The optimality box in uncertain data for minimizing the sum of the weighted completion times of the given jobs

Yuri N. Sotskov[a], Tsung-Chyan Lai[b], Natalia G. Egorova[a], Frank Werner[c]

[a] *United Institute of Informatics Problems, National Academy of Sciences of Belarus, Surganova Street 6, Minsk, 220012 Belarus;*

[b] *School of Economics and Management, Harbin Engineering University, 145 Nantong Street, Harbin, 150001 China;*

[c] *Faculty of Mathematics, Otto-von-Guericke-University, Universitatsplatz 2, Magdeburg, 39106 Germany*

Abstract:

An uncertain single-machine scheduling problem is considered, where the processing time of a job can take any real value from a given segment. The criterion is to minimize the total weighted completion time of the n jobs, a weight being associated with each given job. We use the optimality box as a stability measure of the optimal schedule and derive an $O(n)$-algorithm for calculating the optimality box for a fixed permutation of the given jobs. We investigate properties of the optimality box using blocks of the jobs. If each job belongs to a single block, then the largest optimality box may be constructed in $O(n \log n)$ time. For the general case, we apply dynamic programming for constructing a job permutation with the largest optimality box. The computational results for finding a permutation with the largest optimality box show that such a permutation is close to an optimal one, which can be determined after completing the jobs when their processing times became known.

Keywords: Single-machine scheduling, Uncertain processing times, Optimality box.

MSC classification: 90B36, 90C31, 68W40.

January 23, 2017

1 Introduction

Uncertainties are present in most real-life scheduling problems. Several approaches were developed for dealing with scheduling problems under uncertainty. In a stochastic approach, uncertain parameters (e.g., processing probability distributions known before scheduling [Pinedo (2002)]. However, in many real-life situations, one may have no sufficient information to characterize the probability distribution for each random parameter. In such situations, other approaches are needed [Herroelen and Leus (2004), Sabuncuoglu and Goren (2009), Sotskov and Werner (2014)]. In the approach of seeking a robust schedule [Daniels and Kouvelis (1995), Kouvelis and Yu (1997), Yang and Yu (2002), Pereira (2016)], the decision-maker prefers a schedule that hedges against the worst-case scenario.

A fuzzy approach [Harikrishnan and Ishii (2005), Özelkan and Duckstein (1999), Wu (2010)] allows a scheduler to determine best schedules with respect to fuzzy parameters. A stability approach [Sotskov, Egorova, and Lai (2009), Sotskov and Lai (2012), Sotskov and Werner (2014)] combines a stability analysis of optimal schedules and the solution concept of a minimal dominant set of semi-active schedules.

In this paper, we consider a single-machine scheduling problem with interval processing times of n jobs. In Section 2, we present the problem settings and the state-of-the-art. In Section 3, we introduce the optimality box and derive an algorithm for constructing the optimality box for a fixed job permutation, which runs in $O(n)$ time. In Subsection 3.2, we investigate properties of the optimality box. Properties of the largest optimality box are investigated in Subsection 3.3, where efficient algorithms are derived for finding permutations with the largest optimality boxes for several special cases of the uncertain single-machine scheduling problem. The complexities of the derived algorithms vary from $O(\log n)$ to $O(n^2 \log n)$ depending on the characteristics of the solved instances. A general algorithm for constructing a job permutation with the largest optimality box is developed in Section 4. An illustrative example is given in Subsection 4.2. In Section 5, we report on computational results for finding a permutation with the largest perimeter of the optimality box. Section 6 includes comments and discusses some perspectives.

2 Problem settings and related literature

There are n jobs $\mathcal{J} = \{J_1, J_2, ..., J_n\}$ to be processed on a single machine. Associated with a job $J_i \in \mathcal{J}$ there is a weight $w_i > 0$ reflecting the job importance. The processing time p_i of the job $J_i \in \mathcal{J}$ can take any real value no less than a lower bound $p_i^L > 0$ and no greater than an upper bound $p_i^U \geq p_i^L$. The exact value $p_i \in [p_i^L, p_i^U]$ of the job processing time may remain unknown until the completion of the job J_i.

Let T denote the set of the feasible processing time vectors $p = (p_1, p_2, \ldots, p_n)$ in the space R_+^n of the non-negative n-dimensional real vectors. A vector $p \in T$ of the processing times is called a **scenario**. The set T of scenarios is the Cartesian product of the segments $[p_i^L, p_i^U]$:

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

Let $S = \{\pi_1, \pi_2, \ldots, \pi_{n!}\}$ denote the set of all permutations $\pi_k = (J_{k_1}, \ J_{k_2}, \ldots, J_{k_n})$ of the jobs $\mathcal{J}$. Given a scenario $p \in T$ and a permutation $\pi_k \in S$, let $C_i = C_i(\pi_k, p)$ denote the completion time of the job $J_i \in \mathcal{J}$ in the **semi-active** schedule [Pinedo (2002)] determined by the permutation π_k. The considered criterion $\sum w_i C_i$ denotes 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 the permutation $\pi_t = (J_{t_1}, J_{t_2}, \ldots, J_{t_n}) \in S$ is optimal for the criterion $\sum w_i C_i$. By adopting the three-field notation $\alpha|\beta|\gamma$ introduced by [Graham et al. (1979)], the above problem is denoted by $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$.

Next, we survey problem settings and some known results for scheduling jobs with uncertain or fixed numerical parameters, the results being used in Sections 3 – 6 are presented in detail. In the survey, an uncertain problem with the objective function $\gamma = f(C_1, C_2, \ldots, C_n)$ is denoted by $\alpha|p_i^L \leq p_i \leq p_i^U|\gamma$ and its deterministic counterpart by $\alpha||\gamma$. If the vector $p \in T$ of the job processing times is fixed before scheduling, i.e., the segment $[p_i^L, p_i^U]$ is degenerated into a point $[p_i, p_i]$ for each job $J_i \in \mathcal{J}$, then the uncertain problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ is turned into the deterministic one $1|| \sum w_i C_i$. We use the notation $1|p| \sum w_i C_i$ to indicate an instance of

the deterministic problem $1||\sum w_iC_i$ with a fixed scenario $p \in T$. Any instance $1|p|\sum w_iC_i$ can be solved in $O(n \log n)$ time [Smith (1956)] due to Theorem 1 available from [Coffman (1976)].

Theorem 1 *The 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 the following inequalities hold:*

$$\frac{w_{k_1}}{p_{k_1}} \geq \frac{w_{k_2}}{p_{k_2}} \geq \ldots \geq \frac{w_{k_n}}{p_{k_n}}. \tag{2}$$

Theorem 1 implies the following corollary [Sotskov and Lai (2012)].

Corollary 1 *If inequality $\frac{w_u}{p_u} > \frac{w_v}{p_v}$ holds, then job J_u precedes job J_v in all optimal permutations existing for the instance $1|p|\sum w_iC_i$.*

Since a scenario $p \in T$ is not fixed for the uncertain problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$, the completion time C_i of a job $J_i \in \mathcal{J}$ cannot be determined for the job permutation $\pi_k \in S$ before completion of the job J_i. Therefore, the value of the objective function $\sum w_iC_i$ for the permutation π_k remains uncertain until the jobs have been completed.

In the OR literature, several approaches for scheduling problems with uncertain numerical parameters have been developed [Herroelen and Leus (2004), Sabuncuoglu and Goren (2009), Sotskov and Werner (2014)]. Since for the problem $1|p_i^L \leq p_i \leq p_i^U|\gamma$, there does usually not exist a permutation of the jobs $\mathcal{J}$ remaining optimal for all scenarios T, an additional criterion is often introduced for dealing with the uncertain problem $1|p_i^L \leq p_i \leq p_i^U|\gamma$. A robust schedule minimizing the worst-case deviation from the optimality to hedge against data uncertainty has been developed in [Daniels and Kouvelis (1995), Goren and Sabuncuoglu (2008), Kasperski and Zielinski (2008), Kasperski and Zielinski(2011)] and in [Kouvelis and Yu (1997), Lu, Lin, and Ying (2012), Tadayon and Smith (2015)].

In [Daniels and Kouvelis (1995), Yang and Yu (2002)], the problem $1|p_i^L \leq p_i \leq p_i^U|\sum C_i$ was investigated using the robust approach. While the deterministic problem $1||\sum C_i$ is polynomially solvable [Smith (1956)], finding a permutation $\pi_t \in S$ minimizing the worst-case absolute regret or the relative regret are binary NP-hard even for two feasible scenarios [Daniels and Kouvelis (1995)]. The latter problem becomes unary NP-hard for an unbounded number of the discrete scenarios [Yang and Yu (2002)].

In [Kouvelis, Daniels, and Vairaktarakis (2000)], the binary NP-hardness was proven for finding a permutation $\pi_t \in S$ that minimizes the worst-case absolute regret for the two-machine flow-shop problem $F2|p_i^L \leq p_i \leq p_i^U|C_{\max}$ with the makespan criterion $C_{max} = \max\{C_i(\pi_t, p) : J_i \in \mathcal{J}\}$ even for two feasible scenarios. In [Daniels and Kouvelis (1995), Yang and Yu (2002)], several algorithms were developed to minimize the worst-case regret for the same problem. [Pereira (2016)] developed a branch-and-bound algorithm to find a permutation π_k minimizing the absolute regret for the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$. The computational experiments showed that the developed algorithm is able to find such a permutation π_k for the instances of moderate sizes, namely, for instances with up to 25 – 40 jobs depending on the instance characteristics.

The fuzzy scheduling technique was used in [Kasperski and Zielinski(2011), Wu (2010)] to develop a fuzzy analogue of dispatching rules or to solve mathematical programming problems to determine a schedule that minimizes a fuzzy-valued objective function [Harikrishnan and Ishii (2005), Özelkan and Duckstein (1999)].

In [Allahverdi, Aydilek, and Aydilek (2014)], several heuristics were developed for the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$. The computational experiments including different probability

distributions of the processing times showed that the error of the best performing heuristic was about 1% of the optimal objective function value obtained after completing the jobs when their factual processing times became known.

A similar approach was used in [Allahverdi and Aydilek (2010)], where heuristics were developed and compared on randomly generated instances for the problem $F2|p_i^L \leq p_i \leq p_i^U|C_{\max}$. By the computational experiments, it was shown that three of the proposed heuristics performed with an overall average error of less than 1%. The heuristic applying Johnson's algorithm to the mid-points of the given segments for the feasible processing times performed as the best one with an average error of less than 0.71%.

In Sections 3 – 5, we adopt the stability approach [Sotskov and Lai (2012)] for solving the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$ with the additional criterion of maximizing an optimality box for the desired job permutation. We use such an additional criterion since the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$ has often no certain solution due to the uncertainties of the job processing times. So, in what follows, a job permutation $\pi_k \in S$ is called **optimal** for the uncertain problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$ if the permutation $\pi_k \in S$ has the largest optimality box defined in Section 3, where the adjective "largest" is substantiated in Subsection 3.3 after discussing properties of the optimality box in Subsection 3.2.

3 The optimality box

For investigating properties of the optimality box for a job permutation $\pi_k \in S$, we introduce a strong dominance relation on the set $\mathcal{J}$ of jobs and prove a sufficient and necessary condition for a strong domination.

