inclusion and cardinality. The next useful property of the instance  $1^*|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  is that relation  $\mathcal{A} \subseteq \mathcal{J} \times \mathcal{J}$  is a strict order on the set of jobs  $\mathcal{J}$  (due to Theorem 8)

and the corresponding digraph  $(\mathcal{J}, \mathcal{A})$  has neither a loop nor a contour. Due to Theorem 6, for any permutation  $\pi_k \in S(T)$ , there exists a scenario  $p \in T$  such that  $\pi_k$  is the unique optimal permutation for the instance  $1|p|\sum w_i C_i$ .

Instead of using the digraph  $(\mathcal{J}, \mathcal{A})$ , one can adopt a reduction  $G = (\mathcal{J}, \mathcal{A}^0)$  of digraph  $(\mathcal{J}, \mathcal{A})$ , which is obtained from the latter via deleting the transitive arcs  $\mathcal{A} \setminus \mathcal{A}^0$ .

### 2.3. Properties of a Stability Box and Region

In [28], the notion of a stability box of a permutation  $\pi_k \in S$  has been introduced. A stability box is a subset of the stability region [9, 12, 13, 20] and is similar to a stability ball investigated in the papers [9, 12–14, 20] within a post-optimality analysis of the optimal permutation constructed for the deterministic scheduling problem.

**Definition 3.** A maximal closed rectangular box  $\mathcal{SB}(\pi_k, T) = \times_{i=1}^n [l_i, u_i] \subseteq T$  is called a stability box of permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$  with respect to  $T$ , if for each  $i \in \{1, 2, \dots, n\}$ , permutation  $\pi_k$  remains optimal for the instance  $1|p'|\sum w_i C_i$  with any scenario  $p' = (p'_1, p'_2, \dots, p'_n) \in \left\{ \times_{j=1, j \neq i}^n [p_{k_j}^L, p_{k_j}^U] \right\} \times [l_i, u_i]$ . If there does not exist a scenario  $p \in T$  such that permutation  $\pi_k$  is optimal for the instance  $1|p|\sum w_i C_i$ , then  $\mathcal{SB}(\pi_k, T) = \emptyset$ .

In Definition 3, the maximality of a rectangular box  $\mathcal{SB}(\pi_k, T)$  means that for each position  $i \in \{1, 2, \dots, n\}$  in permutation  $\pi_k$ , the lower bound  $l_i$  (the upper bound  $u_i$ ) for the variation of the processing time  $p_{k_i}$  of job  $J_{k_i}$ , which is located at position  $i$  in permutation  $\pi_k$ , preserving the optimality of permutation  $\pi_k$  has to be as small (as large) as possible provided that the processing time of each other job  $J_{k_j}$ ,  $j \in \{1, 2, \dots, n\} \setminus \{i\}$ , may vary independently and simultaneously within the whole given segment  $[p_{k_j}^L, p_{k_j}^U]$ . We call the dimension of a stability box  $\mathcal{SB}(\pi_k, T)$  the cardinality  $|N_k|$  of the set  $\{p_{k_i} \mid k_i \in N_k\}$  of the processing times in the scenario  $p'$  which may be modified in vector  $p$  with preserving the optimality of permutation  $\pi_k$ . The cardinality  $|N_k|$  of set  $N_k$  is an important characteristic of the stability box  $\mathcal{SB}(\pi_k, T)$ : it defines the maximum number of the processing times in  $p'$  which are modifiable in scenario  $p$  without violating the optimality of permutation  $\pi_k$ . Note that the processing times of the remaining set  $\{p'_{k_j} \mid k_j \in N \setminus N_k\}$  have to remain the same as those in the original vector  $p$ :  $p'_{k_j} = p_{k_j}$ .

In [12, 13, 20], the stability region of an optimal semi-active schedule was investigated for a job-shop problem with the mean flow time and the makespan criterion, respectively. Using the notations introduced for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , the stability region  $\mathcal{K}(\pi_k, T)$  of a permutation  $\pi_k \in S$  with respect to  $T$  is defined as follows:

$$\mathcal{K}(\pi_k, T) = \left\{ p \mid p \in T, \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p) = \min_{\pi_l \in S} \left\{ \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_l, p) \right\} \right\}. \quad (7)$$

Definition 3 of a stability box and definition (7) of a stability region imply inclusion

$$\mathcal{SB}(\pi_k, T) \subseteq \mathcal{K}(\pi_k, T).$$

In our heuristic algorithms for solving problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ , we shall use the stability box in spite of set  $\mathcal{SB}(\pi_k, T)$  being often a proper subset of set  $\mathcal{K}(\pi_k, T)$ . We can argue for such a choice by the simplicity of a stability box allowing us to provide a polynomial time for calculating  $\mathcal{SB}(\pi_k, T)$  for a permutation  $\pi_k \in S(T)$ .

In [28], the following properties of the stability box and the stability region have been derived.

**Theorem 11** [28]. *The stability box  $\mathcal{SB}(\pi_k, T)$  (the stability region  $\mathcal{K}(\pi_k, T)$ ) is empty, if and only if there is no scenario  $p \in T$  such that permutation  $\pi_k$  is optimal for the instance  $1|p|\sum w_i C_i$ .*

**Theorem 12** [28]. *There exists a scenario  $p \in T$  such that permutation  $\pi_k = (J_{k_1}, \dots, J_{k_n}) \in S$  is optimal for the instance  $1|p| \sum w_i C_i$  if and only if there is no job  $J_{k_i}$ ,  $i \in \{1, 2, \dots, n-1\}$ , such that inequality*

$$\frac{w_{k_i}}{p_{k_i}^L} < \frac{w_{k_j}}{p_{k_j}^U} \tag{8}$$

*holds for at least one job  $J_{k_j}$ , where  $j \in \{i+1, i+2, \dots, n\}$ .*

**Theorem 13** [28]. *The stability box  $\mathcal{SB}(\pi_k, T)$  (stability region  $\mathcal{K}(\pi_k, T)$ ) is empty, if and only if there exists job  $J_{k_i}$ ,  $i \in \{1, 2, \dots, n-1\}$ , such that inequality (8) holds for at least one job  $J_{k_j}$ , where  $j \in \{i+1, i+2, \dots, n\}$ .*

Definitions 3 and (7) imply the following claim.

**Theorem 14** [28]. *If there exists exactly one scenario  $p \in T$  such that permutation  $\pi_k \in S$  is optimal for the instance  $1|p| \sum w_i C_i$ , then  $\mathcal{SB}(\pi_k, T) = \{p\} = \mathcal{K}(\pi_k, T)$ .*

Theorem 15 characterizes another extreme case for the stability box and the stability region.

**Theorem 15** [28].  *$\mathcal{SB}(\pi_k, T) = T = \mathcal{K}(\pi_k, T)$  if and only if inequalities (4) hold.*

As follows from Definition 1, one can restrict the search by the permutations of a minimal dominant set  $S(T)$  while solving problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$  exactly. For each permutation of set  $S(T)$ , both the stability region and the stability box are non-empty, i.e., the following claim holds.

**Theorem 16** [28]. *If  $\pi_k \in S(T)$ , then  $\mathcal{SB}(\pi_k, T) \neq \emptyset$  and  $\mathcal{K}(\pi_k, T) \neq \emptyset$ .*

In the next subsection, we present an  $O(n \log n)$ -algorithm STABOX for calculating the stability box  $\mathcal{SB}(\pi_k, T)$  for a permutation  $\pi_k = (J_{k_1}, \dots, J_{k_{i-1}}, J_{k_i}, J_{k_{i+1}}, \dots, J_{k_n}) \in S$ . In Section 3, we show how to use this algorithm for solving approximately problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ .

