A Stability Approach for Minimizing Total Weighted Completion Time with Uncertain Data [1]

Yuri N. Sotskov, Natalja G. Egorova,

United Institute of Informatics Problems, National Academy of Sciences of Belarus, Minsk, Belarus

Frank Werner

Faculty of Mathematics, Otto-von-Guericke-University, Magdeburg, Germany.

January 7, 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 job 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 $5 \leq n \leq 100$.

Keywords: Scheduling, Single machine problems, Total weighted completion time, Stability, Uncertainty, Interval processing times

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 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.

[1] The research of the first author was supported by Belorussian Republican Foundation for Fundamental Research.

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

A set of n jobs $\mathcal{J} = \{J_1, J_2, ..., J_n\}$, $n \geq 2$, 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 $p = (p_1, p_2, \ldots, p_n)$ 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 $[p_i^L, p_i^U]$, $i \in \{1, 2, \ldots, n\}$:

$$T = \{p \mid p \in R_+^n,\ p_i^L \leq p_i \leq p_i^U,\ i \in \{1, 2, \ldots, n\}\} = \times_{i=1}^{n} [p_i^L, p_i^U]. \quad (1)$$

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

Let $S = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ be the set of all permutations $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n})$ of the jobs $\mathcal{J} = \{J_{k_1}, J_{k_2}, \ldots, J_{k_n}\}$. If both the permutation π_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 π_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 $\pi_t = (J_{t_1}, J_{t_2}, \ldots, J_{t_n}) \in S$ 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, $i \in \{1, 2, \ldots, n\}$), 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 $\gamma = f(C_1, C_2, \ldots, C_n)$ 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:

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

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

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

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 π_k with the objective function value equal to γ_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. A 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] and [6], respectively). Only a few very special cases are known to be polynomially solvable for minimizing the worst-case regret for the problems $\alpha|p_i^L \leq p_i \leq p_i^U|\gamma$.

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, 20], a two-stage scheduling decision framework (the off-line planning stage and the on-line scheduling stage) [9, 10], 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, 21, 22] for the makespan criterion, and in [9, 12, 22, 23] for the total completion time criterion. Article [23] addresses the total completion time in a two-machine flow-shop problem $F2|p_i^L \le p_i \le p_i^U|\sum C_i$. A geometrical algorithm has been developed for solving the flow-shop problem $Fm|p_i^L \le p_i \le p_i^U, n = 2|\sum C_i$ 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 [22] 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 [21], for a two-machine flow-shop problem $F2|p_i^L \le p_i \le p_i^U|C_{max}$ sufficient conditions were identified when a transposition of two jobs minimizes C_{max}. In [12], for a job-shop problem $Jm|p_i^L \le p_i \le p_i^U|\sum C_i$ 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 $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$, we remind some known results for the uncertain scheduling problem and for its deterministic counterpart.

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

$$\frac{w_{k_1}}{p_{k_1}} \ge \frac{w_{k_2}}{p_{k_2}} \ge \ldots \ge \frac{w_{k_n}}{p_{k_n}}, \tag{2}$$

where inequality $p_{k_i} > 0$ holds for each job $J_{k_i} \in \mathcal{J}$. Thus, problem $1||\sum w_iC_i$ 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 [24].

Theorem 1 [16, 24] *Permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ is optimal for the deterministic problem $1||\sum w_iC_i$ if and only if inequalities* (2) *hold.*

A minimal dominant set $S(T)$ for problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ 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 $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ 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 $1||\sum w_iC_i$ associated with a scenario $p \in T$ is a singleton, $S(T) = \{\pi_k\}$, where $T = \{p\}$. Hence, relations $J_{k_1} \mapsto J_{k_2} \mapsto J_{k_3} \mapsto \ldots \mapsto J_{k_{n-1}} \mapsto J_{k_n}$ with respect to $T = \{p\}$ hold for the deterministic problem $1||\sum w_iC_i$. The following claim has been proven in [11].