Definition 1 *Job J_u strongly dominates job J_v with respect to T if there is no optimal permutation $\pi_k \in S$ for the instance $1|p|\sum w_iC_i$ with some vector $p \in T$ such that job J_v precedes job J_u in the permutation π_k.*

Theorem 2 *Job J_u strongly dominates job J_v with respect to T if and only if*

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

Proof. *Sufficiency.* Let the inequality (3) hold. We need to prove that job J_u strongly dominates job J_v with respect to T, i.e., there is no optimal permutation $\pi_k \in S$ for the instance $1|p|\sum w_iC_i$ with a vector $p \in T$ such that job J_v precedes job J_u in the permutation π_k. For any scenario $p \in T$, the inequalities $p_u^L \leq p_u \leq p_u^U$ imply

$$\frac{w_u}{p_u^L} \geq \frac{w_u}{p_u} \geq \frac{w_u}{p_u^U} \tag{4}$$

while the inequalities $p_v^L \leq p_v \leq p_v^U$ imply

$$\frac{w_v}{p_v^L} \geq \frac{w_v}{p_v} \geq \frac{w_v}{p_v^U}. \tag{5}$$

From (3), (4), and (5), we obtain the inequalities $\frac{w_u}{p_u} \geq \frac{w_u}{p_u^U} > \frac{w_v}{p_v^L} \geq \frac{w_v}{p_v}$, i.e., for all feasible processing times $p_u \in [p_u^L, p_u^U]$ and $p_v \in [p_v^L, p_v^U]$, the following inequality holds:

$$\frac{w_u}{p_u} > \frac{w_v}{p_v}. \tag{6}$$

Due to Corollary 1, the inequality (6) implies that job J_u precedes job J_v in any optimal permutation $\pi_k \in S$ existing for the instance $1|p|\sum w_iC_i$ with the scenario $p \in T$. In other words, job J_u strongly dominates job J_v with respect to T. Sufficiency of the condition (3) has been proven.

Necessity. We prove necessity of condition (3) by a contradiction. Let job J_u strongly dominate job J_v with respect to T, i.e., there is no optimal permutation π_k for the instance $1|p|\sum w_iC_i$ with some scenario $p \in T$ such that job J_v precedes job J_u in the permutation π_k. However, we assume that condition (3) is violated, i.e., the opposite inequality

$$\frac{w_u}{p_u^U} \leq \frac{w_v}{p_v^L}$$

holds. We set the job processing times as follows: $p'_u = p_u^U$ and $p'_v = p_v^L$. Due to the assumption, we obtain $\frac{w_u}{p'_u} = \frac{w_u}{p_u^U} \leq \frac{w_v}{p_v^L} = \frac{w_v}{p'_v}$. Thus, the inequality $\frac{w_u}{p'_u} \leq \frac{w_v}{p'_v}$ holds and due to the sufficiency of Theorem 1, there exists an optimal permutation in the set S such that job J_v precedes job J_u. The obtained contradiction with our assumption completes the proof. ■

Let N_k denote a subset of the set $N = \{1, 2, \ldots, n\}$. We define an **optimality box** for the job permutation $\pi_k \in S$ as follows.

Definition 2 *The maximal rectangular box $\mathcal{OB}(\pi_k, T) = \times_{k_i \in N_k}[l^*_{k_i}, u^*_{k_i}] \subseteq T$ is called an optimality box for the permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$ with respect to T, if the permutation π_k being optimal for the instance $1|p|\sum w_iC_i$ with the scenario $p = (p_1, p_2, \ldots, p_n) \in T$ remains optimal for the instance $1|p'|\sum w_iC_i$ with any scenario $p' \in \mathcal{OB}(\pi_k, T) \times \{\times_{k_j \in N \setminus N_k}[p_{k_j}, p_{k_j}]\}$. If there does not exist a scenario $p \in T$ such that the permutation π_k is optimal for the instance $1|p|\sum w_iC_i$, then $\mathcal{OB}(\pi_k, T) = \emptyset$.*

Definition 2 implies that any variation p'_{k_i} of the processing time p_{k_i}, $J_{k_i} \in \mathcal{J}$, within the maximal segment $[l^*_{k_i}, u^*_{k_i}]$ determined in Definition 2 cannot violate the optimality of the permutation $\pi_k \in S$ provided that $p'_{k_i} \in [l^*_{k_i}, u^*_{k_i}]$. The non-empty segment $[l^*_{k_i}, u^*_{k_i}]$, where $l^*_{k_i} \leq u^*_{k_i}$, indicated in Definition 2 (and the non-empty segment $[\frac{w_{k_i}}{u^*_{k_i}}, \frac{w_{k_i}}{l^*_{k_i}}]$, where $\frac{w_{k_i}}{u^*_{k_i}} \leq \frac{w_{k_i}}{l^*_{k_i}}$) is called an **optimality segment** of the job J_{k_i} in the permutation π_k (an **optimality segment** of the weight-to-process ratio $\frac{w_{k_i}}{p_{k_i}}$, respectively). If the maximal segment $[l^*_{k_i}, u^*_{k_i}]$ is empty for the job $J_{k_i} \in \mathcal{J}$ in the permutation π_k, we say that job J_{k_i} **has no optimality segment** in the permutation π_k.

3.1 Calculating the optimality box for a fixed permutation of n jobs

We show that for calculating the optimality box $\mathcal{OB}(\pi_k, T)$, one can calculate the optimality box for the instance $1|\widehat{p}_i^L \leq p_i \leq \widehat{p}_i^U|\sum w_iC_i$, where the segments $[p_i^L, p_i^U]$ for the job processing times p_i are reduced, $[\widehat{p}_i^L, \widehat{p}_i^U] \subseteq [p_i^L, p_i^U]$, using the following formulas:

$$\frac{w_i}{\widehat{p}_i^L} = \min_{1 \leq j \leq i \leq n}\left\{\frac{w_j}{p_j^L}\right\}, \quad \frac{w_i}{\widehat{p}_i^U} = \max_{1 \leq i \leq j \leq n}\left\{\frac{w_j}{p_j^U}\right\}. \tag{7}$$

Theorem 3 *The optimality box of a permutation $\pi_k \in S$ for the problem $1|p_i^L \leq p_i \leq p_i^U|\sum w_iC_i$ is equal to the optimality box of the same permutation π_k for the problem $1|\widehat{p}_i^L \leq p_i \leq \widehat{p}_i^U|\sum w_iC_i$ with the given segments $[\widehat{p}_i^L, \widehat{p}_i^U]$, $J_i \in \mathcal{J}$, determined in (7).*

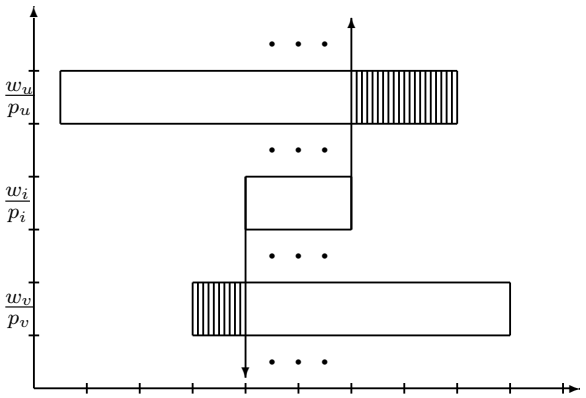

Figure 1: The reduced parts of the feasible segments of the weight-to-process ratios for the job J_u and job J_v caused by the job J_i (the reduced parts of the feasible segments are dashed).

Proof. From the sufficient and necessary conditions (2) for the optimality of a permutation π_k for the instance $1|p|\sum w_iC_i$, where $p \in T$, we obtain the following implications. If the inequalities $\frac{w_{k_i}}{p^L_{k_i}} \leq \frac{w_{k_i}}{p_{k_i}} < \frac{w_{k_i}}{\widehat{p}^L_{k_i}}$ hold for at least one processing time p_{k_i}, then the permutation $\pi_k = (J_{k_1}, \ldots, J_{k_i}, \ldots, J_{k_n})$ cannot be optimal for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$ for any fixed scenario $p \in T$ (see Fig. 1, where the reduced segments of the weight-to-process ratios are dashed). Similarly, if the inequalities $\frac{w_{k_i}}{p^U_{k_i}} \geq \frac{w_{k_i}}{p_{k_i}} > \frac{w_{k_i}}{\widehat{p}^U_{k_i}}$ hold for at least one processing time p_{k_i}, then the permutation $\pi_k = (J_{k_1}, \ldots, J_{k_i}, \ldots, J_{k_n})$ cannot be optimal for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$ for any fixed scenario $p \in T$ (see Fig. 1). Therefore, the set of all scenarios $p \in T$, for which the permutation π_k may be optimal for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$, is contained in the set of all scenarios $p \in T$, for which the permutation π_k may be optimal for the problem $1|\widehat{p}^L_i \leq p_i \leq \widehat{p}^U_i|\sum w_iC_i$ with the reduced segments $[\widehat{p}^L_i, \widehat{p}^U_i]$, $J_i \in \mathcal{J}$.

On the other hand, since the inclusion $[\widehat{p}^L_i, \widehat{p}^U_i] \subseteq [p^L_i, p^U_i]$ holds for each job $J_i \in \mathcal{J}$, the set of all scenarios $p \in T$, for which the permutation π_k may be optimal for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$, contains the set of all scenarios $p \in T$, for which the permutation π_k may be optimal for the problem $1|\widehat{p}^L_i \leq p_i \leq \widehat{p}^U_i|\sum w_iC_i$ with the reduced segments $[\widehat{p}^L_i, \widehat{p}^U_i]$, $J_i \in \mathcal{J}$. Thus, one can conclude that the scenario p and the sets of scenarios p' considered in Definition 2 for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$ and those for the problem $1|\widehat{p}^L_i \leq p_i \leq \widehat{p}^U_i|\sum w_iC_i$ are the same. Hence, the optimality box of the permutation $\pi_k \in S$ for the problem $1|p^L_i \leq p_i \leq p^U_i|\sum w_iC_i$ is equal to the optimality box of the permutation π_k for the problem $1|\widehat{p}^L_i \leq p_i \leq \widehat{p}^U_i|\sum w_iC_i$. ■

Based on Theorem 3, we propose the following $O(n)$-algorithm for determining the optimality box $\mathcal{OB}(\pi_k, T)$ for any fixed permutation $\pi_k \in S$.