#### 2.4. Polynomial Algorithm for Calculating the Stability Box

Due to the additivity of the objective function  $\gamma = \sum w_i C_i$ , in order to find a rectangular box  $\mathcal{SB}(\pi_k, T)$ , it is sufficient to calculate the maximal range of the possible variation of each processing time  $p_{k_i}$ ,  $i \in \{1, 2, \dots, n\}$ , which preserves the optimality of permutation  $\pi_k$ . Let a “possible variation”  $[l_{k_i}, u_{k_i}]$  (respectively,  $[L_{k_i}, U_{k_i}]$ ) of the processing time  $p_{k_i}$  (of the weight-to-process ratio of job  $J_{k_i}$ ) mean the following. If  $\pi_k$  is an optimal permutation for the instance  $1|p| \sum w_i C_i$  with  $p = (p_1, p_2, \dots, p_n) \in T$ , then permutation  $\pi_k$  remains optimal for any instance  $1|p'| \sum w_i C_i$  with  $p' = (p'_1, p'_2, \dots, p'_n) \in T$ , where  $p'_t = p_t$  for each  $t \neq k_i$  and  $p_{k_i} \in [l_{k_i}, u_{k_i}]$  (respectively,  $\frac{w_{k_i}}{p_{k_i}} \in [L_{k_i}, U_{k_i}]$ ). It is easy to show that the lower bound  $d_{k_i}^-$  for the maximal possible variation of the weight-to-process ratio is as follows:

$$d_{k_i}^- = \max \left\{ \frac{w_{k_i}}{p_{k_i}^U}, \max_{i < j \leq n} \left\{ \frac{w_{k_j}}{p_{k_j}^L} \right\} \right\}. \tag{9}$$

The upper bound  $d_{k_i}^+$  for the maximal possible variation of the weight-to-process ratio is as follows:

$$d_{k_i}^+ = \min \left\{ \frac{w_{k_i}}{p_{k_i}^L}, \min_{1 \leq j < i} \left\{ \frac{w_{k_j}}{p_{k_j}^U} \right\} \right\}. \tag{10}$$

In [28], the following claims have been proven.

**Theorem 17** [28]. *If there is no job  $J_{k_i}$ ,  $i \in \{1, 2, \dots, n-1\}$ , in permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$  such that inequality (8) holds for at least one job  $J_{k_j}$ , where  $j \in \{i+1, i+2, \dots, n\}$ , then*

$$\mathcal{SB}(\pi_k, T) = \times_{d_i^- \leq d_i^+} \left[ \frac{w_{k_i}}{d_{k_i}^+}, \frac{w_{k_i}}{d_{k_i}^-} \right] \times \left\{ \times_{d_j^- > d_j^+} [p_{k_j}, p_{k_j}] \right\}. \quad (11)$$

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

If  $d_j^- > d_j^+$ , then there is no possible variation of the weight-to-process ratio for job  $J_{k_j}$ . The dimension  $|N_k|$  of the stability box  $\mathcal{SB}(\pi_k, T)$  is equal to the number of jobs  $J_{k_i} \in \mathcal{J}$  for which the opposite inequality  $d_i^- \leq d_i^+$  holds. The maximal possible variation of the weight-to-process ratio  $[L_{k_i}, U_{k_i}]$  of such a job  $J_{k_i}$  is not empty.

**Corollary 1** [28]. *If  $\mathcal{SB}(\pi_k, T) \neq \emptyset$  for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  with the scenario set  $T$ , then the singleton  $\{\pi_k\}$  is a minimal dominant set for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  with the scenario set  $T^* = \mathcal{SB}(\pi_k, T)$ .*

The following  $O(n \log n)$ -algorithm is based on Theorem 17.

#### **Algorithm STABOX** [28]

**Input:** Segments  $[p_i^L, p_i^U]$ , weights  $w_i$ ,  $J_i \in \mathcal{J}$ ; permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$ .

**Output:** Stability box  $\mathcal{SB}(\pi_k, T)$ .

*Step 1:* Construct the list  $L$  of fractions  $\frac{w_{k_i}}{p_{k_i}^L}$ ,  $i = 1, 2, \dots, n$ , in non-increasing order;

find the position  $r_i$  of element  $\frac{w_{k_i}}{p_{k_i}^L}$  in the list  $L$ , let  $L_{r_i} = \frac{w_{k_i}}{p_{k_i}^L}$ .

*Step 2:* Construct the list  $U$  of fractions  $\frac{w_{k_i}}{p_{k_i}^U}$ ,  $i = 1, 2, \dots, n$ , in non-decreasing order;

find the position  $m_i$  of element  $\frac{w_{k_i}}{p_{k_i}^U}$  in the list  $U$ , let  $U_{m_i} = \frac{w_{k_i}}{p_{k_i}^U}$ .

*Step 3:* Construct the list  $U^0$  of fractions  $\frac{w_{k_i}}{p_{k_i}^U}$ ,  $i = 1, 2, \dots, n$ , in non-increasing order;

find the position  $t_i$  of element  $\frac{w_{k_i}}{p_{k_i}^U}$  in the list  $U^0$ , let  $U_{t_i}^0 = \frac{w_{k_i}}{p_{k_i}^U}$ .

*Step 4:* **FOR**  $i = 1$  to  $n$  **DO**

set  $U^0 := U^0 \setminus \{U_{t_i}^0\}$ ; test inequality  $\frac{w_{k_i}}{p_{k_i}^L} < U_1^0$ ,

where  $U_1^0$  is the first (maximal) element in the list  $U^0$ ;

**IF** inequality  $\frac{w_{k_i}}{p_{k_i}^L} < U_1^0$  holds

**THEN**  $\mathcal{SB}(\pi_k, T) = \emptyset$  **STOP.**

**END FOR**

*Step 5:* **FOR**  $i = 1$  to  $n-1$  **DO**

set  $L := L \setminus \{L_{r_i}\}$ ; calculate  $d_{k_i}^- = \max \left\{ \frac{w_{k_i}}{p_{k_i}^L}, L_1 \right\}$ ,

where  $L_1$  is the first (maximal) element in the list  $L$ .

**END FOR**

*Step 6:* **FOR**  $i = n$  to  $2$  **DO**

set  $U := U \setminus \{U_{m_i}\}$ ; calculate  $d_{k_i}^+ = \max \left\{ \frac{w_{k_i}}{p_{k_i}^U}, U_1 \right\}$ ,

where  $U_1$  is the first (minimal) element in the list  $U$ .

**END FOR**

Step 7: Set  $d_{k_n}^- := \frac{w_{k_n}}{p_{k_n}^U}$ ;  $d_{k_1}^+ := \frac{w_{k_1}}{p_{k_1}^L}$ .

Step 8: **FOR**  $J_i \in \mathcal{J}$  **DO**

**IF**  $d_{k_i}^+ < d_{k_i}^-$  **THEN** processing time  $p_{k_i}$  has to be fixed in  $\mathcal{SB}(\pi_k, T)$

**ELSE**  $\left[ \frac{w_{k_i}}{d_{k_i}^+}, \frac{w_{k_i}}{d_{k_i}^-} \right]$  is the maximal range of the possible variation of  $p_{k_i}$ .

**END FOR**

Step 9: Set  $\mathcal{SB}(\pi_k, T) := \times_{d_i^- \leq d_i^+} \left[ \frac{w_{k_i}}{d_{k_i}^+}, \frac{w_{k_i}}{d_{k_i}^-} \right] \times \{ \times_{d_j^- > d_j^+} [p_{k_j}, p_{k_j}] \}$  **STOP**.

Formula (11) is valid for any permutation  $\pi_k \in S$  with a non-empty stability box, e.g., for each permutation in the set  $S(T)$ , since the following claim holds.

**Theorem 18** [28]. *If  $\pi_k \in S(T)$ , then  $\mathcal{SB}(\pi_k, T) \neq \emptyset$  and  $\mathcal{K}(\pi_k, T) \neq \emptyset$ .*

In Section 3, we show how to solve problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$  approximately using a minimal dominant set  $S(T)$  and a permutation  $\pi_k \in S(T)$  with the stability box  $\mathcal{SB}(\pi_k, T)$  of maximal relative volume. The permutation, which has the largest relative volume of the stability box, seems to be the most attractive one among the permutations of a minimal dominant set.