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

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

The cardinality $|S(T)|$ of a minimal dominant set $S(T)$ may be considered as a measure of uncertainty for problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$. In the least uncertain case, a minimal dominant set for problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ is a singleton, $\{\pi_k\} = S(T)$, which is also a solution to the deterministic counterpart $1||\sum w_iC_i$ 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* $S(T) = \{\pi_k\} = \{(J_{k_1}, J_{k_2}, \ldots, J_{k_n})\}$ *for problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$*, inequalities (4) are necessary and sufficient:*

$$\frac{w_{k_1}}{p_{k_1}^U} \ge \frac{w_{k_2}}{p_{k_2}^L}; \quad \frac{w_{k_2}}{p_{k_2}^U} \ge \frac{w_{k_3}}{p_{k_3}^L}; \quad \ldots; \quad \frac{w_{k_{n-1}}}{p_{k_{n-1}}^U} \ge \frac{w_{k_n}}{p_{k_n}^L}. \tag{4}$$

The most uncertain case of problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ 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* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ *with a maximum cardinality* $|S(T)| = n!$*, inequality (5) is necessary and sufficient:*

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

In [25], the uniqueness of a minimal dominant set $S(T)$ was investigated. Let notation $1|p|\sum w_iC_i$ be used for indicating an individual problem (an instance) of the mass problem $1||\sum w_iC_i$ 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 $\mathcal{J}$:

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

Theorem 5 [25] *For problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$*, a minimal dominant set* $S(T)$ *is uniquely determined if and only if there is no real* $r \in [a, b]$ *such that* $|\mathcal{J}_r| \ge 2$*.*

Theorem 6 [25] *Let* $S(T)$ *be a minimal dominant set for problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$*. For any permutation* $\pi_k \in S(T)$*, there exists a scenario* $p \in T$ *such that* π_k *is the unique optimal permutation for the instance* $1|p|\sum w_iC_i$ *if and only if there is no* $r \in [a, b]$ *such that* $|\mathcal{J}_r| \ge 2$*.*

Theorem 4 is generalized in [25] as follows.

Theorem 7 [25] *Let there exist no real* $r \in [a, b]$ *such that* $|\mathcal{J}_r| \ge 2$*. For the existence of a minimal dominant set* $S(T)$ *for problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ *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 $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ 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 [25] *Digraph* $(\mathcal{J}, \mathcal{A})$ *constructed for problem* $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ *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* $|\mathcal{J}_r| \ge 2$*.*

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 [25] as follows.

Theorem 9 [25] *If inequality $|\mathcal{J}_{r_q}| \geq 2$ holds for each real $r_q \in \{r_1, r_2, \ldots, r_m\}$, 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 $|\mathcal{J}_{r_q}| \geq 2$, $r_q \in \{r_1, r_2, \ldots, r_m\}$, 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: $\pi_l = (\ldots, \pi(\mathcal{J}_{r_q}), \ldots)$, where $\pi(\mathcal{J}_{r_q})$ is a permutation of the jobs $\mathcal{J}_{r_q}$. Moreover, the order of the jobs $\{J_{q(1)}, J_{q(2)}, \ldots, J_{q(|\mathcal{J}_{r_q}|)}\} = \mathcal{J}_{r_q}$ 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 $\pi_k = (\ldots, \pi(\mathcal{J}_{r_q}), \ldots)$ (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 $\{J_{q(1)}, J_{q(2)}, \ldots, J_{q(|\mathcal{J}_{r_q}|)}\} = \mathcal{J}_{r_q}$ 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 $\mathcal{J}_{r_q}$, $r_q \in \{r_1, r_2, \ldots, r_m\}$, $|\mathcal{J}_{r_q}| \geq 2$, 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 [25] *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 π_k of the instance $1^*|p| \sum w_i C_i$ is obtained from the corresponding optimal permutation π_t of the instance $1|p| \sum w_i C_i$ of the original uncertain problem via deleting from permutation π_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 π_k is the unique optimal permutation for the instance $1|p|\sum w_iC_i$.