Algorithm FIX-OPTBOX

Input: Segments $[p^L_i, p^U_i]$ and weights w_i of the jobs $J_i \in \mathcal{J}$.
The permutation $\pi_k = (J_{k_1}, J_{k_2}, \ldots, J_{k_n}) \in S$.
Output: Optimality box $\mathcal{OB}(\pi_k, T)$ for the given permutation $\pi_k \in S$.
Step 1: **FOR** $i = 1$ to n **DO** set $\widehat{p}^L_i = p^L_i$, $\widehat{p}^U_i = p^U_i$ **END FOR**
set $t_L = \frac{w_1}{\widehat{p}^L_1}$, $t_U = \frac{w_n}{\widehat{p}^U_n}$
Step 2: **FOR** $i = 2$ to n **DO**
IF $\frac{w_i}{\widehat{p}^L_i} > t_L$ **THEN** set $\frac{w_i}{\widehat{p}^L_i} = t_L$ **ELSE** set $t_L = \frac{w_i}{\widehat{p}^L_i}$ **END FOR**

Step 3: **FOR** $i = n-1$ to 1 **STEP** -1 **DO**
IF $\frac{w_i}{\widehat{p}_i^U} < t_U$ **THEN** set $\frac{w_i}{\widehat{p}_i^U} = t_U$ **ELSE** set $t_U = \frac{w_i}{\widehat{p}_i^U}$ **END FOR**

Step 4: Calculate the weight-to-process ratios $\frac{w_i}{\widehat{p}_i^L}$ and $\frac{w_i}{\widehat{p}_i^U}$ using formulas (7)

Step 5: Set $w_0 = 1$, $\widehat{p}_0^U = \frac{\widehat{p}_1^L}{w_1}$, $w_{n+1} = 1$, $\widehat{p}_{n+1}^L = \frac{\widehat{p}_n^U}{w_n}$

Step 6: **FOR** $i = 1$ to n **DO** set $\widehat{d}_{k_i}^- = \max\left\{\frac{w_{k_i}}{\widehat{p}_{k_i}^U}, \frac{w_{k_{i+1}}}{\widehat{p}_{k_{i+1}}^L}\right\}$, $\widehat{d}_{k_i}^+ = \min\left\{\frac{w_{k_i}}{\widehat{p}_{k_i}^L}, \frac{w_{k_{i-1}}}{\widehat{p}_{k_{i-1}}^U}\right\}$
END FOR

Step 7: Set $\mathcal{OB}(\pi_k, T) = \times_{\widehat{d}_i^- \le \widehat{d}_i^+}\left[\frac{w_{k_i}}{\widehat{d}_{k_i}^+}, \frac{w_{k_i}}{\widehat{d}_{k_i}^-}\right]$ **STOP**.

In steps 1 – 4 of Algorithm FIX-OPTBOX, the problem $1|\widehat{p}_i^L \le p_i \le \widehat{p}_i^U|\sum w_iC_i$ with the reduced segments of the job processing times is constructed. Due to Theorem 3, the optimality box of the job permutation for the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ coincides with the optimality box for the same permutation for the problem $1|\widehat{p}_i^L \le p_i \le \widehat{p}_i^U|\sum w_iC_i$. In steps 5 – 7, the optimality box for the fixed permutation π_k for the problem $1|\widehat{p}_i^L \le p_i \le \widehat{p}_i^U|\sum w_iC_i$ is constructed. It takes $O(n)$ time to construct the optimality box for any fixed permutation $\pi_k \in S$ using Algorithm FIX-OPTBOX.

3.2 Properties of job permutations based on blocks of jobs

Using the following definition and the proof of Lemma 1, which follows, one can determine subsets of the set of jobs $\mathcal{J}$ called blocks for the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$.

Definition 3 *A maximal set $B_r = \{J_{r_1}, J_{r_2}, \ldots, J_{r_{|B_r|}}\} \subseteq \mathcal{J}$ of the jobs, for which the inequality*

$$\min_{J_{r_i} \in B_r}\left\{\frac{w_{r_i}}{p_{r_i}^L}\right\} \ge \max_{J_{r_i} \in B_r}\left\{\frac{w_{r_i}}{p_{r_i}^U}\right\}$$

holds, is called a **block**. *The segment $[b_r^L, b_r^U]$, where*

$$b_r^L = \max_{J_{r_i} \in B_r}\left\{\frac{w_{r_i}}{p_{r_i}^U}\right\} \text{ and } b_r^U = \min_{J_{r_i} \in B_r}\left\{\frac{w_{r_i}}{p_{r_i}^L}\right\},$$

is called the **core** *of the block B_r.*

An example of the block $B_1 = \{J_1, J_2, J_3, J_4, J_5\} \subseteq \mathcal{J}$ is presented in Fig. 2, where the jobs of the block B_1 are ordered as follows: $(J_1, J_2, J_3, J_4, J_5)$. The core $[b_1^L, b_1^U]$ of the block B_1 is dashed in Fig. 2. If job $J_i \in \mathcal{J}$ belongs to only one block, we say that job J_i is **fixed** in this block. Otherwise, if job $J_k \in \mathcal{J}$ belongs to two or more blocks, we say that job J_k is **non-fixed** in the blocks.

Lemma 1 *For the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$, the set of blocks $B = \{B_1, B_2, \ldots, B_m\}$ can uniquely be determined in $O(n \log n)$ time.*

Proof. Since the maximum in the finite set $\{\frac{w_1}{p_1^U}, \frac{w_2}{p_2^U}, \ldots, \frac{w_n}{p_n^U}\}$ of real numbers is uniquely determined, one can find the left bound b_1^L of the core of the first block $B_1 \in B$ as follows: $b_1^L = \max_{J_i \in \mathcal{J}}\{\frac{w_i}{p_i^U}\}$. Then all jobs included into the first block B_1 may be determined as follows: $B_1 = \{J_i \in \mathcal{J} : \frac{w_i}{p_i^U} \le b_1^L \le \frac{w_i}{p_i^L}\}$. The right bound b_1^U of the core of first block B_1 is uniquely determined as follows: $b_1^U = \min_{J_i \in B_1}\{\frac{w_i}{p_i^L}\}$.

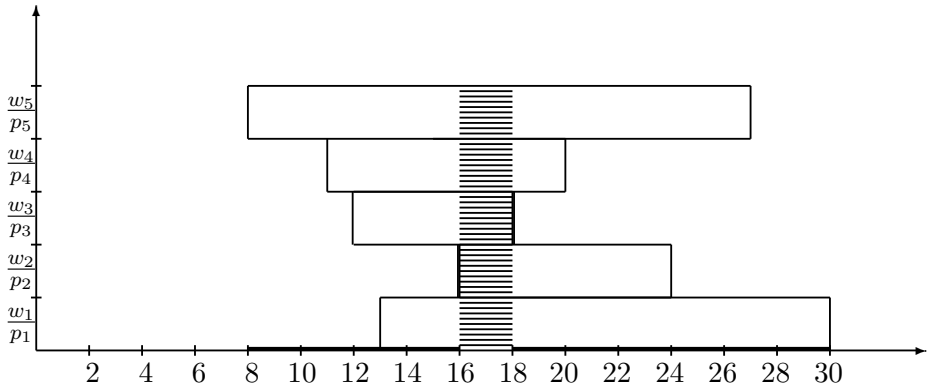

Figure 2: The feasible weight-to-process ratios for the block $B_1 = \{J_1, J_2, J_3, J_4, J_5\} \subseteq \mathcal{J}$ (the weight-to-process ratios corresponding to the core of the block B_1 are dashed).

Then one can determine the second block $B_2 \in B$ via applying the above procedure to the subset of set $\mathcal{J}$ without the jobs J_i, for which the equality $b_1^L = \frac{w_i}{p_i^U}$ holds.

This process is continued until determining the last block B_m in the set B.

Thus, one can use the following $O(n \log n)$-algorithm for constructing the set of blocks $B = \{B_1, B_2, \ldots, B_m\}$. All jobs $J_i \in \mathcal{J}$ are considered in non-increasing order of the values $\widetilde{b}_i^L = \frac{w_i}{p_i^U}$. For each currently considered job J_i, one determines the set of jobs $B_j = \{J_k \in \mathcal{J} : \frac{w_k}{p_k^U} \leq \widetilde{b}_j^L \leq \frac{w_k}{p_k^L}\}$ and finds the value of $\widetilde{b}_i^U = \min_{J_v \in B_j}\{\frac{w_v}{p_v^L}\}$. If the inequality $\widetilde{b}_i^U < \widetilde{b}_{i-1}^L$ holds, then the segment $[\widetilde{b}_i^L, \widetilde{b}_i^U]$ is the core of the new block B_z, i.e., $[b_z^L, b_z^U] := [\widetilde{b}_i^L, \widetilde{b}_i^U]$. If $\widetilde{b}_i^L < \min_{J_g \in \mathcal{J}}\{\frac{w_g}{p_g^L}\}$, the set B of blocks is already constructed. ■

Lemma 2 *If the optimality box $\mathcal{OB}(\pi_k, T)$ is not empty, then any two fixed jobs in different blocks of the set B must be ordered in the permutation $\pi_k \in S$ with decreasing left (and right) bounds of the cores of their blocks.*

Proof. We prove Lemma 2 by a contradiction. Let the condition $\mathcal{OB}(\pi_k, T) \neq \emptyset$ hold. However, we assume that there are a fixed job $J_v \in B_r$ and a fixed job $J_u \in B_s$, $r \neq s$, which are located in the permutation $\pi_k \in S$ in increasing order of the left bounds of the cores of their blocks. In such a case, the permutation π_k looks as follows: $\pi_k = \{\ldots, J_v, \ldots, J_u, \ldots\}$, where $b_r^L < b_s^L$.

Using Definition 3, it is easy to convince that the cores of any two blocks have no common point. Therefore, the inequality $b_r^L < b_s^L$ implies the inequality $b_r^U < b_s^L$.

For any feasible processing time p_v of the job $J_v \in \mathcal{J}$ and any feasible processing time p_u of the job $J_u \in \mathcal{J}$, $p \in T$, we obtain the following inequalities: $\frac{w_v}{p_v} \leq \frac{w_v}{p_v^L} \leq b_r^U < b_s^L \leq \frac{w_u}{p_u^L} \leq \frac{w_u}{p_u}$. Hence, the inequality $\frac{w_v}{p_v} < \frac{w_u}{p_u}$ holds and due to Corollary 1, the permutation $\pi_k \in S$ cannot be optimal for any fixed scenario $p \in T$. The equality $\mathcal{OB}(\pi_k, T) = \emptyset$ holds due to Definition 2. The obtained contradiction completes the proof. ■

In what follows, it is assumed that all blocks in the set $B = \{B_1, B_2, \ldots, B_m\}$ are numbered according to decreasing left bounds of their cores, i.e., the inequality $b_v^L < b_u^L$ implies the inequality $v < u$.

Remark 1 *Any permutation $\pi_k \in S$ determines a block of the set B, to which a non-fixed job (if any) belongs. Due to such a fixing of the positions of all non-fixed jobs, the permutation $\pi_k \in S$ may generate an additional set $B(\pi_k)$ of the blocks. Each block from the generated set $B(\pi_k)$ is called a* **virtual block**.