### 3. A PERMUTATION WITH THE LARGEST STABILITY BOX

The above results motivate us for developing a branch-and-bound algorithm for solving problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$  approximately. Intuitively, a permutation with a larger volume of the stability box is better than that with a smaller volume of the stability box. For simplicity, the presentation of the algorithm will be based on the following example.

#### 3.1. Example

The input data for the example of problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$  are given in columns 1–4 of Table 1.

For each pair of jobs  $J_u \in \mathcal{J}^*$  and  $J_v \in \mathcal{J}^*$ , we check condition (3) as follows:

$$\begin{aligned} \frac{w_1}{p_1^U} = 80 \geq 50 = \frac{w_3}{p_3^L}; \quad \frac{w_1}{p_1^U} = 80 \geq 40 = \frac{w_4}{p_4^L}; \quad \frac{w_2}{p_2^U} = 60 \geq 50 = \frac{w_3}{p_3^L}; \\ \frac{w_2}{p_2^U} = 60 \geq 40 = \frac{w_4}{p_4^L}; \quad \frac{w_4}{p_4^U} = 40 \geq 40 = \frac{w_5}{p_5^L}; \quad \frac{w_4}{p_4^U} = 40 \geq 40 = \frac{w_6}{p_6^L}. \end{aligned}$$

Thus, condition (3) is satisfied for the following pairs of ordered jobs:  $\{J_1, J_3\}$ ,  $\{J_1, J_4\}$ ,  $\{J_2, J_3\}$ ,  $\{J_2, J_4\}$ ,  $\{J_4, J_5\}$ ,  $\{J_4, J_6\}$ . Due to Theorem 2, the following relations hold:  $J_1 \mapsto J_3$ ,  $J_1 \mapsto J_4$ ,  $J_2 \mapsto J_3$ ,  $J_2 \mapsto J_4$ ,  $J_4 \mapsto J_5$ ,  $J_4 \mapsto J_6$ . A minimal dominant set  $S(T)$  is defined by the digraph

**Table 1.** Data for the example

1	2	3	4	5	6	7	8	9	10
$i$	$p_i^L$	$p_i^U$	$w_i$	$\frac{w_i}{p_i^L}$	$\frac{w_i}{p_i^U}$	$d_i^-$	$d_i^+$	$\frac{w_i}{d_i^+}$	$\frac{w_i}{d_i^-}$
1	4	5	400	100	80	90	100	4	$4\frac{4}{9}$
2	6	9	540	90	60	60	80	$6\frac{3}{4}$	9
3	4	10	200	50	20	40	50	4	5
4	6	6	240	40	40	40	40	6	6
5	3	4	120	40	30	40	20	–	–
6	4	16	160	40	10	10	20	8	16

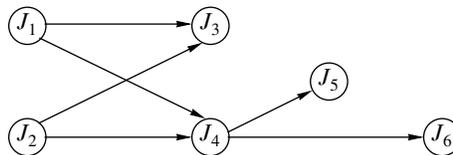


Fig. 1. The reduction of digraph  $G = (\mathcal{J}, \mathcal{A})$  defining a unique minimal dominant set  $S(T)$ .

$G = (\mathcal{J}, \mathcal{A})$  (the reduction of this digraph is represented in Fig. 1). Due to Theorem 5, a minimal dominant set is unique for this example.

3.2. Branch-and-bound Algorithm

For finding a permutation with the largest relative volume of a stability box, we develop a tree-like algorithm called MAXSTABOX. Let  $\mathcal{T} = (V, E)$  denote the solution tree, where  $V$  is the set of vertices and  $E$  is the set of edges. The solution tree  $\mathcal{T} = (V, E)$  constructed for the above example is represented in Fig. 2. Each vertex of the solution tree  $\mathcal{T} = (V, E)$  is a permutation  $\pi^{k,m} = (J_{k_1}, J_{k_2}, \dots, J_{k_m}) \in V, 1 \leq m \leq n$ , of some jobs from set  $\mathcal{J}$ . Edge  $[\pi^{k,m}, \pi^{l,m+1}]$  belongs to set  $E$ , if permutation  $\pi^{l,m+1} = (J_{l_1}, J_{l_2}, \dots, J_{l_{m+1}}), m \leq n - 1$ , was obtained from permutation  $\pi^{k,m} = (J_{k_1}, J_{k_2}, \dots, J_{k_m})$ , i.e., equality  $J_{k_i} = J_{l_i}$  holds for each index  $i \in \{1, 2, \dots, m\}$ .

The root of the solution tree  $\mathcal{T} = (V, E)$  is the empty permutation (we denote the empty permutation by  $\pi^{*,0}$ ). The search process is started with the solution tree defined as follows:

$$\mathcal{T} := (\{\pi^{*,0}\}, \emptyset).$$

The set of vertices of rank  $h = 1$  in the solution tree is defined by the set of jobs  $X(h) \subseteq \mathcal{J}$ , which have no predecessors in the digraph  $G = (\mathcal{J}, \mathcal{A})$ . In the above example, we obtain  $X(h) = \{J_1, J_2\}$  since both jobs  $J_1$  and  $J_2$  have no predecessors in the digraph  $G = (\mathcal{J}, \mathcal{A})$  (see Fig. 1).

In the first iteration, the solution tree  $\mathcal{T} = (V, E)$  for the example is constructed as follows:

$$V := \{\pi^{*,0}, \pi^{1,1} = (J_1), \pi^{2,1} = (J_2)\}, \quad E := \{[\pi^{*,0}, (J_1)], [\pi^{*,0}, (J_2)]\}.$$

This means that either job  $J_1$  or job  $J_2$  may be located at the first position in the desired permutation with the largest relative volume of a stability box.

The set of vertices of rank  $h = 2$  in the solution tree is defined by the set of jobs  $X(h)$ , which have no predecessors in the digraph obtained from  $G = (\mathcal{J}, \mathcal{A})$  after deleting the vertex of set  $X(h - 1) = X(1)$ .

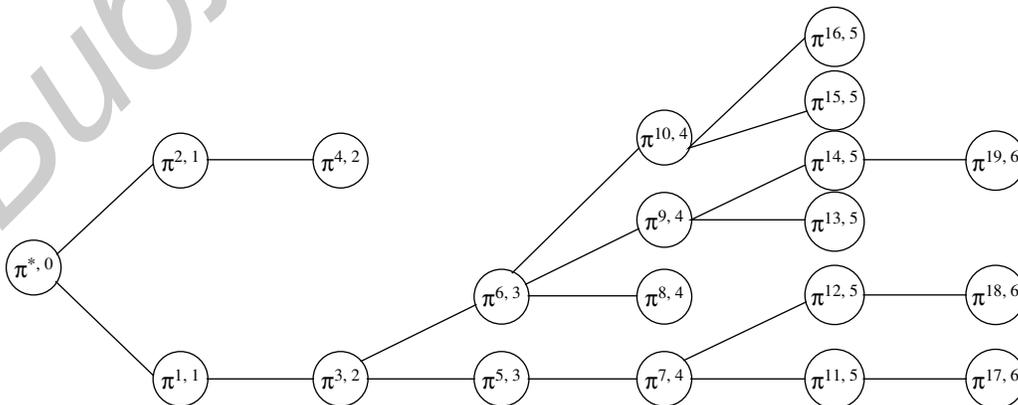


Fig. 2. Solution tree  $\mathcal{T} = (V, E)$  constructed for the example.

In the second iteration, the solution tree  $\mathcal{T} = (V, E)$  for the example is constructed as follows:

$$V := \left\{ \pi^{*,0}, \pi^{1,1} = (J_1), \pi^{2,1} = (J_2), \pi^{3,2} = (J_1, J_2), \pi^{4,2} = (J_2, J_1) \right\},$$

$$E := \left\{ [\pi^{*,0}, (J_1)], [\pi^{*,0}, (J_2)], [(J_1), (J_1, J_2)], [(J_2), (J_2, J_1)] \right\}.$$

The whole solution tree is constructed similarly until a subset of the complete permutations  $\pi^{k,n} = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in V$  will be obtained. This subset of set  $S$  has to contain at least one permutation of the set  $S(T)$  with the largest relative volume of a stability box.