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 [25], 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 $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ with respect to T, if for each $i \in \{1, 2, \ldots, n\}$, permutation π_k remains optimal for the instance $1|p'|\sum w_iC_i$ with any scenario $p' = (p'_1, p'_2, \ldots, p'_n) \in \{\times_{j=1, j\neq i}^{n}[p_{k_j}^L, p_{k_j}^U]\} \times [l_{k_i}, u_{k_i}]$. 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 $\mathcal{SB}(\pi_k, T) = \emptyset$.*

In Definition 3, the maximality of a rectangular box $\mathcal{SB}(\pi_k, T)$ means that for each position $i \in \{1, 2, \ldots, n\}$ in permutation π_k, the lower bound l_{k_i} (the upper bound u_{k_i}) of the variation of the processing time p_{k_i} of job J_{k_i}, which is located at position i in permutation π_k, preserving the optimality of permutation π_k has to be as small (as large) as possible provided that the processing time of each other job J_{k_j}, $j \in \{1, 2, \ldots, n\} \setminus \{i\}$, 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 π_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 π_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] and [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 $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$, the stability region $\mathcal{K}(\pi_k, T)$ of a permutation $\pi_k \in S$ with respect to T is defined as follows:

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

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 $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$, 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 [25], the following properties of the stability box and the stability region have been derived.

Theorem 11 [25] *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 π_k is optimal for the instance $1|p|\sum w_iC_i$.*

Theorem 12 [25] *There exists a scenario $p \in T$ such that permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ is optimal for the instance $1|p|\sum w_iC_i$ if and only if there is no job J_{k_i}, $i \in \{1, 2, \ldots, n-1\}$, such that inequality*

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

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

Theorem 13 [25] *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}, $i \in \{1, 2, \ldots, n-1\}$, such that inequality (8) holds for at least one job J_{k_j}, where $j \in \{i+1, i+2, \ldots, n\}$.*

Definitions 3 and (7) imply the following claim.

Theorem 14 [25] *If there exists exactly one scenario $p \in T$ such that permutation $\pi_k \in S$ is optimal for the instance $1|p|\sum w_iC_i$, 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 [25] *$\mathcal{SB}(\pi_k, T) = T = \mathcal{K}(\pi_k, T)$ 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 $1|p^L_i \le p_i \le p^U_i|\sum w_iC_i$ 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 [25] *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 $\pi_k = (J_{k_1}, \ldots, J_{k_{i-1}}, J_{k_i}, J_{k_{i+1}}, \ldots, J_{k_n}) \in S$. In Section 3, we show how to use this algorithm for solving approximately problem $1|p^L_i \le p_i \le p^U_i|\sum w_iC_i$.

2.4 Polynomial algorithm for calculating the stability box

Due to the additivity of the objective function $\gamma = \sum w_iC_i$, 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}, $i \in \{1, 2, \ldots, n\}$, which preserves the optimality of permutation π_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 π_k is an optimal permutation for the instance $1|p|\sum w_iC_i$ with $p = (p_1, p_2, \ldots, p_n) \in T$, then permutation π_k remains optimal for any instance $1|p'|\sum w_iC_i$ with $p' = (p'_1, p'_2, \ldots, p'_n) \in T$, 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:

$$d^-_{k_i} = \max\left\{\frac{w_{k_i}}{p^U_{k_i}},\ \max_{i<j\le n}\left\{\frac{w_{k_j}}{p^L_{k_j}}\right\}\right\}. \tag{9}$$

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

$$d^+_{k_i} = \min\left\{\frac{w_{k_i}}{p^L_{k_i}},\ \min_{1\le j<i}\left\{\frac{w_{k_j}}{p^U_{k_j}}\right\}\right\}. \tag{10}$$

In [25], the following claims have been proven.