Lemma 3 *Assume that $\mathcal{OB}(\pi_k, T) \neq \emptyset$. If there exists a non-fixed job $J_i \in \mathcal{J}$ belonging to the blocks $B_k \in B$ and $B_{k+s} \in B$, $s \geq 2$, then job J_i must belong to all pairwise adjacent blocks $B_k, B_{k+1}, \ldots, B_{k+s}$ from the set B.*

Proof. Let the non-fixed job J_i belong to two blocks $B_k \in B$ and $B_{k+s} \in B$, where $s \geq 2$. Then the following relations must hold:

$$\frac{w_i}{p_i^U} \leq b_k^L = \max_{J_i \in B_k} \left\{ \frac{w_i}{p_i^U} \right\} \text{ and } \frac{w_i}{p_i^L} \geq b_{k+s}^L = \min_{J_i \in B_{k+s}} \left\{ \frac{w_i}{p_i^L} \right\}.$$

In any permutation π_k with a non-empty optimality box, $\mathcal{OB}(\pi_k, T) \neq \emptyset$, all fixed jobs must be ordered according to decreasing left bounds of the cores of their blocks (due to Lemma 2). As a result, the following inequalities must hold: $b_k^L \leq b_k^U < b_{k+1}^L \leq b_{k+1}^U < \ldots < b_{k+s}^L \leq b_{k+s}^U$. Therefore, the cores of all blocks $B_{k+1}, B_{k+2}, \ldots, B_{k+s-1}$ must be contained in the segment $[\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}]$. Hence, the non-fixed job J_i must be contained in all blocks $B_k, B_{k+1}, \ldots, B_{k+s}$. Furthermore, all blocks containing the non-fixed job J_i must be one by one adjacent. ■

We call a block $B_r \in B$ **trivial** if it consists only of identical jobs, i.e., $B_r = \{J_i \in \mathcal{J} : p_i^U = p^U(r),\ p_i^L = p^L(r),\ w_i = w(r)\}$. If the set $B_r \in B$ is a singleton, $|B_r| = 1$, then the block B_r is trivial. Obviously, any virtual block $B_t \in B(\pi_k)$ cannot be trivial.

Theorem 4 *For the problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$, each permutation $\pi_k \in S$ has an empty optimality box $\mathcal{OB}(\pi_k, T) = \emptyset$ if and only if any block $B_r \in B$ is trivial with $|B_r| \geq 2$ and all jobs $J_i \in \mathcal{J}$ are fixed in their blocks from the set B.*

Proof. *Sufficiency.* Due to Lemma 2, in the permutation $\pi_k \in S$ with the non-empty optimality box $\mathcal{OB}(\pi_k, T) \neq \emptyset$, all blocks of the set B must be located in decreasing order of the left (and right) bounds of their cores. Let all blocks in the set B be trivial and all jobs from the set $\mathcal{J}$ be fixed in their blocks. Hence, in the desired permutation $\pi_k \in S$, the following equalities must hold for each block $B_r \in B$:

$$\min_{J_i \in B_r} \left\{ \frac{w_i}{p_i^U} \right\} = \frac{w(r)}{p^U(r)} = \max_{J_i \in B_r} \left\{ \frac{w_i}{p_i^U} \right\} \text{ and } \max_{J_i \in B_r} \left\{ \frac{w_i}{p_i^L} \right\} = \frac{w(r)}{p^L(r)} = \min_{J_i \in B_r} \left\{ \frac{w_i}{p_i^L} \right\}.$$

Since $|B_r| \geq 2$, there is no job $J_v \in B_r$, which has an optimality segment in any fixed permutation $\pi_k \in S$. Hence, the equality $\mathcal{OB}(\pi_k, T) = \emptyset$ must hold. The sufficiency has been proven.

Necessity. We prove necessity by a contradiction. Let the permutation $\pi_k \in S$ have a non-empty optimality box $\mathcal{OB}(\pi_k, T) \neq \emptyset$. However, we assume that there exists a non-trivial block $B_r \in B$ or there is a non-fixed job $J_i \in \mathcal{J}$. On the one hand, for the non-trivial block $B_r \in B$ (if any), at least one of the following inequalities must hold:

$$\min_{J_i \in B_r} \left\{ \frac{w_i}{p_i^U} \right\} < \max_{J_i \in B_r} \left\{ \frac{w_i}{p_i^U} \right\} \text{ or } \max_{J_i \in B_r} \left\{ \frac{w_i}{p_i^L} \right\} > \min_{J_i \in B_r} \left\{ \frac{w_i}{p_i^L} \right\}.$$

Hence, in such a case, there is a job in the block B_r with the following optimality segment: $\max_{J_i \in B_r}\{\frac{w_i}{p_i^U}\} - \min_{J_i \in B_r}\{\frac{w_i}{p_i^U}\} + \max_{J_i \in B_r}\{\frac{w_i}{p_i^L}\} - \min_{J_i \in B_r}\{\frac{w_i}{p_i^L}\} > 0$.

On the other hand, if there exists a non-fixed job $J_i \in \mathcal{J}$, then due to Lemma 3, there are at least two adjacent blocks B_r and B_{r+1}, which both contain the job J_i. Then the permutation $\pi_k \in S$ with a non-empty optimality box $\mathcal{OB}(\pi_k, T) \neq \emptyset$ will be obtained if job J_i will be located in the permutation π_k on the last place among the jobs belonging to block B_r and on the first place among the jobs belonging to block B_{r+1}.

In both cases tested, we obtain a contradiction that there exists a permutation $\pi_k \in S$ with a non-empty optimality box $\mathcal{OB}(\pi_k, T) \neq \emptyset$. Theorem 4 has been proven. ■

3.3 A job permutation with the largest optimality box

Before investigating properties of the permutation $\pi_k \in S$ with the largest optimality box $\mathcal{OB}(\pi_k, T)$, we have to define what the adjective "largest" may mean, since an optimality box is characterized by several parameters such as dimension, volume and perimeter of the box. Note that due to the possible existence of virtual blocks (see Remark 1), optimality boxes constructed for different permutations from the set S may have different dimensions, which in turn may influence the volumes of the optimality boxes.

If the permutation $\pi_k \in S$ has the optimality box $\mathcal{OB}(\pi_k, T)$ with a larger volume and a larger dimension than the optimality box $\mathcal{OB}(\pi_t, T)$, where $\pi_k \neq \pi_t \in S$, then the permutation $\pi_k \in S$ seems to be better than the permutation π_t with respect to the objective function value $\sum w_i C_i$. In other words, one can expect that the inequality

$$\sum_{J_i \in \mathcal{J}} w_i C_i(\pi_k, p) < \sum_{J_i \in \mathcal{J}} w_i C_i(\pi_t, p) \tag{8}$$

holds for more scenarios p from the set T than the inequality, which is opposite to inequality (8). However, it is not clear, which permutation will be better: either the permutation $\pi_k \in S$ having an optimality box with a larger volume and a smaller dimension of the optimality box or the permutation $\pi_t \in S$ having an optimality box with a larger dimension and a smaller volume of the optimality box. Another unclear question arises when the optimality segment $[l_i^*, u_i^*]$ of a job $J_i \in \mathcal{J}$ is degenerated into one point: $l_i^* = u_i^*$. The volume $Vol\mathcal{OB}(\pi_k, T)$ of the optimality box $\mathcal{OB}(\pi_k, T)$ is calculated as follows:

$$Vol\mathcal{OB}(\pi_k, T) = \prod_{J_i \in \mathcal{J}(\pi_k)} (u_i^* - l_i^*), \tag{9}$$

where the set $\mathcal{J}(\pi_k)$ consists of all jobs from the set $\mathcal{J}$ having optimality segments in the permutation π_k.

Note that all jobs $J_i \in \mathcal{J}$ having an optimality segment $[l_i^*, l_i^*]$ of a zero length, $l_i^* - l_i^* = 0$, must be excluded from the product on the right-hand side of the equality (9) used for calculating the volume $Vol\mathcal{OB}(\pi_k, T)$. Otherwise, one obtains a zero volume of the optimality box, $Vol\mathcal{OB}(\pi_k, T) = 0$, irrespectively of other optimality segments $[l_r^*, u_r^*]$ for the jobs $J_r \in \mathcal{J}$ with $r \neq i$, for which the strict inequality $l_i^* < u_i^*$ may hold.

To avoid the above questions, we shall compare the perimeters of the optimality boxes for permutations from the set S. We shall call the permutation $\pi_k \in S$ **optimal** for the uncertain problem $1|p_i^L \leq p_i \leq p_i^U| \sum w_i C_i$ if the permutation $\pi_k \in S$ has an optimality box $\mathcal{OB}(\pi_k, T)$ having the largest perimeter. Nevertheless, in the claims proven in this subsection, the term "largest optimality box" may be used with respect to any of the three characteristics: dimension, volume or perimeter of the optimality box $\mathcal{OB}(\pi_k, T)$.

Lemma 4 *In any permutation $\pi_k \in S$, at most two jobs from the block $B_r \in B \bigcup B(\pi_k)$ may have optimality segments. If the equality $B_1 = \mathcal{J}$ holds, then it takes $O(\log n)$ time to find the largest optimality box $\mathcal{OB}(\pi_k, T)$ for such a problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$.*

Proof. The core $[b_r^L, b_r^U]$ of the block B_r is included into the segment $[\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}]$ for any job $J_i \in B_r$, i.e., $[b_r^L, b_r^U] \subseteq [\frac{w_i}{p_i^U}, \frac{w_i}{p_i^L}]$. Hence, due to Theorem 3, only two jobs from the set B_r may have optimality segments in any fixed permutation $\pi_k \in S$ (see Fig. 2, where only job J_1 and job J_5 have optimality segments). To be precise, we say that only job $J_{k_i} \in B_r$, which is the first one in the permutation π_k among the jobs from the block B_r, and only job $J_{k_v} \in B_r$, which is the last one in the permutation π_k among the jobs from the block B_r, may have optimality segments. Such a permutation π_k looks as follows: $\pi_k = (\ldots, J_{k_i}, J_{k_{i+1}}, \ldots, J_{k_v}, \ldots)$, where $\{J_{k_i}, J_{k_{i+1}}, \ldots, J_{k_v}\} = B_r$.

It is easy to convince that the length $u_{k_i}^* - l_{k_i}^*$ of the optimality segment $[l_{k_i}^*, u_{k_i}^*]$ of the job J_{k_i} located in the permutation π_k is determined by the second job $J_{k_{i+1}}$ in the permutation π_k among the jobs from the block B_r (see Fig. 2, where the length of the optimality segment for the job J_1 is determined by the job J_2).

Similarly, the length $u_{k_v}^* - l_{k_v}^*$ of the optimality segment $[l_{k_v}^*, u_{k_v}^*]$ of the job J_{k_v} located in the permutation π_k is determined by the second-to-last job $J_{k_{v-1}}$ in the permutation π_k among the jobs from the block B_r (see Fig. 2, where the length of the optimality segment for the job J_5 is determined by the job J_4).