The rule for cutting a branch in the solution tree  $\mathcal{T} = (V, E)$  is based on Theorem 19, where  $VolSB(\pi^{k,m}, T)$  denotes the relative volume of the stability box  $SB(\pi^{k,m}, T)$ .

**Theorem 19.** *Let for vertices  $\pi^{k,m} \in V$  and  $\pi^{l,m} \in V$ ,  $1 \leq m < n$ , the following conditions hold:*

$$\{J_{k_1}, J_{k_2}, \dots, J_{k_m}\} = \{J_{l_1}, J_{l_2}, \dots, J_{l_m}\}, \tag{12}$$

$$VolSB(\pi^{k,m}, T) \geq VolSB(\pi^{l,m}, T). \tag{13}$$

Then vertex  $\pi^{l,m} \in V$  can be eliminated from further branching in the solution tree  $\mathcal{T} = (V, E)$ .

**Proof.** For each partial permutation  $\pi^{k,m} = (J_{k_1}, J_{k_2}, \dots, J_{k_m}) \in V$  with  $1 \leq m < n$ , one can calculate the stability box  $SB(\pi^{k,m}, T)$  using equality (11) and calculate the value  $VolSB(\pi^{k,m}, T)$  as the product of the relative maximal possible variation of the weight-to-process ratio for all jobs  $J_{k_i}$ ,  $i \in \{1, 2, \dots, m\}$ .

Assigning a job  $J_{k_m}$  to position  $m$  in the permutation  $\pi^{k,m}$  partitions the set of jobs  $\mathcal{J}$  into two subsets with respect to the complete permutation  $\pi_u = (J_{k_1}, J_{k_2}, \dots, J_{k_{m-1}}, J_{k_m}, J_{k_{m+1}} \dots, J_{r_m}) \in S$ . One set is the set of jobs  $\{J_{k_1}, J_{k_2}, \dots, J_{k_{m-1}}\}$  located before job  $J_{k_m}$ , and the other set is the set of jobs  $\{J_{k_{m+1}} \dots, J_{r_m}\}$  located after job  $J_{k_m}$  in the permutation  $\pi_u$ . The maximal possible variation of the weight-to-process ratio for job  $J_{k_m}$  may be calculated using equalities (9) and (10). It is clear that the result of this calculation does not depend on the order of the jobs within the set  $\{J_{k_1}, J_{k_2}, \dots, J_{k_{m-1}}\}$  and within the set  $\{J_{k_{m+1}} \dots, J_{r_m}\}$ .

Thus, if equality (12) and inequality (13) hold for permutations  $\pi^{k,m} \in V$  and  $\pi^{l,m} \in V$ , then permutation  $\pi^{l,m} \in V$  can be eliminated from further branching in the solution tree.  $\square$

Returning to the example, we construct the solution tree presented in Fig. 2, where

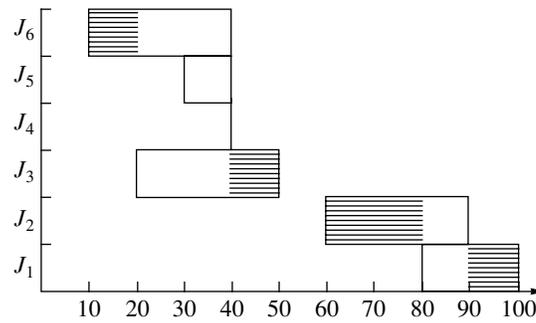
$$\begin{aligned} \pi^{5,3} &= (J_1, J_2, J_3), & \pi^{6,3} &= (J_1, J_2, J_4), & \pi^{7,4} &= (J_1, J_2, J_3, J_4), \\ \pi^{8,4} &= (J_1, J_2, J_4, J_3), & \pi^{9,4} &= (J_1, J_2, J_4, J_5), & \pi^{10,4} &= (J_1, J_2, J_4, J_6), \\ \pi^{11,5} &= (J_1, J_2, J_3, J_4, J_5), & \pi^{12,5} &= (J_1, J_2, J_3, J_4, J_6), & \pi^{13,5} &= (J_1, J_2, J_4, J_5, J_3), \\ \pi^{14,5} &= (J_1, J_2, J_4, J_5, J_6), & \pi^{15,5} &= (J_1, J_2, J_4, J_6, J_3), & \pi^{16,5} &= (J_1, J_2, J_4, J_6, J_5), \\ \pi^{17,6} &= (J_1, J_2, J_3, J_4, J_5, J_6), & \pi^{18,6} &= (J_1, J_2, J_3, J_4, J_6, J_5), & \pi^{19,6} &= (J_1, J_2, J_4, J_5, J_6, J_3). \end{aligned}$$

It is easy to convince that for the vertices  $\pi^{3,2} = (J_1, J_2) \in V$  and  $\pi^{4,2} = (J_2, J_1) \in V$  of the solution tree  $\mathcal{T} = (V, E)$ , condition (13) holds:

$$VolSB(\pi^{3,2}, T) \geq VolSB(\pi^{4,2}, T).$$

Moreover, permutation  $\pi^{3,2}$  and permutation  $\pi^{4,2}$  include the same set  $\{J_1, J_2\}$  of jobs, i.e., condition (12) also holds for the vertices  $\pi^{3,2} \in V$  and  $\pi^{4,2} \in V$ . Thus, due to Theorem 19, it is not necessary to branch vertex  $\pi^{4,2}$  in the solution tree  $\mathcal{T} = (V, E)$ .

Due to a similar reason, there is no need to branch vertices  $\pi^{8,4} = (J_1, J_2, J_4, J_3) \in V$ ,  $\pi^{13,5} = (J_1, J_2, J_4, J_5, J_3) \in V$ ,  $\pi^{15,5} = (J_1, J_2, J_4, J_6, J_3) \in V$ , and  $\pi^{16,5} = (J_1, J_2, J_4, J_6, J_5) \in V$  (see Fig. 2).



**Fig. 3.** Possible variations of the weight-to-process ratios for the permutation  $\pi_1$  are dashed.

Due to these eliminations of the vertices from the set used for branching in the solution tree  $\mathcal{T} = (V, E)$ , only three permutations of set  $S$  are constructed in the solution tree, namely, permutation  $\pi_1 = \pi^{17,6} = (J_1, J_2, J_3, J_4, J_5, J_6) \in S$ , permutation  $\pi_2 = \pi^{18,6} = (J_1, J_2, J_3, J_4, J_6, J_5) \in S$  and permutation  $\pi_3 = \pi^{19,6} = (J_1, J_2, J_4, J_5, J_6, J_3) \in S$ . Due to Theorem 19, it is sufficient to examine only three permutations  $\pi_1$ ,  $\pi_2$  and  $\pi_3$  as candidates that have the largest volume of a stability box.

Let us calculate a stability box  $\mathcal{SB}(\pi_1, T)$  for permutation  $\pi_1 = (J_1, J_2, J_3, J_4, J_5, J_6) \in V$ . The intervals of the weight-to-process ratios for permutation  $\pi_1$  are defined in columns 5 and 6 in Table 1. The values  $d_i^-$  and  $d_i^+$  defined by (9) and (10) are presented in columns 7 and 8 in Table 1. The maximal possible variations of the weight-to-process ratios of jobs in permutation  $\pi_1$  are presented in a coordinate system in Fig. 3. The abscissa axis is used for indicating the weight-to-process ratio and the ordinate axis is used for presenting the jobs of set  $\mathcal{J}$  ordered as in permutation  $\pi_1$ . The possible variations of the weight-to-process ratios of the jobs are dashed. The maximal possible variations of the job processing times in permutation  $\pi_1$  are presented in columns 9 and 10 in Table 1. Thus, the volume of the stability box  $\mathcal{SB}(\pi_1, T)$  of permutation  $\pi_1$  is equal to  $(4\frac{4}{9} - 4) \times (9 - 6\frac{3}{4}) \times (5 - 4) \times (16 - 8) = 13\frac{1}{3}$ . The dimension  $|N_1|$  of the stability box  $\mathcal{SB}(\pi_1, T)$  is equal to 5 since inequality  $d_5^- > d_5^+$  holds only for job  $J_5$  (see columns 7 and 8 in Table 1).