Theorem 17 [25] *If there is no job J_{k_i}, $i \in \{1, 2, \ldots, n-1\}$, in permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ such that inequality (8) holds for at least one job J_{k_j}, where $j \in \{i+1, i+2, \ldots, n\}$, then*

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

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

Corollary 1 [25] *If $\mathcal{SB}(\pi_k, T) \neq \emptyset$ for problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ with the scenario set T, then the singleton $\{\pi_k\}$ is a minimal dominant set for problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ with the scenario set $T = \mathcal{SB}(\pi_k, T)$.*

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

Algorithm STABOX [25]

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

Step 1: Construct the list L of fractions $\frac{w_{k_i}}{p_{k_i}^L}$, $i = 1, 2, \ldots, n$, 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 $\frac{w_{k_i}}{p_{k_i}^U}$, $i = 1, 2, \ldots, n$, 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 $\frac{w_{k_i}}{p_{k_i}^U}$, $i = 1, 2, \ldots, n$, 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 $d_{k_i}^- = \max\left\{\frac{w_{k_i}}{p_{k_i}^U},\ L_1\right\}$,
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 $\mathcal{SB}(\pi_k, T) := \times_{d_i^- \le d_i^+} \left[\frac{w_{k_i}}{d_{k_i}^+}, \frac{w_{k_i}}{d_{k_i}^-}\right] \times \{\times_{d_j^- > d_j^+} [p_{k_j}, p_{k_j}]\}$ **STOP.**

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	8	200	50	25	40	50	4	5
4	6	6	240	40	40	40	25	-	-
5	3	4	120	40	30	40	25	-	-
6	4	32	160	40	5	20	25	$6\frac{2}{5}$	8

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 [25] *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:

$$\frac{w_1}{p_1^U} = 80 \geq 50 = \frac{w_3}{p_3^L}; \quad \frac{w_1}{p_1^U} = 80 \geq 40 = \frac{w_4}{p_4^L}; \quad \frac{w_2}{p_2^U} = 60 \geq 50 = \frac{w_3}{p_3^L};$$

$$\frac{w_2}{p_2^U} = 60 \geq 40 = \frac{w_4}{p_4^L}; \quad \frac{w_4}{p_4^U} = 40 \geq 40 = \frac{w_5}{p_5^L}; \quad \frac{w_4}{p_4^U} = 40 \geq 40 = \frac{w_6}{p_6^L}.$$

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 $G = (\mathcal{J}, \mathcal{A})$ (the reduction of this digraph is represented in Figure 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.

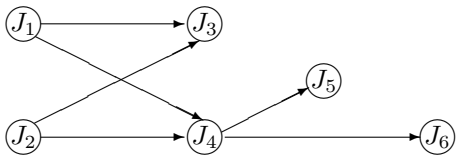

Figure 1: The reduction of digraph $G = (\mathcal{J}, \mathcal{A})$ defining a unique minimal dominant set $S(T)$.

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 Figure 2. Each vertex of the solution tree $\mathcal{T} = (V, E)$ is a permutation $\pi^{k,m} = (J_{k_1}, J_{k_1}, \ldots, J_{k_m}) \in V$, $m \leq n$, of some jobs from set $\mathcal{J}$. Edge $[\pi^{k,m}, \pi^{l,m+1}]$ belongs to set E, if permutation $\pi^{l,m+1} = (J_{l_1}, J_{l_1}, \ldots, J_{l_{m+1}})$, $m \leq n-1$, was obtained from permutation $\pi^{k,m} = (J_{k_1}, J_{k_1}, \ldots, J_{k_m})$, i.e., equality $J_{k_i} = J_{l_i}$ holds for each index $i \in \{1, 2, \ldots, m\}$.

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 $X(h)$, 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 Figure 1).

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

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

This means that either job J_1 or job J_2 may be located at the first position in the desired job permutation.

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 $X(h-1) = X(1)$.

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

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

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

The whole solution tree is constructed similarly until a subset of the complete job permutations $\pi^{k,n} = (J_{k_1}, J_{k_1}, \ldots, J_{k_n}) \in V$ 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 the stability box.

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

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

$$\{J_{k_1}, J_{k_1}, \ldots, J_{k_m}\} = \{J_{l_1}, J_{l_1}, \ldots, J_{l_m}\}, \tag{12}$$

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

Then vertex $\pi^{l,m} \in V$ can be eliminated from further branching in the solution tree.