Hence, to find the largest optimality box $\mathcal{OB}(\pi_k, T)$ provided that $B_r = \mathcal{J}$, where $r = 1$ in this case, it is needed to find only four jobs J_{k_i}, $J_{k_{i+1}}$, $J_{k_{v-1}}$ and J_{k_v} in the block B_r, which provide the largest optimality segments for job J_{k_i} and job J_{k_v}. To this end, we should choose the job J_{k_i} such that the following equality holds:

$$\frac{w_{k_i}}{p_{k_i}^L} = \max_{J_{k_s} \in B_r} \left\{ \frac{w_{k_s}}{p_{k_s}^L} \right\}.$$

Then, if $B_r \setminus \{J_{k_i}\} \neq \emptyset$, we should choose the job J_{k_v} such that

$$\frac{w_{k_v}}{p_{k_v}^U} = \min_{J_{k_s} \in B_r \setminus \{J_{k_i}\}} \left\{ \frac{w_{k_s}}{p_{k_s}^U} \right\}.$$

If $B_r \setminus \{J_{k_i}, J_{k_v}\} \neq \emptyset$, we should choose the job $J_{k_{i+1}}$ such that

$$\frac{w_{k_{i+1}}}{p_{k_{i+1}}^L} = \min_{J_{k_s} \in B_r \setminus \{J_{k_i}, J_{k_v}\}} \left\{ \frac{w_{k_s}}{p_{k_s}^L} \right\}.$$

If $B_r \setminus \{J_{k_i}, J_{k_{i+1}}, J_{k_v}\} \neq \emptyset$, we should choose the job $J_{k_{v-1}}$ such that

$$\frac{w_{k_{v-1}}}{p_{k_{v-1}}^U} = \min_{J_{k_s} \in B_r \setminus \{J_{k_i}, J_{k_{i+1}}, J_{k_v}\}} \left\{ \frac{w_{k_s}}{p_{k_s}^U} \right\}.$$

Thus, one has either to use a single job, if $|B_r| = 1$, or to test two jobs, if $|B_r| = 2$, or three jobs, if $|B_r| = 3$, or to choose and test four jobs from the block B_r, if $|B_r| \ge 4$, independently of how large the cardinality $|B_r| \le n$ of the set $B_r \in B \bigcup B(\pi_k)$ is. In the worst case, one has to test at most $4! = 24$ permutations of the jobs chosen from the set B_r. Due to such a direct testing of at most 24 permutations of the chosen jobs, one selects a permutation of $|B_r|$ jobs included in the block B_r, which provides the largest optimality box $\mathcal{OB}(\pi_k(B_r), T)$

for all $|B_r|!$ permutations $\pi_k(B_r)$ of the jobs from the block B_r. It is easy to convince that the described algorithm for finding a permutation $\pi_k(B_r)$ of the jobs B_r with the largest optimality box $\mathcal{OB}(\pi_k(B_r), T)$ takes $O(\log |B_r|)$ time or $O(\log n)$ time for the case when $B_1 = \mathcal{J}$. Lemma 4 has been proven. ■

To illustrate Lemma 4, we consider the block $B_1 = \{J_1, J_2, J_3, J_4, J_5\}$ presented in Fig. 2. Let the equality $B_1 = \mathcal{J}$ hold, i.e., $n = 5$ for the considered problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$. In what follows, we call this problem Example 1.

For the permutation $\pi_s = (J_1, J_2, J_3, J_4, J_5)$ used in Fig. 2, the two jobs J_1 and J_5 have optimality segments while the other three jobs J_2, J_3 and J_4 have no optimality segments. From Fig. 2, it follows that the optimality segment of the weight-to-process ratio $[\frac{w_1}{u_1^*}, \frac{w_1}{l_1^*}] = [8, 11]$ for the job J_1 has a length equal to 3. The weight-to-process ratio $[\frac{w_5}{u_5^*}, \frac{w_5}{l_5^*}] = [24, 30]$ for the job J_5 has a length equal to 6. Using the $O(\log n)$-algorithm described in Lemma 4, one can find the largest optimality box $\mathcal{OB}(\pi_k, T)$ along with the following permutation $\pi_k = (J_1, J_2, J_4, J_3, J_5) \in S$ having the largest optimality box. In the obtained permutation $\pi_k = (J_1, J_2, J_4, J_3, J_5)$, the optimality segment $[8, 16]$ of the weight-to-process ratio for the job J_1 has a maximal possible length equal to $Per^1 = 8$ and the optimality segment $[18, 30]$ of the weight-to-process ratio for the job J_5 has a maximal possible length equal to $Per^2 = 12$. Note that the mid-point permutation $\pi_e = (J_1, J_3, J_4, J_2, J_5) \in S$, where all jobs J_i are ordered according to decreasing mid-points $\frac{p_i^U - p_i^L}{2}$ of their segments $[p_i^L, p_i^U]$, has a smaller optimality box $\mathcal{OB}(\pi_e, T)$ with the optimality segment $[8, 12]$ of the weight-to-process ratio for the job J_1 and the optimality segment $[24, 30]$ of the weight-to-process ratio for the job J_5.

The perimeter of the optimality box $\mathcal{OB}(\pi_k, T)$ for the permutation $\pi_k = (J_1, J_2, J_4, J_3, J_5)$ is equal to $8 + 12 = 20$, while that for the permutation $\pi_e = (J_1, J_3, J_4, J_2, J_5)$ is equal to $4 + 6 = 10$. It is easy to convince that permutation π_k reaches the maximal possible sum of the length $Per^1 + Per^2$ of the optimality segments of the jobs $\mathcal{J} = B_1$.

Lemma 5 *Let there exist two adjacent blocks $B_r \in B$ and $B_{r+1} \in B$, such that $B_r \bigcap B_{r+1} = \emptyset$. The problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ can be decomposed into the subproblem P_1 with the set of jobs $\mathcal{J}_1 = \bigcup_{k=1}^{r} B_k$ and the subproblem P_2 with the set of jobs $\mathcal{J}_2 = \bigcup_{k=r+1}^{m} B_k = \mathcal{J} \setminus \mathcal{J}_1$. The largest optimality box $\mathcal{OB}(\pi_k, T)$ for such a problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ may be obtained as the Cartesian product of the largest optimality boxes constructed for the problems P_1 and P_2.*

Proof. Since the blocks B_r and B_{r+1} have no common jobs, the inequality $\frac{w_u}{p_u^U} > \frac{w_v}{p_v^L}$ holds for each pair of jobs $J_u \in \bigcup_{k=1}^{r} B_k$ and $J_v \in \bigcup_{k=r+1}^{m} B_k$. Due to Theorem 2, any job $J_u \in \bigcup_{k=1}^{r} B_k$ strongly dominates any job $J_v \in \bigcup_{k=r+1}^{m} B_k$ with respect to T. Due to Definition 1, there is no optimal permutation $\pi_k \in S$ for the instance $1|p| \sum w_i C_i$ with a scenario $p \in T$ such that job J_v precedes job J_u in the permutation π_k. Due to Definition 2, a non-empty optimality box $\mathcal{OB}(\pi_k, T)$ may be possible only if job J_u is located before job J_v in the permutation π_r. The largest optimality box $\mathcal{OB}(\pi_k, T)$ for the problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ may be obtained as the Cartesian product of the largest optimality boxes constructed for the problems P_1 and P_2, where the set of jobs $\bigcup_{k=1}^{r} B_k$ and the set of jobs $\bigcup_{k=r+1}^{m} B_k$ are considered separately one from another. ■

Similarly to the above proof, one can prove Theorem 5, as a straightforward generalization of Lemma 5 for the case, where each pair of adjacent block subsets $\mathcal{B}_{r_i}^{t_i} = \{B_{r_i+1}, B_{r_i+2}, \ldots, B_{r_i+t_i}\}$ and $\mathcal{B}_{r_{i+1}}^{t_{i+1}} = \{B_{r_{i+1}+1}, B_{r_{i+1}+2}, \ldots, B_{r_{i+1}+t_{i+1}}\}$ from the set $\{\mathcal{B}_{r_1}^{t_1}, \mathcal{B}_{r_2}^{t_2}, \ldots, \mathcal{B}_{r_v}^{t_v}\}$ of the block subsets have no common job, i.e., the equality $\{B_{r_i+1} \cup B_{r_i+2} \cup \ldots \cup B_{r_i+t_i}\} \cap \{B_{r_{i+1}+1} \cup B_{r_{i+1}+2} \cup$

$\ldots \cup B_{r_{i+1}+t_{i+1}}\} = \emptyset$ holds. Here and in Theorem 5, it is assumed that the equality $r_{i+1} = r_i + t_i$ holds for each index $i \in \{1, 2, \ldots, v-1\}$. The notation $\mathcal{J}(\mathcal{B}_{r_i}^{t_i})$ is used for the subset of the set $\mathcal{J}$ of the jobs belonging to the following union of blocks: $B_{r_i+1} \cup B_{r_i+2} \cup \ldots \cup B_{r_i+t_i}$, where $\{B_{r_i+1}, B_{r_i+2}, \ldots B_{r_i+t_i}\} = \mathcal{B}_{r_i}^{t_i}$.

Theorem 5 *Let each pair of adjacent block subsets $\mathcal{B}_{r_i}^{t_i} = \{B_{r_i+1}, B_{r_i+2}, \ldots, B_{r_i+t_i}\}$ and $\mathcal{B}_{r_{i+1}}^{t_{i+1}} = \{B_{r_{i+1}+1}, B_{r_{i+1}+2}, \ldots, B_{r_{i+1}+t_{i+1}}\}$ from the set $\{\mathcal{B}_{r_1}^{t_1}, \mathcal{B}_{r_2}^{t_2}, \ldots, \mathcal{B}_{r_v}^{t_v}\}$ of the block subsets have no common job. Then the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ can be decomposed into the subproblems $P_1, P_2, \ldots, P_v$ with the job sets $\mathcal{J}(\mathcal{B}_{r_1}^{t_1})$, $\mathcal{J}(\mathcal{B}_{r_2}^{t_2}), \ldots, \mathcal{J}(\mathcal{B}_{r_v}^{t_v})$, respectively. The largest optimality box $\mathcal{OB}(\pi_k, T)$ may be obtained as the Cartesian product of the largest optimality boxes constructed for the problems $P_1, P_2, \ldots, P_v$.*

The constructive proof of the following claim allows us to describe an $O(n \log n)$-algorithm for constructing an optimal permutation for a special case of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$.

Theorem 6 *Let all jobs from the set $\mathcal{J}$ be fixed in their blocks from the set B. Then the permutation π_k with the largest optimality box $\mathcal{OB}(\pi_k, T)$ may be constructed in $O(n \log n)$ time.*

Proof. Based on Lemmas 1, 2, 4, 5, and Theorem 5, we derive the following algorithm for constructing a permutation π_k with the largest optimality box $\mathcal{OB}(\pi_k, T)$ for the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ provided that the set $\mathcal{J}$ consists only of fixed jobs in their blocks. First, using the $O(n \log n)$-algorithm described in the proof of Lemma 1, we construct a set of blocks $B = \{B_1, B_2, \ldots, B_m\}$. Then, using Lemma 2, we order the jobs of the set $\mathcal{J}$ according to decreasing left (and right) bounds of the cores of their blocks. As a result, all n jobs of the set $\mathcal{J}$ will be linearly ordered since all jobs $\mathcal{J}$ are fixed in their blocks. Such an ordering of all jobs $\mathcal{J}$ takes $O(n \log n)$ time.

Then, using Lemma 5 and Theorem 5, we decompose the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ into m subproblems $P_1, P_2, \ldots, P_m$, where the set of jobs $\mathcal{J}_k$ for the problem P_k is determined as follows: $\mathcal{J}_k = B_k$, $k \in \{1, 2, \ldots, m\}$.

Using the $O(\log n)$-algorithm described in the proof of Lemma 4, we construct the largest optimality box $\mathcal{OB}(\pi^{s(k)}, T^k)$ for each problem P_k, $k \in \{1, 2, \ldots, m\}$, where $\pi^{s(k)}$ denotes a permutation of the jobs $\mathcal{J}_k = B_k$ and $T^k \subset T$.