Similarly, one can calculate the volume and the dimension of the stability boxes for permutations  $\pi_2 \in V$  and  $\pi_3 \in V$ . The volume of the stability box  $\mathcal{SB}(\pi_2, T)$  of permutation  $\pi_2$  is equal to  $(4\frac{4}{9} - 4) \times (9 - 6\frac{3}{4}) \times (5 - 4) = 1\frac{2}{3}$ . The dimension  $|N_2|$  of the stability box  $\mathcal{SB}(\pi_2, T)$  is equal to 4. The volume of the stability box  $\mathcal{SB}(\pi_3, T)$  of permutation  $\pi_3$  is equal to  $(4\frac{4}{9} - 4) \times (9 - 6\frac{3}{4}) = 1\frac{2}{3}$ . The dimension  $|N_3|$  of the stability box  $\mathcal{SB}(\pi_3, T)$  is equal to 3.

Thus, the largest volume of the stability box has permutation  $\pi_1 = \pi^{17,6} = (J_1, J_2, J_3, J_4, J_5, J_6)$ .

Next, we present the scheme of the branch-and-bound algorithm, where initially set  $X(h)$  is the set of vertices of digraph  $G = (\mathcal{J}, \mathcal{A})$  without predecessors, and  $V(h)$  is the set of vertices of the solution tree which have to be branched.

### Algorithm MAXSTABOX

- Input:** Segments  $[p_i^L, p_i^U]$ , weights  $w_i$ ,  $J_i \in \mathcal{J}$ .  
**Output:** Permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$  with the largest  $\text{VolSB}(\pi_k, T)$ .  
*Step 1:* Construct the digraph  $G = (\mathcal{J}, \mathcal{A})$ .  
*Step 2:* Define the search tree  $\mathcal{T} := (\{\pi^{*,0}\}, \emptyset)$ .  
*Step 3:* Set  $h = 1$  and  $V(h) = X(h)$ . Include the vertex set  $X(h)$  into the tree  $\mathcal{T}$ .  
*Step 4:* **IF** the rank of the tree  $\mathcal{T}$  is equal to  $n$  **THEN GOTO** step 7 **ELSE**  
*Step 5:* **FOR**  $\pi^{k,m} \in V(h)$  **DO** restore the path  $\mu(\pi^{k,m})$  from the root of tree  $\mathcal{T}$  to vertex  $\pi^{k,m}$ ;

delete all vertices of path  $\mu(\pi^{k,m})$  from digraph  $G = (\mathcal{J}, \mathcal{A})$ ;  
 each vertex of the obtained digraph defines a new vertex in the tree  $T$ ;  
 using (11) calculate the stability box for the partial job permutation.

**END FOR**

*Step 6:* Set  $h := h + 1$  and  $V(h) = \emptyset$ .

Test condition (13) for all leaves of the tree  $\mathcal{T}$  for each pair for which condition (12) holds.

**IF** both conditions of Theorem 19 hold **THEN** delete all such leaves from the tree  $\mathcal{T}$  except one with the largest volume of the stability box and include this vertex into set  $V(h)$ .

**GOTO** step 4.

*Step 7:* Using Algorithm STABOX, select the permutation  $\pi_k$  with the largest  $Vol\mathcal{SB}(\pi_k, T)$   
**STOP.**

The algorithm MAXSTABOX allows us to find a permutation  $\pi_k = (J_{k_1}, J_{k_2}, \dots, J_{k_n}) \in S$  with the largest  $Vol\mathcal{SB}(\pi_k, T)$ . However, there might be several permutations with the largest relative volume of a stability box. In particular, if several adjacent jobs within permutation  $\pi_k$  have no possible variation of their weight-to-process ratio. To order such a set of jobs, we use one of the following two heuristics: A lower-point scenario heuristic, which solves to optimality the deterministic problem  $1|p^L|\sum w_i C_i$  with the processing times  $p^L = (p_1^L, p_2^L, \dots, p_n^L)$ , and an upper-point scenario heuristic, which solves to optimality the deterministic problem  $1|p^U|\sum w_i C_i$  with the processing times  $p^U = (p_1^U, p_2^U, \dots, p_n^U)$ . Combining algorithm MAXSTABOX with the lower-point scenario heuristic is called Algorithm SL, and that combined with the upper-point scenario heuristic is called Algorithm UL.

#### 4. COMPUTATIONAL RESULTS

Table 2 presents some computational results for testing randomly generated instances of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  with  $n \in \{5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100\}$ . We answer (by experiments on a Laptop) the question of how large the relative error  $\Delta$  of the value  $\gamma_{p^*}^k$  of the objective function  $\gamma = \sum_{i=1}^n w_i C_i$  is obtained for the permutation  $\pi_k$  with the largest relative volume of the stability box  $\mathcal{SB}(\pi_k, T)$  with respect to the actually optimal objective function value  $\gamma_{p^*}^t$  calculated for the actual processing times  $p^* = (p_1^*, p_2^*, \dots, p_n^*) \in T$ :

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

Remind that the actual processing times are assumed to be unknown before scheduling. Algorithms SL and UL were coded in C++ and were used to find a permutation  $\pi_k \in S(T)$  with the largest relative volume of the stability box. An integer lower bound  $p_i^L$  and an integer upper bound  $p_i^U$  for the possible real values  $p_i \in R_+^1$  of the job processing times,  $p_i \in [p_i^L, p_i^U]$ , were generated as follows. First, an integer center  $C$  of the closed interval  $[p_i^L, p_i^U]$  was generated using a uniform distribution in the given range  $[L, U]$ :  $L \leq C \leq U$ . Then the lower bound  $p_i^L$  for the possible processing time was defined using the equality