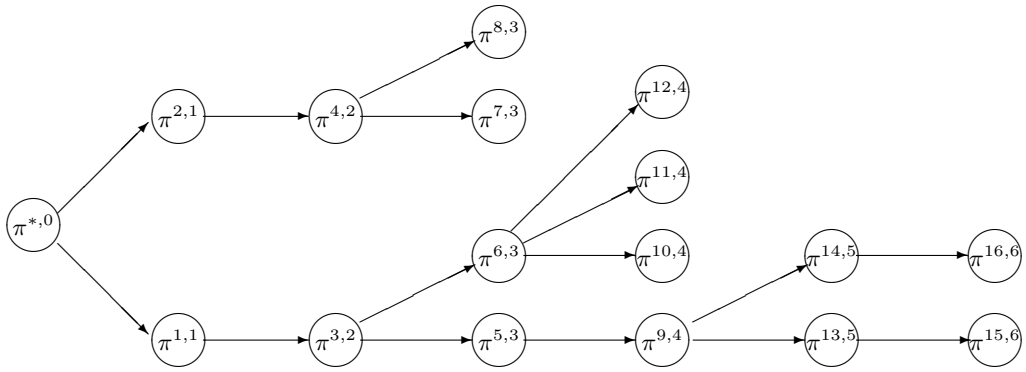

Figure 2: Solution tree $\mathcal{T} = (V, E)$ constructed for the example.

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

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 $\pi_u = (J_{k_1}, J_{k_1}, \ldots, J_{k_{m-1}}, J_{k_m}, J_{k_{m+1}} \ldots, J_{r_m}) \in S$. One set is the set of jobs $\{J_{k_1}, J_{k_1}, \ldots, J_{k_{m-1}}\}$ located before job J_{k_m}, and the other set is the set of jobs $\{J_{k_{m+1}} \ldots, J_{r_m}\}$ located after job J_{k_m} in the permutation π_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 $\{J_{k_1}, J_{k_1}, \ldots, J_{k_{m-1}}\}$ and within the set $\{J_{k_{m+1}} \ldots, J_{r_m}\}$.

Thus, if inequality (13) and equality (12) 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. ■

Returning to the example, we can construct the solution tree presented in Figure 2, where $\pi^{5,3} = \{J_1, J_2, J_3\}$, $\pi^{6,3} = \{J_1, J_2, J_4\}$, $\pi^{7,3} = \{J_2, J_1, J_3\}$, $\pi^{8,3} = \{J_2, J_1, J_4\}$, $\pi^{9,4} = \{J_1, J_2, J_3, J_4\}$, $\pi^{10,4} = \{J_1, J_2, J_4, J_3\}$, $\pi^{11,4} = \{J_1, J_2, J_4, J_5\}$, $\pi^{12,4} = \{J_1, J_2, J_4, J_6\}$, $\pi^{13,5} = \{J_1, J_2, J_3, J_4, J_5\}$, $\pi^{14,5} = \{J_1, J_2, J_3, J_4, J_6\}$, $\pi^{15,6} = \{J_1, J_2, J_3, J_4, J_5, J_6\}$, $\pi^{16,6} = \{J_1, J_2, J_3, J_4, J_6, J_5\}$.

It is easy to convince that for the vertices $\pi^{5,3} = \{J_1, J_2, J_3\}$ and $\pi^{7,3} = \{J_2, J_1, J_3\}$ of the solution tree, condition (13) holds:

$$Vol\mathcal{SB}(\pi^{5,3}, T) \geq Vol\mathcal{SB}(\pi^{7,3}, T),$$

Moreover, permutation $\pi^{5,3}$ and permutation $\pi^{7,3}$ include the same set of jobs $\{J_1, J_2, J_3\}$, i.e., condition (12) also holds. Due to Theorem (12), it is not necessary to branch vertex $\pi^{7,3}$. Due to a similar reason, there is no need to branch vertices $\pi^{8,3} = \{J_2, J_1, J_4\}$ and $\pi^{10,4} = \{J_1, J_2, J_4, J_3\}$ (see Figure 2).