It takes $O(|B| \log n(B))$ time to construct the optimality boxes $\mathcal{OB}(\pi^{s(k)}, T^k)$ for all problems P_k, $k \in \{1, 2, \ldots, m\}$, where $n(B) = \max\{|B_k| : k \in \{1, 2, \ldots, m\}\}$. Due to Lemma 5 and Theorem 5, the largest optimality box $\mathcal{OB}(\pi_k, T)$ for the original problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ is determined as the Cartesian product of the largest optimality boxes $\mathcal{OB}(\pi^{s(k)}, T^k)$ constructed for the problems P_k, where $k \in \{1, 2, \ldots, m\}$, i.e., $\mathcal{OB}(\pi_k, T) = \times_{k \in \{1,2,\ldots,m\}} \mathcal{OB}(\pi^{s(k)}, T^k)$. The permutation π_k with the largest optimality box $\mathcal{OB}(\pi_k, T)$ for the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ is determined as follows: $\pi_k = (\pi^{s(1)}, \pi^{s(2)}, \ldots, \pi^{s(m)})$. Using the complexities of the above algorithms, we conclude that the total complexity of the described algorithm can be estimated by $O(n \log n)$. ■

Theorem 6 allows one to find the largest optimality box $\mathcal{OB}(\pi_k, T)$ for the general case of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ via an enumeration of promising permutations based on the sets $B(\pi_k)$, where $\pi_k \in S$. To this end, one can construct all possible distributions of non-fixed jobs to their blocks. Then one can apply the algorithm described in the proof of Theorem 6 to each constructed problem $P^1, P^2, \ldots, P^{2^d}$, where each job from the set $\mathcal{J}$ belongs or distributed to one block. Then one can choose the largest optimality box from all constructed optimality

boxes. The total complexity of such an algorithm can be estimated by $O(2^d n \log n)$. If for the problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$, the growth of the number of non-fixed jobs is restricted by $O(\log n)$, then such an enumerative algorithm becomes polynomial with the complexity of $O(n^2 \log n)$.

Remark 2 *If $J_i \notin B_r \in B$, then job J_i must be located either on the left side from the core of the block B_r or on the right side from this core in any permutation π_k having the largest optimality box $\mathcal{OB}(\pi_k, T)$. Hence, job J_i must precede (or succeed, respectively) each job $J_v \in B_k$. Therefore, the optimal permutation π_t of the jobs from the block B_r obtained using the algorithm described in the proof of Lemma 4, where only jobs B_r are sequenced, remains the same if job $J_i \notin B_r \in B$ is added to the permutation π_t.*

Lemma 6 *Within constructing a permutation π_k with the largest optimality box $\mathcal{OB}(\pi_k, T)$, any job J_i may be moved only within the blocks containing this job.*

Proof. Let the job J_i be located in the block B_r in the permutation π_k such that $J_i \notin B_r$. Then for each job $J_v \in B_r$, either the inequality $\frac{w_v}{p_v^L} < \frac{w_i}{p_i^U}$ or the inequality $\frac{w_v}{p_v^U} > \frac{w_i}{p_i^L}$ must hold. If the inequality $\frac{w_v}{p_v^L} < \frac{w_i}{p_i^U}$ holds, then job J_u strongly dominates job J_i (due to Corollary 1). If the inequality $\frac{w_v}{p_v^U} > \frac{w_i}{p_i^L}$ holds, then job J_i strongly dominates job J_u (due to Corollary 1). Hence, if the job J_i is located in the permutation π_k between job $J_v \in B_r$ and job $J_w \in B_r$, then $\mathcal{OB}(\pi_k, T) = \emptyset$ (see Definition 2). ■

Corollary 2 *If job J_i is fixed in the block $B_k \in B$ (or is non-fixed but distributed to the block $B_k \in B$), then job J_i is located within the jobs from the block B_k in any permutation π_k with the largest optimality box $\mathcal{OB}(\pi_k, T)$ (provided that job J_i is fixed in the block $B_k \in B$ due to an appropriate restriction of the segment $[p_i^L, p_i^U]$).*

4 An algorithm for constructing a job permutation with the largest perimeter of the optimality box

Based on the properties of the optimality box proven in Subsections 3.2 and 3.3, we propose the following algorithm for constructing an optimal permutation for the problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$, i.e., such a permutation $\pi_k \in S$, whose optimality box $\mathcal{OB}(\pi_k, T)$ has the largest perimeter among all permutations of the set S.

Algorithm MAX-OPTBOX

Input: Segments $[p_i^L, p_i^U]$ and weights w_i given for all jobs $J_i \in \mathcal{J}$.
Output: Optimal permutation $\pi_k \in S$ and optimality box $\mathcal{OB}(\pi_k, T)$.
Step 1: **IF** condition of Theorem 4 holds
THEN $\mathcal{OB}(\pi_k, T) = \emptyset$ for any permutation $\pi_k \in S$ **STOP**
Step 2: **ELSE** determine the set B of blocks using the
$O(n \log n)$-algorithm described in the proof of Lemma 1
Step 3: Index the blocks $B = \{B_1.B_2, \ldots, B_m\}$ according to decreasing left bounds of their cores (see Lemma 2)
Step 4: **IF** $\mathcal{J} = B_1$ **THEN** problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ is called problem P_1, set $v = 1$ (see Lemma 4) **GOTO** step 6
Step 5: **IF** each pair of adjacent block subsets $\mathcal{B}_{r_i}^{t_i} = \{B_{r_i+1}, B_{r_i+2}, \ldots, B_{r_i+t_i}\}$

and $\mathcal{B}_{r_{i+1}}^{t_{i+1}} = \{B_{r_{i+1}+1}, B_{r_{i+1}+2}, \ldots, B_{r_{i+1}+t_{i+1}}\}$ from the set $\{\mathcal{B}_{r_1}^{t_1}, \mathcal{B}_{r_2}^{t_2}, \ldots, \mathcal{B}_{r_v}^{t_v}\}$ of the block subsets have no common job **THEN** decompose the problem $1|p_i^L \le p_i \le p_i^U|w_i \sum C_i$ into the subproblems $P_1, P_2, \ldots, P_v$ with the job sets $\mathcal{J}(\mathcal{B}_{r_1}^{t_1})$, $\mathcal{J}(\mathcal{B}_{r_2}^{t_2}), \ldots, \mathcal{J}(\mathcal{B}_{r_v}^{t_v})$ using Lemma 5 and Theorem 5 **GOTO** step 6
ELSE set $v = 1$

Step 6: **FOR** $i = 1$ to v **DO**
IF there exist non-fixed jobs in the set $\mathcal{J}(\mathcal{B}_{r_i}^{t_i})$ **THEN** construct an optimal permutation $\pi^{s(i)}$ for the problem P_i using Algorithm DM described in Subsection 4.1 **GOTO** step 8

Step 7: **ELSE** construct an optimal permutation $\pi^{s(i)}$ for the problem P_i using the $O(n \log n)$-algorithm described in the proof of Theorem 6

Step 8: Construct the optimality box $\mathcal{OB}(\pi^{s(i)}, T^i)$ for the permutation $\pi^{s(i)}$ using Algorithm FIX-OPTBOX **GOTO** step 6 with $i := i + 1 \le v$
END FOR

Step 9: Determine an optimal permutation $\pi_k = (\pi^{s(1)}, \pi^{s(2)}, \ldots, \pi^{s(v)}) \in S$ set $\mathcal{OB}(\pi_k, T) = \times_{i \in \{1,2,\ldots,v\}} \mathcal{OB}(\pi^{s(i)}, T^i)$ **STOP**.

4.1 Algorithm DM for the problem with non-fixed jobs in the blocks

For solving the problem P_i at step 6 of the Algorithm MAX-OPTBOX, we use Algorithm DM based on dynamic programming. Algorithm DM allows us to construct an optimal permutation $\pi_k \in S$ for the problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$, where the job set $\mathcal{J}$ consists of more than one block, $m \ge 2$, and neither the condition of Lemma 5 nor the condition of Theorem 5 holds for the set $\mathcal{J} = \{J_1, J_2, \ldots, J_n\} = \mathcal{J}(\mathcal{B}_m^0)$ of jobs, where $\mathcal{B}_m^0$ denotes the following set of blocks: $\{B_1, B_2, \ldots, B_m\} = \mathcal{B}_m^0$.

For the problem $1|p_i^L \le p_i \le p_i^U| \sum w_i C_i$ with $\mathcal{J} = \{J_1, J_2, \ldots, J_n\} = \mathcal{J}(\mathcal{B}_m^0)$, one can calculate a tight upper bound Per_{max} on the perimeter $Per\mathcal{OB}(\pi_k, T)$ of the optimality box $\mathcal{OB}(\pi_k, T)$ determined as follows:

$$Per_{max} = \max_{i=1}^{n} p_i^U - \min_{i=1}^{n} p_i^L - \sum_{B_k \in B \setminus \underline{B}} (b_k^U - b_k^L) \ge Per\mathcal{OB}(\pi_k, T), \tag{10}$$

where $\underline{B}$ denotes the set of all blocks $B_i \in B$ which are singletons: $|B_i| = 1$. Indeed it is clear that the optimality segment of any job $J_i \in \mathcal{J}$ cannot include the core $[b_k^L, b_k^U]$ of any block $B_k \in B \setminus \underline{B}$. Therefore, the sum of the lengths of the optimality segments for all jobs $J_i \in \mathcal{J}$ cannot be greater than the difference between the length $\max_{i=1}^n p_i^U - \min_{i=1}^n p_i^L$ of the segment $[\min_{i=1}^n p_i^L, \max_{i=1}^n p_i^U]$ and the sum $\sum_{k=1}^m (b_k^U - b_k^L)$ of the lengths of the cores for all blocks B_k from the set $B \setminus \underline{B}$.

Remind that for Example 1 discussed in Subsections 3.2 and 3.3, the upper bound Per_{max} determined in (10) is reached for the permutation $\pi_k = (J_1, J_2, J_4, J_3, J_5) \in S$, namely, $Per_{max} = Per^1 + Per^2 = 8 + 12 = 20$. In Fig. 2, the maximal optimality segment of length Per^1 and that of length Per^2 are indicated in bold fashion on the abscissa axis.

Instead of describing Algorithm DM in a formal way, we describe two iterations of Algorithm DM along with the application of Algorithm DM to a small Example 2 with three blocks and two non-fixed jobs in these blocks (see Subsection 4.2).

Iterations of Algorithm DM and notations

Let $\mathcal{T} = (V, E)$ denote the solution tree constructed by Algorithm DM at the last iteration, where V is the set of vertexes and E is the set of edges. The subgraph of the solution tree $\mathcal{T} = (V, E)$ constructed at the iteration h is denoted by $\mathcal{T}_h = (V_h, E_h)$. All vertexes $i \in V$ of the solution tree have their ranks from the set $\{0, 1, \ldots, m = |B|\}$. The single vertex 0 in the tree $\mathcal{T}_h = (V_h, E_h)$ has a zero rank. The vertex 0 is characterized by a partial job permutation $\pi^0(\emptyset; \emptyset)$, where the non-fixed jobs are not distributed to their blocks. All vertexes of the tree $\mathcal{T}_h$ having the first rank are generated at iteration 1 from vertex 0 via distributing the non-fixed jobs $\mathcal{J}[B_1]$ of the block B_1, where $\mathcal{J}[B_1] \subseteq B_1$. Each job $J_v \in \mathcal{J}[B_1]$ must be distributed either to block B_1 or to another block $B_j \in B$ with the inclusion $J_v \in \mathcal{J}[B_j]$. Let B_k^t denote the set of all non-fixed jobs $J_i \in B_k$, which are not distributed to their blocks at the iterations with numbers less than t.