$$p_i^L = C \left( 1 - \frac{\delta}{100} \right).$$

An upper bound  $p_i^U$  was defined using the equality

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

**Table 2.** Randomly generated instances with  $[L, U] = [10, 1000]$  and  $w_i \in [1, 50]$

$n$	$\delta$ (%)	$ \mathcal{A} $ (%)	Average $ N_k $	Average $VolSB(\pi_k, T)$	Exact solutions		Average error		Maximal error		CPU time
					SL	SU	SL	SU	SL	SU	
1	2	3	4	5	6	7	8	9	10	11	12
5	0.1	100	5	1	10	10	0	0	0	0	0.0
5	0.5	100	5	1	10	10	0	0	0	0	0.0
5	1	97.00	4.9	0.741	10	10	0	0	0	0	0.0
5	5	93.00	5	0.509	10	10	0	0	0	0	0.0
5	10	92.00	5	0.441	7	7	0.000931	0.000931	0.005934	0.005934	0.0
5	15	88.00	4.9	0.518	9	9	0.000381	0.000381	0.003811	0.003811	0.0
5	25	82.00	4.7	0.295	6	6	0.004766	0.004766	0.0326	0.0326	0.0
5	50	53.0	3.8	0.0441	5	5	0.010454	0.010454	0.058311	0.058311	0.0
10	0.1	100	10	1	10	10	0	0	0	0	0.0
10	0.5	99.33	9.9	0.757	10	10	0	0	0	0	0.0
10	1	98.89	10	0.677	9	9	0.000003	0.000003	0.000029	0.000029	0.0
10	5	95.78	9.8	0.258	9	9	0.000065	0.000065	0.000651	0.000651	0.0
10	10	93.11	9.7	0.0521	7	7	0.000158	0.000158	0.000961	0.000961	0.0
10	15	87.56	9.0	0.0254	3	3	0.002768	0.002768	0.00882	0.00882	0.0
10	25	71.11	6.5	0.0120	0	0	0.006057	0.006057	0.018855	0.018855	0.0
10	50	52.00	4.0	0.0139	0	0	0.02628	0.02628	0.05833	0.05833	0.0
15	0.1	99.90	15	0.982	10	10	0	0	0	0	0.0
15	0.5	99.62	15	0.806	10	10	0	0	0	0	0.0
15	1	98.48	14.9	0.337	7	7	0.000041	0.000041	0.000203	0.000203	0.0
15	5	96.10	14.5	0.0701	4	4	0.000273	0.000273	0.001192	0.001192	0.0
15	10	88.00	11.7	0.00690	2	2	0.001158	0.001158	0.004991	0.004991	0.0
15	15	84.29	10.5	0.00225	2	2	0.002474	0.002474	0.005859	0.005859	0.0
15	25	78.57	7.8	0.00371	1	1	0.006385	0.006385	0.017376	0.017376	0.0
15	50	54.38	3.8	0.0469	0	0	<b>0.025886</b>	0.026099	0.050611	0.050611	0.0
20	0.1	99.95	20	0.901	10	10	0	0	0	0	0.0
20	0.5	99.63	20	0.553	8	8	0.000003	0.000003	0.000029	0.000029	0.0
20	1	99.05	19.8	0.250	6	6	0.000019	0.000019	0.000112	0.000112	0.0
20	5	94.58	17.7	0.000272	1	1	0.000479	0.000479	0.001646	0.001646	0.0
20	10	88.26	13.6	0.000806	0	0	0.001955	0.001955	0.006086	0.006086	0.0
20	15	87.11	13.2	0.000130	1	1	0.002976	0.002976	0.008585	<b>0.008544</b>	0.0
20	25	70.89	6.8	0.0285	0	0	0.009162	0.009162	0.015119	0.015119	0.0
20	50	47.95	3.5	0.0437	0	0	<b>0.049569</b>	0.04958	<b>0.135189</b>	0.135295	3.3
30	0.1	99.89	29.9	0.699	8	8	0.000002	0.000002	0.000008	0.000008	0.0
30	0.5	99.53	29.3	0.255	6	6	0.000005	0.000005	0.000023	0.000023	0.0
30	1	98.78	29.3	0.00608	5	5	0.000021	0.000021	0.000097	0.000097	0.0
30	5	94.76	24.0	0.000001	0	0	<b>0.000409</b>	0.000412	<b>0.00143</b>	0.001469	0.0
30	10	89.17	16.5	0.00003	0	0	0.001823	0.001823	0.004153	0.004153	0.0
30	15	85.15	14.0	0.000298	0	0	<b>0.00396</b>	0.003976	0.007035	0.007035	0.1
30	25	75.17	6.5	0.0114	0	0	0.0069	<b>0.006895</b>	0.011764	<b>0.011716</b>	0.3
30	50	63.45	3.0	0.00031	0	0	0.041807	0.041807	0.041807	0.041807	1.0
40	0.1	99.92	39.9	0.699	10	10	0	0	0	0	0.0
40	0.5	99.38	39.0	0.0818	5	5	0.000009	0.000009	0.00003	0.00003	0.0
40	1	98.90	38.7	0.00206	1	1	<b>0.000017</b>	0.000019	<b>0.00005</b>	0.000064	0.0
40	5	94.73	28.1	$\approx 0$	0	0	0.000438	<b>0.000427</b>	0.001186	0.001186	0.0
40	10	90.08	19.0	$\approx 0$	0	0	<b>0.001663</b>	0.00173	0.003472	0.003472	0.1
40	15	84.56	13.8	$\approx 0$	0	0	<b>0.004143</b>	0.004315	<b>0.006127</b>	0.006651	0.4
40	25	77.18	4.5	0.01076	0	0	0.009055	0.009055	0.012444	0.012444	1.0

**Table 2.** (Contd.) Randomly generated instances with  $[L, U] = [10, 1000]$  and  $w_i \in [1, 50]$

$n$	$\delta$ (%)	$ \mathcal{A} $ (%)	Average $ N_k $	Average $VolSB(\pi_k, T)$	Exact solutions		Average error		Maximal error		CPU time
					SL	SU	SL	SU	SL	SU	
1	2	3	4	5	6	7	8	9	10	11	12
50	0.1	99.92	49.4	0.612	7	7	0.000001	0.000001	0.000006	0.000006	0.1
50	0.5	99.46	48.9	0.00918	5	5	0.000005	0.000005	0.000031	0.000031	0.0
50	1	98.99	48.0	0.00029	1	1	0.000016	0.000016	0.000046	0.000046	0.0
50	5	94.75	31.8	$\approx 0$	0	0	<b>0.000455</b>	0.00046	<b>0.00073</b>	0.00073	0.2
50	10	89.81	18.6	$\approx 0$	0	0	0.001859	<b>0.001856</b>	0.002762	<b>0.002741</b>	0.6
50	15	85.42	13.7	$\approx 0$	0	0	0.003332	<b>0.00329</b>	0.005219	0.005219	6.0
50	25	76.33	7.0	0.03944	0	0	<b>0.007341</b>	0.007546	0.007953	0.007953	10.0
60	0.1	99.92	59.3	0.583	8	8	$\approx 0$	$\approx 0$	0.000001	0.000001	0.1
60	0.5	99.58	59.0	0.00394	3	3	0.000004	0.000004	0.000023	0.000023	0.1
60	1	98.95	56.3	$\approx 0$	0	0	<b>0.000027</b>	0.000028	0.000086	0.000086	0.1
60	5	94.69	33.7	$\approx 0$	0	0	<b>0.000385</b>	0.000395	0.000978	<b>0.000973</b>	0.4
60	10	89.03	17.6	0.00117	0	0	0.001624	<b>0.001608</b>	0.002507	0.002507	5.5
60	15	84.60	10.9	0.00025	0	0	0.003656	<b>0.003645</b>	0.004973	0.004973	13.2
60	25	78.59	10.0	$\approx 0$	0	0	0.012855	<b>0.012783</b>	0.012856	<b>0.012784</b>	29.0
70	0.1	99.93	69.4	0.307	<b>8</b>	7	$\approx 0$	0.000001	0.000002	0.000002	0.1
70	0.5	99.42	66.4	$\approx 0$	0	0	<b>0.000008</b>	0.000009	0.000018	0.000018	0.2
70	1	98.90	65.1	$\approx 0$	0	0	0.000023	0.000023	0.000053	0.000053	0.3
70	5	95.10	35.7	$\approx 0$	0	0	0.000402	<b>0.000401</b>	0.000652	<b>0.000647</b>	0.9
70	10	89.87	17.4	0.000009	0	0	<b>0.001888</b>	0.001893	0.002908	<b>0.002904</b>	10.7
70	15	84.72	12.0	$\approx 0$	0	0	<b>0.002139</b>	0.002234	<b>0.002139</b>	0.002234	34.8
70	25	76.89	5.0	0.000003	0	0	0.008478	<b>0.008449</b>	0.008479	<b>0.00845</b>	361.0
80	0.1	99.95	79.4	0.467	7	7	$\approx 0$	$\approx 0$	0.000002	0.000002	0.3
80	0.5	99.44	76.4	$\approx 0$	1	1	0.000006	0.000006	0.000017	0.000017	0.4
80	1	98.87	71.2	$\approx 0$	0	0	0.000017	0.000017	0.00003	0.00003	0.5
80	5	94.43	34.1	$\approx 0$	0	0	<b>0.000471</b>	0.000472	<b>0.000756</b>	0.000777	6.7
80	10	89.32	16.0	$\approx 0$	0	0	0.00179	<b>0.001771</b>	<b>0.002297</b>	0.002304	106.5
80	15	85.38	11.0	0.00002	0	0	0.004137	<b>0.004123</b>	0.004137	<b>0.004123</b>	63.9
90	0.1	99.92	88.4	0.245	<b>6</b>	5	$\approx 0$	$\approx 0$	0.000001	0.000001	0.6
90	0.5	99.50	85.4	$\approx 0$	0	0	0.000003	0.000003	0.000007	0.000007	0.7
90	1	98.99	81.6	$\approx 0$	0	0	0.000021	0.000021	0.000038	0.000038	0.9
90	5	94.69	34.2	$\approx 0$	0	0	0.00047	<b>0.000469</b>	0.00067	0.00067	8.2
90	10	87.57	15.0	$\approx 0$	0	0	0.001945	<b>0.001898</b>	0.001945	<b>0.001898</b>	206.0
90	15	82.85	8.0	$\approx 0$	0	0	0.003454	<b>0.003328</b>	0.003454	<b>0.003328</b>	2842.0
100	0.1	99.93	98.1	0.0888	4	<b>5</b>	$\approx 0$	$\approx 0$	0.000002	<b>0.000001</b>	1.0
100	0.5	99.49	94.8	$\approx 0$	0	0	0.000006	0.000006	0.00001	0.00001	1.2
100	1	98.89	85.7	$\approx 0$	0	0	<b>0.000018</b>	0.00002	0.000029	0.000029	1.6
100	5	94.98	37.2	$\approx 0$	0	0	<b>0.000465</b>	0.000471	0.00067	<b>0.000634</b>	19.2
100	10	91.92	21.0	$\approx 0$	0	0	0.001067	<b>0.001063</b>	0.001068	<b>0.001064</b>	46.0