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 $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ with the largest relative volume of a stability box $\mathcal{SB}(\pi_k, T)$.
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 less than 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;
set $h := h + 1$.
END FOR
Step 6: Set $V(h) = \emptyset$.
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 with
the largest relative volume of the stability box **STOP.**

The algorithm MAXSTABOX allows us to find a permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ with the largest relative volume of a stability box $\mathcal{SB}(\pi_k, T)$. However, there might be several permutations with the largest relative volume of a stability box $\mathcal{SB}(\pi_k, T)$. In particular, if several adjacent jobs within permutation π_k have a zero *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 $1|p^L|\sum w_iC_i$ with the processing times $p^L = (p_1^L, p_2^L, \ldots, p_n^L)$, and

an upper-point scenario heuristic, which solves to optimality the deterministic problem $1|p^U|\sum w_iC_i$ with the processing times $p^U = (p_1^U, p_2^U, \ldots, p_n^U)$.

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 $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ 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 Δ of the value $\gamma_{p^*}^k$ of the objective function $\gamma = \sum_{i=1}^n w_iC_i$ is, obtained for the permutation π_k with the largest relative volume of the stability box $\mathcal{SB}(\pi_1, T)$ with respect to the actually optimal objective function value $\gamma_{p^*}^t$ calculated for the actual processing times $p^* = (p_1^*, p_2^*, \ldots, p_n^*) \in T$:

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

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 $[L, U]$: $L \le C \le U$. Then the lower bound p_i^L for the possible processing time was defined using the equality

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

An upper bound p_i^U was defined using the equality

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

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 $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ 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 1 represents the computational results for 85 series of randomly generated instances of problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$. Each series contains 10 instances with the same combination of n and δ. The number n of jobs in an instance is given in column 1. The maximum possible error δ 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):

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

Column 4 represents the average number $|N_k|$ of the processing times which may vary in the stability box $\mathcal{SB}(\pi_k, T)$ of the permutation π_k with the largest relative volume of the stability box. In other words, $|N_k|$ denotes the number of jobs with a non-zero relative maximal *possible variation* of the weight-to-process. Column 5 represents the average relative volume $Vol\mathcal{SB}(\pi_k, T)$ of the permutations with the largest $Vol\mathcal{SB}(\pi^{l,m}, T)$. If $\mathcal{SB}(\pi^{l,m}, T) = T$ for all the instances in the series, then columnn 5 contains number one.

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

The average (maximum) relative error Δ of the objective function value $\gamma_{p^*}^k$ calculated for permutation π_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 algorithms SL and SU (since there is no difference in their running times).

From the experiments, it follows that condition (4) of Theorem 3 holds only for instances with a small relative error $\delta\%$, $\delta \in \{0.1,\ 0.5\}$, of the job processing times (see column 3 for the series with numbers 1–10) and for the series with number 11. For each instance of these series, the permutation π_k with the largest relative volume of a stability box provides an optimal solution (columns 6, 7 and 8). If one algorithm outperforms the other one, the corresponding

number is presented in Table 2 in bold face. The permutation π_k with the largest relative volume of a stability box provides an optimal solution for each instance of a series for which there is no single dominant permutation for at least one instance.

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 very 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 no 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 [25], we pick out in the scenario set T a subset of scenarios $\mathcal{SB}(\pi_k, T)$ for which permutation π_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 $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$. 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 $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$ under consideration. A minimal dominant set of permutations $S(T) \subseteq S$ is uniquely determined for problem $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$ 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 a 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, 20, 26, 27, 28] 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 to the possible variation of the input data within a maximal ball.

We used an exact formula for characterizing the stability box of any fixed permutation $\pi_k \in S$ which runs in $O(n \log n)$ time without using a special data structure. 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*. 2nd ed. USA, 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, *European Journal of Operational Research*, 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, *International Journal of Computer 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, *Management Science*, 1995, vol. 41, no. 2, pp. 363–376.

[5] Kouvelis, P. and Yu, G., *Robust discrete optimization and its applications*, USA, Boston: Kluwer Academic Publishers, 1997.

[6] Yang, J. and Yu, G., On the robust single machine scheduling problem, *Journal of Combinatorial Optimization*, 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, *Mathematical and Computer Modelling*, 1997, vol. 26, pp. 67–86.

[8] Lai, T.-C. and Sotskov, Y.N., Sequencing with uncertain numerical data for makespan minimization, *Journal of the Operational Research Society*, 1999, vol. 50, pp. 230–243.