A partial permutation of the jobs is characterized by the notation $\pi^u(B_k; \mathcal{J}_{[u]})$, where u denotes vertex $u \in V_t$ in the constructed solution tree $\mathcal{T}_t = (V_t, E_t)$ and $\mathcal{J}_{[u]}$ denotes the non-fixed jobs from the set $\mathcal{J}[B_k]$, which are distributed to the block B_k in the vertex $j \in V_t$ of the solution tree such that arc (j, u) belongs to set E_t.

At the **first iteration** of Algorithm DM, the set $\mathcal{P}(\mathcal{J}[B_1^1]) = \{\mathcal{J}[B_1^1], \ldots, \emptyset\}$ of all subsets of the set $\mathcal{J}[B_1^1]$ is constructed and $2^{|\mathcal{J}[B_1^1]|}$ partial permutations are generated for all subsets $\mathcal{P}(\mathcal{J}[B_1^1])$ of the non-fixed jobs $\mathcal{J}[B_1^1]$. As a result, the constructed solution tree $\mathcal{T}_1 = (V_1, E_1)$ consists of the vertexes $0, 1, \ldots, 2^{|\mathcal{J}[B_1^1]|}$ and $2^{|\mathcal{J}[B_1^1]|}$ arcs connecting vertex 0 with the other vertexes in the tree $\mathcal{T}_1$.

For each generated permutation $\pi^u(B_k; \mathcal{J}_{[u]})$, where $\mathcal{J}_{[u]} \in \mathcal{P}(\mathcal{J}[B_k^t])$, a penalty ϕ_u is calculated, which determines the difference between the maximal possible perimeter Per_{max} of the optimality box $\mathcal{OB}(\pi_k, T)$ and the perimeter of the optimality box $\mathcal{OB}(\pi^u(B_k; \mathcal{J}_{[u]}), T)$, which may be constructed for the permutation $\pi^u(B_k; \mathcal{J}_{[u]})$:

$$Per\mathcal{OB}(\pi^u(B_k; \mathcal{J}_{[u]}), T) = \sum_{i \le k} Per^i - \phi_u, \tag{11}$$

where Per^i denotes the maximal optimality segment of the first ordered job from the block B_i. The penalty ϕ_u is calculated using the $O(\log n)$-algorithm from the proof of Lemma 4. The complete permutation $\pi^{0 \to u}(B_k; \mathcal{J}_{[u]})$ with the end $\pi^u(B_k; \mathcal{J}_{[u]})$ is determined based on the permutations $\pi^s(B_c; \mathcal{J}_{[s]})$, where each vertex s belongs to the chain $(0 \to u)$ between vertex 0 and vertex u in the solution tree $\mathcal{T}_t = (V_t, E_t)$ and each block B_c belongs to the set $B^u = \{B_1, B_2, \ldots, B_k\}$. The permutation $\pi^{0 \to u}(B_k; \mathcal{J}_{[u]})$ includes all jobs, which are fixed or distributed in the blocks $B_c \in B^u$ in optimal order providing the penalty ϕ_u. The aim of Algorithm DM is to construct a complete job permutation $\pi^f(B_m; \mathcal{J}_{[f]}) = \pi_k \in S$ such that the penalty ϕ_f is minimal for all job permutations from the set S. For consideration at the next iteration, a partial permutation $\pi^s(B_2; \mathcal{J}_{[s]})$ is chosen from the constructed solution tree such that the penalty ϕ_s is minimal among the permutations corresponding to the leafs of the solution tree.

At the **second iteration**, the set $\mathcal{P}(\mathcal{J}[B_2^2]) = \{\mathcal{J}[B_2^2], \ldots, \emptyset\}$ of all subsets of the set $\mathcal{J}[B_2^2]$ is generated and $2^{|\mathcal{J}[B_2^2]|}$ permutations for all subsets $\mathcal{P}(\mathcal{J}[B_2^2])$ of the jobs from the block B_2 are constructed. For each generated permutation $\pi^v(B_2^2; \mathcal{J}_{[v]})$, where $\mathcal{J}_{[v]} \in \mathcal{P}(\mathcal{J}[B_2^2])$, the penalty ϕ_v is calculated using the equality $\phi_v = \phi_l + \Delta\phi_k$, where the arc (l, v) belongs to the solution tree $\mathcal{T}_2 = (V_2, E_2)$ and $\Delta\phi_k$ denotes the penalty reached for the optimal permutation $\pi^{0 \to v}(B_2; \mathcal{J}_{[v]})$ constructed from the permutation $\pi^v(B_1; \mathcal{J}_{[v]})$ using the $O(\log n)$-algorithm described in the proof of Lemma 4.

For consideration at the next iteration, one chooses the partial permutation $\pi^d(B_c; \mathcal{J}_{[d]})$ with the minimal value of the penalty for the partial permutations in the leaves of the solution tree. The whole solution tree is constructed similarly until there is a partial permutation with a smaller value of the penalty ϕ_t in the constructed solution tree.

4.2 Example 2 of the problem $1|p_i^L \le p_i \le p_i^U|\sum C_i$

For simplicity of the verification of the solution by hand, we apply Algorithm DM to the special case $1|p_i^L \le p_i \le p_i^U|\sum C_i$ of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_i C_i$, where all jobs $J_i \in \mathcal{J}$ have equal weights: $w_i = 1$. The input data for Example 2 of the problem $1|p_i^L \le p_i \le p_i^U|\sum C_i$ are given in Table 1.

Table 1: Input data for Example 2 of the problem $1|p_i^L \le p_i \le p_i^U|\sum C_i$

i	p_i^L	p_i^U	$\frac{w_i}{p_i^U}$	$\frac{w_i}{p_i^L}$
1	3	7	1/7	1/3
2	5	9	1/9	1/5
3	3	19	1/19	1/3
4	14	16	1/16	1/14
5	11	17	1/17	1/11
6	13	15	1/15	1/13
7	13	18	1/18	1/13
8	2	29	1/29	1/2
9	22	26	1/26	1/22
10	21	24	1/24	1/21

The given segments of the feasible job processing times are presented in a coordinate system in Fig. 3, where the abscissa axis is used for indicating the given segments for job processing times and the ordinate axis for the jobs from the set $\mathcal{J}$. The jobs J_1, J_2 and J_3 belong to the block B_1. The jobs J_3, J_4, J_5, J_6, J_7 and J_8 belong to the block B_2. The jobs J_8, J_9 and J_{10} belong to the block B_3. The jobs J_3 and J_8 are non-fixed and $\mathcal{J}[B_1] = \{J_3, J_8\} = \mathcal{J}[B_2]$, $\mathcal{J}[B_3] = \{J_8\}$. The job J_3 must be distributed either to the block B_1 or to the block B_2. The job J_8 must be distributed either to the block B_1, B_2 or to the block B_3 depending on the constructed permutation $\pi_k \in S$ of the jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_{10}\}$. The maximal optimality segment $[l_{k_i}^*, u_{k_i}^*]$ of the jobs are as follows: $[l_{k_1}^*, u_{k_1}^*] = [2, 5]$, $[l_{k_z}^*, u_{k_z}^*] = [7, 14]$, $[l_{k_v}^*, u_{k_v}^*] = [15, 22]$, $[l_{k_6}^*, u_{k_6}^*] = [24, 29]$. Therefore, the perimeter of the optimality box $\mathcal{OB}(\pi_k, T)$ for any job permutation $\pi_k \in S$ cannot be greater than Per_{max} = (5-2)+(14-7)+(22-15)+(29-24)= 22 due to the upper bound on the perimeter $Per\mathcal{OB}(\pi_k, T)$ of the optimality box $\mathcal{OB}(\pi_k, T)$ given in (10).

The solution tree $\mathcal{T} = (V, E)$ obtained by the application of Algorithm DM to Example 2 is represented in Fig. 4. The search process is started with the solution tree defined as follows: $\mathcal{T}_0 = (V_0, E_0) = (\pi^0(\emptyset; \emptyset), \emptyset)$ with a zero penalty $\phi_0 = 0$.

At the first iteration, the set $\mathcal{P}(\mathcal{J}[B_1^1]) = \{\emptyset, \{J_3\}, \{J_8\}, \{J_3, J_8\}\}$ is constructed and permutations $\pi^1(B_1^1; \emptyset), \pi^2(B_1^1; J_3), \pi^3(B_1^1; J_8), \pi^4(B_1^1; J_3, J_8)$ are generated. We calculate the reduced segments $[\widetilde{p}_j^L, \widetilde{p}_j^U] = [\max\{p_j^L, b_{i-1}^U\}, \min\{p_j^U, b_{i+1}^L\}]$ of the job processing times for the non-fixed jobs by distributing them to block B_1. For each element of the set $\mathcal{P}(\mathcal{J}[B_1^1])$, we construct a permutation with the maximal perimeter of the optimality box and calculate the penalty. For block B_1, we obtain $[\widetilde{p}_3^L, \widetilde{p}_3^U] = [3, 14]$ and $[\widetilde{p}_8^L, \widetilde{p}_8^U] = [2, 14]$. We obtain the following

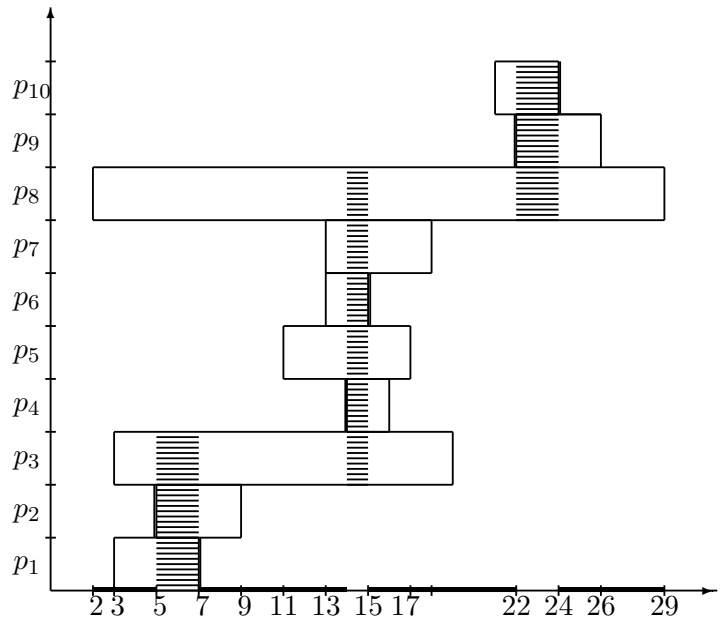

Figure 3: The given segments $[p_i^L, p_i^U]$ for the feasible processing times of the jobs $J_i \in \mathcal{J} = \{J_1, J_2, \ldots, J_{10}\}$ (the cores of the blocks B_1, B_1 and B_3 are dashed).

permutations with their perimeters:

$\pi^{0\to1}(B_1^1;\emptyset) = (J_1, J_2)$, $\phi_1 = 1$; $\pi^{0\to2}(B_1^1; J_3) = (J_1, J_2, J_3)$, $\phi_2 = 1$;

$\pi^{0\to3}(B_1^1; J_8) = (J_1, J_2, J_8)$, $\phi_3 = 1$; $\pi^{0\to4}(B_1^1; J_3, J_8) = (J_8, J_2, J_1, J_3)$, $\phi_4 = 0$.