As a result, the maximum possible relative error of the uncertain processing time was equal to  $\delta\%$ . In the experiments, we tested instances of problem  $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$  with the relative errors  $\delta\%$  of the random processing times defined by the values of  $\delta \in \{0.1, 0.5, 1.0, 5.0, 10.0, 15.0, 25.0, 50.0\}$ . The same range  $[L, U]$  for the varying center  $C$  of the closed interval  $[p_i^L, p_i^U]$  was used, namely:  $L = 10$  and  $U = 1000$ . For each job  $J_i \in \mathcal{J}$ , the weight  $w_i \in R_+^1$  was uniformly distributed in the range  $[1, 50]$ . In contrast to the actual processing time  $p_i^*$ , the weight  $w_i$  is known before scheduling.

For the experiments, we used a Laptop with Intel Pentium Dual Core with CPU 1.86 GHz and RAM 2 GB. Table 2 represents the computational results for 85 series of randomly generated instances of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ . Each series contains 10 instances with the same combination of  $n$  and  $\delta$ . The number  $n$  of jobs in an instance is given in column 1. The maximum possible error  $\delta$  of the random processing times (in percentages) is given in column 2. Column 3 represents the average relative number  $|\mathcal{A}|$  of the arcs in the digraph  $G = (\mathcal{J}, \mathcal{A})$  constructed using the condition (3) of Theorem 2 (in percentages of the arc number in a complete circuit-free digraph of order  $n$ ):

$$|\mathcal{A}| : \frac{n(n-1)}{2} \times 100\%.$$

Column 4 represents the dimension  $|N_k|$  of the stability box  $\mathcal{SB}(\pi_k, T)$  of the permutation  $\pi_k$  with the largest relative volume of the stability box. In other words,  $|N_k|$  denotes the number of jobs in permutation  $\pi_k$  with  $d_i^- \leq d_i^+$ , where  $d_i^-$  and  $d_i^+$  are defined by (9) and (10), respectively.

Column 5 represents the average value  $Vol\mathcal{SB}(\pi_k, T)$  of the permutations with the largest relative volume of a stability box. If  $\mathcal{SB}(\pi_k, T) = T$  for all the instances in the series, then column 5 contains number one.

Column 6 and 7 represent the number of instances (from 10 ones in a series) for which a permutation  $\pi_k$  with the largest relative volume of the stability box provides an optimal solution of an instance due to Algorithm SL and Algorithm UL, respectively.

The average (maximum) relative error  $\Delta$  of the objective function value  $\gamma_{p^*}^k$  calculated for permutation  $\pi_k$  constructed by the branch-and-bound Algorithm MAXSTABOX with respect to the optimal objective function value  $\gamma_{p^*}^t$  defined for the actual job processing times is given in columns 8 and 9 for Algorithm SL (in columns 10 and 11 for Algorithm SU, respectively). The CPU-time is given in column 12 for each of the Algorithm SL and Algorithm SU (since there is no difference in their running times).

From the experiments, it follows that condition (4) of Theorem 3 holds for instances with a small relative error  $\delta\%$ ,  $\delta \in \{0.1, 0.5\}$ , of the job processing times (see column 3) and for the series with number 11. For each instance of these series, the permutation  $\pi_k$  with the largest relative volume of a stability box provides an optimal solution (columns 6, 7). If one algorithm outperforms the other one, the corresponding number is presented in Table 2 in bold face.

## 5. CONCLUDING REMARKS

In today's innovative, dynamic and very competitive marketplace, an enterprise needs to use optimal scheduling decisions (a scheduling policy) as much as possible in spite of data uncertainty. A schedule minimizing the worst-case regret (such a schedule may be constructed due to robust scheduling [4, 5]) is generally useful for the worst-case scenario. However, the worst-case scenario may be practically realized rather seldom. Indeed, it is unlikely that all the processing times assume their worst values just for the factual schedule. Consequently, a schedule which is optimal for the worst-case regret criterion may be not competitive for the actually realized scenario being often very far from the worst-case one. Moreover, to find a schedule minimizing the worst-case regret for the total flow time criterion  $\sum C_i$  is an NP-hard problem even for two possible scenarios. It should be noted that a lot of real-world scheduling problems deal with a large number of jobs to be scheduled and the number of possible scenarios may be also large.

A stochastically optimal schedule for the  $E(\sum w_i C_i)$  criterion (see [1]) in the class of non-preemptive static list policies (i.e., a schedule minimizing the expected sum of the weighted completion times provided that the jobs are ordered at time zero according to a chosen priority list) may be constructed by the weighted shortest processing time (WSEPT) rule: Process the jobs

in non-increasing order of the ratio  $\frac{w_i}{E(p_i)}$ , where  $E(p_i)$  denotes the expected value of the random processing time  $p_i$  (see page 232 in [1]). However, a stochastically optimal schedule is factually efficient, if the probability distribution of each random processing time is known before scheduling and, moreover, if a sufficiently large number of scenarios will be realized in a rather closed scheduling environment. Again, a stochastically optimal schedule may appear not competitive for the unique scenario which is factually realized. Furthermore, an enterprise may have not enough chances to compensate its loss caused by using a stochastically optimal schedule which is not optimal for the factually realized scenario. Using stochastically optimal schedules for a sufficiently long time (and for a large number of similar scenarios) may be practically impossible for an enterprise since other competitors may be more productive via achieving better results on the market due to their better scheduling policies.

Using the results of paper [28], we pick out in the scenario set  $T$  a subset of scenarios  $\mathcal{SB}(\pi_k, T)$  for which permutation  $\pi_k$  is definitely optimal. Due to assuming a reasonable restriction on the job set  $\mathcal{J}$ , set  $S(T)$  turns out to be the unique minimal dominant set for an instance of problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$ . Consequently, a dominant set  $S(T)$  being minimal with respect to inclusion (see condition (b) of Definition 1) becomes minimal with respect to cardinality. A restriction on the set  $\mathcal{J}$  providing a singularity of set  $S(T)$  implies the identification of appropriate jobs without a loss of potentially optimal schedules and with decreasing the size  $n = |\mathcal{J}|$  of the original problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  under consideration. A minimal dominant set of permutations  $S(T) \subseteq S$  is uniquely determined for problem  $1|p_i^L \leq p_i \leq p_i^U|\sum w_i C_i$  if we choose only one job among a subset of jobs with the same fixed weight-to-process ratio. In case there are several jobs with the same fixed weight-to-process ratio, we can even decrease the number of jobs for the consideration in the minimal dominant set. Thus, the condition for the uniqueness of  $S(T)$  is not restrictive and even useful. We used the notion of a stability box of permutation  $\pi_k \in S$ , which is similar to a stability ball [12–14, 20]. The stability box plays a similar role for the uncertain optimization problem as the stability ball [13, 14, 19–21, 29–31] plays for a post-optimality analysis when the input data and the optimal solution for them are already known and one has to know the credibility of the solution at hand with respect to the possible variation of the input data within a maximal ball. We used an exact formula proven in [28] for characterizing the stability box of any fixed permutation  $\pi_k \in S$  which runs in  $O(n \log n)$  time. We developed a tree-like algorithm for finding a permutation with the largest relative volume of a stability box and we presented computational results for two combinations of a stability heuristic with a lower-point scenario heuristic and an upper-point scenario heuristic. Further research on using a stability box for other uncertain scheduling problems seems to be promising.