[9] Sotskov, Y.N. and Sotskova, N.Y., *Teoriya raspisanii: sistemi s neopredelennimi chislovimi parametrami* (Scheduling theory: Systems with uncertain numerical parameters), Belarus, Minsk: National Academy of Sciences of Belarus. United Institute of Informatics Problems, 2004 (in Russian).

[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, *Mathematical and Computer Modelling*, 2009, vol. 49, no. 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, *Mathematical and Computer Modelling*, 2009, vol. 50, pp. 556–573.

[12] Lai, T-C, Sotskov, Y.N., Sotskova, N., and Werner, F., Mean flow time minimization with given bounds of processing times, *European Journal of Operational Research*, 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, *Annals of Operations Research*, 1998, vol. 83, pp. 213–252.

[14] Sotskov, Y.N., Sotskova, N., 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, *Annals of Discrete Mathematics*, 1976, vol. 5, pp. 287–326.

[16] Smith, W.E., Various optimizers for single-stage production, *Naval Research and Logistics 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, *Operations Research Letters*, 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, *Journal of Mathematical Models and Algorithms*, 2007, vol. 6, pp. 287–296.

[19] Sotskov, Y.N., Dolgui, A., Portmann, M.-C., Stability analysis of optimal balance for assembly line with fixed cycle time, *European Journal of Operational Research*, 2006, vol. 168, no. 3, pp. 783–797.

[20] Sotskov, Y.N., Stability of an optimal schedule, *European Journal of Operational Research*, 1991, vol. 55, pp. 91–102.

[21] Allahverdi, A. and Sotskov, Y.N., Two-machine flowshop minimum-length scheduling problem with random and bounded processing times, *International Transactions in Operational Research*, 2003, vol. 10, pp. 65–76.

[22] 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, *International Journal of Mathematics and Mathematical Sciences*, 2003, vol. 39, pp. 2475–2486.

[23] Sotskov, Y.N., Allahverdi, A., and Lai, T.-C., Flowshop scheduling problem to minimize total completion time with random and bounded processing times, *Journal of the Operational Research Society*, 2004, vol. 55, pp. 277–286.

[24] Coffman, E.G. (Editor), *Computer and Job-Shop Scheduling Theory*, USA: John Wiley & Sons, 1976.

[25] Sotskov, Y.N. and Lai, T.-C., Minimizing total weighted completion time under uncertainty using stability box and dominance, *Computers & Operations Research*, (submitted).

[26] 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.

[27] Emelichev, V.A., Krichko, V.N., and Nikulin, Y.V., The stability radius of an efficient solution in mimimax Boolean programming problem. *Control and Cybernetics*, 2004, vol. 33, no. 1, pp. 127–132.

[28] Emelichev, V.A., Kuz'min, K.G., and Leonovich, A.M., Stability in the combinatorial vector optimization problems. *Automation and Remote Control*, 2004, vol. 65, no. 2, pp. 227–240.

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

n	δ	$\lvert\mathcal{A}\rvert$	Average	Relative	Exact solutions		Average error		Maximal error		CPU
	(%)	(%)	$\lvert N_k\rvert$	V_k	SL	SU	SL	SU	SL	SU	time
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	**0.025886**	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	**0.008544**	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	**0.049569**	0.04958	**0.135189**	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	**0.000409**	0.000412	**0.00143**	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	**0.00396**	0.003976	0.007035	0.007035	0.1
30	25	75.17	6.5	0.0114	0	0	0.0069	**0.006895**	0.011764	**0.011716**	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	**0.000017**	0.000019	**0.00005**	0.000064	0.0
40	5	94.73	28.1	≈ 0	0	0	0.000438	**0.000427**	0.001186	0.0001186	0.0
40	10	90.08	19.0	≈ 0	0	0	**0.001663**	0.00173	0.003472	0.003472	0.1
40	15	84.56	13.8	≈ 0	0	0	**0.004143**	0.004315	**0.006127**	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 (continuation): Randomly generated instances with $[L, U] = [10, 1000]$ and $w_i \in [1, 50]$

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