## REFERENCES

1. Pinedo, M., *Scheduling: Theory, Algorithms, and Systems*, New Jersey: Prentice Hall, 2002.
2. Aytug, H., Lawley, M.A., McKay, K., Mohan, S., and Uzsoy, R., Executing Production Schedules in the Face of Uncertainties: A Review and Some Future Directions, *Eur. J. Oper. Res.*, 2005, vol. 161, pp. 86–110.
3. Sabuncuoglu, I. and Goren, S., Hedging Production Schedules Against Uncertainty in Manufacturing Environment with a Review of Robustness and Stability Research, *Int. J. Comput. Integrated Manufacturing*, 2009, vol. 22, no. 2, pp. 138–157.
4. Daniels, R.L. and Kouvelis, P., Robust Scheduling to Hedge Against Processing Time Uncertainty in Single-Stage Production, *Manage. Sci.*, 1995, vol. 41, no. 2, pp. 363–376.
5. Kouvelis, P. and Yu, G., *Robust Discrete Optimization and Its Applications*, Boston: Kluwer, 1997.
6. Yang, J. and Yu, G., On the Robust Single Machine Scheduling Problem, *J. Combinat. Optim.*, 2002, vol. 6, pp. 17–33.

7. Lai, T.-C., Sotskov, Y.N., Sotskova, N., and Werner, F., Optimal Makespan Scheduling with Given Bounds of Processing Times, *Math. Comput. Model.*, 1997, vol. 26, pp. 67–86.
8. Lai, T.-C. and Sotskov, Y.N., Sequencing with Uncertain Numerical Data for Makespan Minimization, *J. Oper. Res. Soc.*, 1999, vol. 50, pp. 230–243.
9. Sotskov, Y.N. and Sotskova, N.Y., *Teoriya raspisaniy: sistemy s neopredelennymi chislovyimi parametrami* (Scheduling Theory: Systems with Uncertain Numerical Parameters), Minsk: Natl. Acad. Sci. Belarus, United Inst. of Informatics Problems, 2004.
10. Matsveichuk, N.M., Sotskov, Y.N., Egorova, N.G., and Lai, T.-C., Schedule Execution for Two-Machine Flow-Shop with Interval Processing Times, *Math. Comput. Modelling*, 2009, vol. 49, nos. 5–6, pp. 991–1011.
11. Sotskov, Y.N., Egorova, N.G., and Lai, T.-C., Minimizing Total Weighted Flow Time of a Set of Jobs with Interval Processing Times, *Math. Comput. Modelling*, 2009, vol. 50, pp. 556–573.
12. Lai, T.-C., Sotskov, Y.N., Sotskova, N.Y., and Werner, F., Mean Flow Time Minimization with Given Bounds of Processing Times, *Eur. J. Oper. Res.*, 2004, vol. 159, no. 3, pp. 558–573.
13. Sotskov, Y.N., Wagelmans, A.P.M., and Werner, F., On the Calculation of the Stability Radius of an Optimal or an Approximate Schedule, *Ann. Oper. Res.*, 1998, vol. 83, pp. 213–252.
14. Sotskov, Y.N., Sotskova, N.Y., and Werner, F., Stability of an Optimal Schedule in a Job Shop, *Omega*, 1997, vol. 25, no. 4, pp. 397–414.
15. Graham, R.L., Lawler, E.L., Lenstra, J.K., and Rinnooy Kan, A.H.G., Optimization and Approximation in Deterministic Sequencing and Scheduling. A Survey, *Ann. Discr. Math.*, 1976, vol. 5, pp. 287–326.
16. Smith, W.E., Various Optimizers for Single-Stage Production, *Nav. Res. Logist. Quarterly*, 1956, vol. 3, no. 1, pp. 59–66.
17. Kasperski, A. and Zielinski, P., A 2-approximation Algorithm for Interval Data Minmax Regret Sequencing Problems with Total Flow Time Criterion, *Oper. Res. Lett.*, 2008, vol. 36, pp. 343–344.
18. Montemanni, R., A Mixed Integer Programming Formulation for the Total Flow Time Single Machine Robust Scheduling Problem with Interval Data, *J. Math. Model. Algorith.*, 2007, vol. 6, pp. 287–296.
19. Sotskov, Y.N., Dolgui, A., and Portmann, M.-C., Stability Analysis of Optimal Balance for Assembly Line with Fixed Cycle Time, *Eur. J. Oper. Res.*, 2006, vol. 168, no. 3, pp. 783–797.
20. Sotskov, Y.N., Stability of an Optimal Schedule, *Eur. J. Oper. Res.*, 1991, vol. 55, pp. 91–102.
21. Sotskov, Y.N., Tanaev, V.S., and Werner, F., Stability Radius of an Optimal Schedule: A Survey and Recent Developments, in *Industr. Appl. Combinat. Optim.*, Yu, G., Ed., Boston: Kluwer, 1998, pp. 72–108.
22. Sotskova, N.Y. and Tanaev, V.S., About the Realization of an Optimal Schedule with Operation Processing Times under Conditions of Uncertainty, *Dokl. Natl. Akad. Nauk Belarusi*, 1998, vol. 42, no. 5, pp. 8–12.
23. Allahverdi, A. and Sotskov, Y.N., Two-machine Flowshop Minimum-length Scheduling Problem with Random and Bounded Processing Times, *Int. Transactions Oper. Res.*, 2003, vol. 10, pp. 65–76.
24. Allahverdi, A., Aldowaisan, T., and Sotskov, Y.N., Two-machine Flowshop Scheduling Problem to Minimize Makespan or Total Completion Time with Random and Bounded Setup Times, *Int. J. Math. Math. Sci.*, 2003, vol. 39, pp. 2475–2486.
25. Ng, C.T., Matsveichuk, N.M., Sotskov, Y.N., and Cheng, T.C.E., Two-machine Flow-shop Minimum-length Scheduling with Interval Processing Times, *Asia-Pacific J. Oper. Res.*, 2009, vol. 26, no. 6, pp. 1–20.
26. Sotskov, Y.N., Allahverdi, A., and Lai, T.-C., Flowshop Scheduling Problem to Minimize Total Completion Time with Random and Bounded Processing Times, *J. Oper. Res. Soc.*, 2004, vol. 55, pp. 277–286.

27. *Computer and Job-Shop Scheduling Theory*, Coffman, E.G., Ed., New York: Wiley, 1976.
28. Sotskov, Y.N. and Lai, T.-C., Minimizing Total Weighted Completion Time under Uncertainty Using Stability Box and Dominance, *Comput. Oper. Res.* (submitted).
29. Emelichev, V.A., Girlich, E.N., Nikulin, Y.V., and Podkopaev, D.P., Stability and Regularization Radius of Vector Problems of Integer Linear Programming. *Optimization*, 2002, vol. 51, no. 4, pp. 645–676.
30. Emelichev, V.A., Krichko, V.N., and Nikulin, Y.V., The Stability Radius of an Efficient Solution in Mimimax Boolean Programming Problem. *Control Cybernet.*, 2004, vol. 33, no. 1, pp. 127–132.
31. Emelichev, V.A., Kuz'min, K.G., and Leonovich, A.M., Stability in the Combinatorial Vector Optimization Problems. *Autom. Remote Control*, 2004, vol. 65, no. 2, pp. 227–240.

*This paper was recommended for publication by A.A. Lazarev, a member of the Editorial Board*