Since all non-fixed jobs are distributed in the permutation $\pi^{0\to4}(B_1^1; J_3, J_8)$, we obtain the complete permutation $(J_8, J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{10}, J_9) \in S$ with the penalty equal to 6.

At the second iteration, we obtain the reduced segments $[\widetilde{p}_3^L, \widetilde{p}_3^U] = [7, 19]$, $[\widetilde{p}_8^L, \widetilde{p}_8^U] = [7, 22]$. We construct the permutations $\pi^5(B_2^2; J_3)$ and $\pi^6(B_2^2; J_3, J_8)$ from the permutation $\pi^1(B_1^1, \emptyset)$, As a result, we obtain the permutation $\pi^{0\to5}(B_2^2; J_3) = (J_1, J_2, J_3, J_4, J_5, J_6, J_7)$ with the penalty $\phi_5 = 3$ and permutation $\pi^{0\to6}(B_2^2; J_3, J_8) = (J_1, J_2, J_3, J_4, J_5, J_7, J_6, J_8)$ with the penalty $\phi_6 = 3$. Since all non-fixed jobs are distributed in the permutation $\pi^{0\to6}(B_2^2; J_3, J_8)$ we obtain the complete permutation $(J_1, J_2, J_3, J_4, J_5, J_7, J_6, J_8, J_{10}, J_9) \in S$ with penalty equal to 7.

Similarly, we obtain the following permutations with their penalties:

$\pi^{0\to7}(B_2^3;\emptyset) = (J_1, J_2, J_3, J_4, J_5, J_6, J_7)$, $\phi_7 = 3$;

$\pi^{0\to8}(B_2^3; J_8) = (J_1, J_2, J_3, J_4, J_5, J_7, J_6, J_8)$, $\phi_8 = 3$;

$\pi^{0\to9}(B_2^4; J_3) = (J_8, J_2, J_1, J_3, J_4, J_5, J_6, J_7)$, $\phi_9 = 0$;

$\pi^{0\to10}(B_3^5; J_8) = (J_1, J_2, J_5, J_4, J_7, J_6, J_3, J_9, J_{10}, J_8)$, $\phi_{10} = 6$;

$\pi^{0\to11}(B_3^6; J_8) = (J_1, J_2, J_3, J_4, J_5, J_7, J_6, J_8, J_{10}, J_9)$, $\phi_{11} = 7$.

Since all non-fixed jobs are distributed in the permutations $\pi^{0\to8}(B_2^3; J_8)$ and $\pi^{0\to9}(B_2^4; J_3)$ we obtain the complete permutations

$(J_1, J_2, J_3, J_4, J_5, J_7, J_6, J_8, J_{10}, J_9) \in S$ and

$(J_8, J_2, J_1, J_3, J_4, J_5, J_6, J_7, J_{10}, J_9) \in S$ with penalties equal to 7 and 6.

Using Algorithm DM for Example 2, we obtain the following two permutations with the largest perimeter of the optimality box: $\pi^{0\to4}(B_1^1; J_3, J_8)$ and $\pi^{0\to10}(B_3^5; J_8)$. This maximal perimeter $Per(\mathcal{OB}(\pi_k, T))$ of the optimality box is equal to $22 - 6 = 16$, where $Per_{max} = 22$ and the minimal penalty obtained for the permutation π_k is equal to 6.

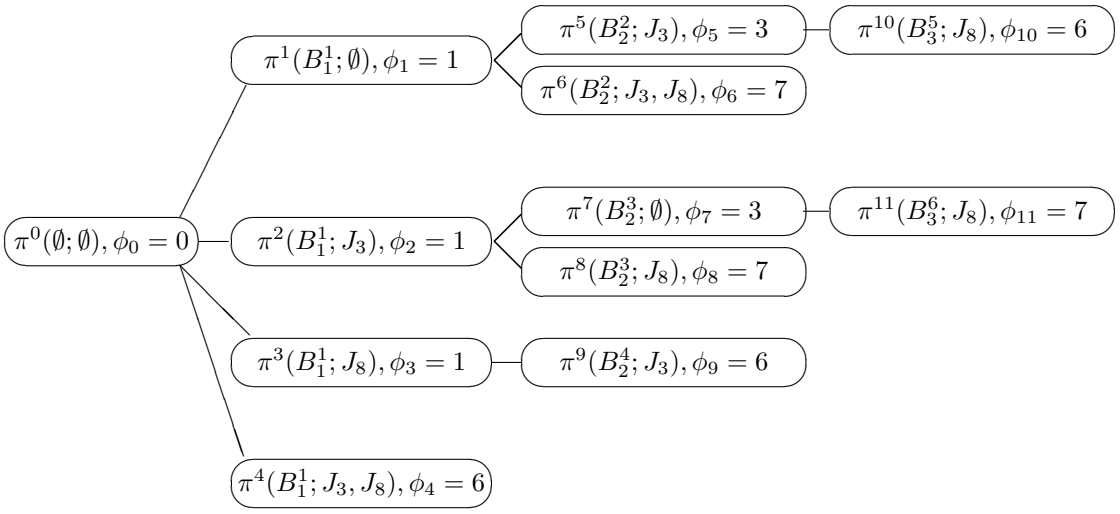

Figure 4: Solution tree $\mathcal{T}=(V,E)$ constructed for Example 2.

5 Computational results

Note that in the published papers [Allahverdi and Aydilek (2010), Sotskov, Egorova, and Lai (2009), Sotskov and Lai (2012)], computational results were presented for randomly generated instances $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ when there were no properties of the randomly generated jobs in the set $\mathcal{J}$, which make the problem harder. For such instances, the mid-point permutation $\pi_e \in S$, where all jobs $J_i \in \mathcal{J}$ are ordered according to decreasing mid-points $\frac{p_i^U-p_i^L}{2}$ of their segments $[p_i^L,p_i^U]$, is close to an optimal permutation. In our experiments, we tested a class of harder problems, where the permutation constructed by Algorithm DM outperforms the mid-point permutation.

Table 2 presents computational results for randomly generated instances of the problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ with $n \in \{10,20,30,50,100\}$. We randomly generated a single non-fixed job J_1, which belongs to all three blocks B_1, B_2, and B_3 of the randomly generated $n-1$ fixed jobs. The lower bound p_i^L and the upper bound p_i^U on the feasible values $p_i \in R_+^1$ of the processing times of the fixed jobs, $p_i \in [p_i^L,p_i^U]$, were generated as follows. We determined a bound of blocks $[\widetilde{b}_i^L,\widetilde{b}_i^U]$ for generating the cores of the blocks $[b_i^L,b_i^U] \subset [\widetilde{b}_i^L,\widetilde{b}_i^U]$ and for generating the segments $[p_i^L,p_i^U]$ for the processing times of $|B_i|$ jobs for all blocks B_i, $1 \le i \le 3$, $[b_i^L,b_i^U] \subset [p_i^L,p_i^U] \subset [\widetilde{b}_i^L,\widetilde{b}_i^U]$. The weights w_i are assumed to be unknown before scheduling. We solved 28 series of randomly generated instances. Each series contains 25 instances with the same combination of the job number n and the numbers $|B_i|$ of the jobs in the blocks B_i, $1 \le i \le 3$.

In the experiments, we answered the question of how large the obtained relative error Δ of the value $\gamma_{p^*}^k$ of the objective function $\gamma=\sum_{i=1}^n w_iC_i$ was for the permutation π_k with the largest perimeter of the optimality box $\mathcal{OB}(\pi_k,T)$ with respect to the actually optimal objective function value $\gamma_{p^*}^t$ calculated for the actual processing times $p^*=(p_1^*,p_2^*,\ldots,p_n^*) \in T$: $\Delta=\frac{\gamma_{p^*}^k-\gamma_{p^*}^t}{\gamma_{p^*}^t}\cdot 100\%$. We also answered the question of how less the obtained relative error Δ of the value $\gamma_{p^*}^k$ of the objective function γ was for the permutation π_k with the largest perimeter

of a optimality box $\mathcal{OB}(\pi_k, T)$ if compared with the relative error Δ_{mid-p} of the value $\gamma^m_{p^*}$ of the objective function γ for the permutation π_{mid-p} obtained for determining the job processing times using the mid-points of the given segments.

The number n of jobs in an instance is given in column 1 of Table 2. The numbers of jobs (and its percentage from the total job number) in each block are given in column 2, column 3 and column 4. Columns 5 and 8 present the weight of the non-fixed job J_1. Columns 7 and 10 present the average error Δ for the permutation π_k with the largest perimeter of the optimality box $\mathcal{OB}(\pi_k, T)$. Columns 6 and 9 present the average error Δ_{mid-p} obtained for the mid-point permutations, where all jobs are ordered according to decreasing mid-points of their segments. Columns 5 – 7 present examples with the equal weights of the fixed jobs: $w_i = 1$. Columns 8 – 10 present examples with the different weights of the fixed jobs, which are randomly generated on the segment $[1, 5]$: $w_i \in [1, 5]$.

Our computational experiments show that for all solved examples, the permutations π_k with the largest perimeter of their optimality boxes generated good objective function values $\gamma^k_{p^*}$, which are smaller than those obtained for the permutations π_{mid-p}. If the weight w_i increases, then the difference of the errors Δ and Δ_{mid-p} increases. The largest errors and the smallest errors are presented in Table 2 in bold fashion. The average errors for all tested series of the examples are presented in the last row of Table 2.

Table 2: Computational results for randomly generated instances.

n	$\lvert B_1\rvert$	$\lvert B_2\rvert$(%)	$\lvert B_3\rvert$(%)	w_1	Δ_{mid-p}	Δ	w_1	Δ_{mid-p}	Δ
1	2	3	4	5	6	7	8	9	10
10	7	1(10%)	1(10%)	1	6.687789	4.54844	5	8.756675	3.886302
10	7	1(10%)	1(10%)	10	6.872457	1.418286	10	9.258109	4.798429
10	6	1(10%)	2(20%)	1	3.862066	3.079768	5	3.536148	3.276581
10	6	1(10%)	2(20%)	10	5.838515	2.736017	10	4.82569	3.427305
10	5	1(10%)	3(30%)	1	3.491078	2.916041	5	3.297388	3.060939
10	5	1(10%)	3(30%)	10	6.575227	2.337764	10	2.030337	**1.390253**
10	4	1(10%)	4(40%)	1	2.500799	2.301744	5	3.038637	2.821627
10	4	1(10%)	4(40%)	10	5.238737	2.248937	10	4.065063	2.458807
20	15	2(10%)	2(10%)	1	4.827593	2.050505	5	6.516302	5.625189
20	15	2(10%)	2(10%)	10	12.50855	2.290443	10	7.798122	6.591335
20	13	2(10%)	4(20%)	1	3.007268	2.20674	5	4.320849	4.102687
20	13	2(10%)	4(20%)	10	11.92242	3.111579	10	5.796945	3.731351
20	11	2(10%)	6(30%)	1	2.078547	1.820955	5	2.893188	2.364526
20	11	2(10%)	6(30%)	10	7.360498	3.364394	10	5.609517	3.305518
20	9	2(10%)	8(40%)	1	**1.41869**	**1.226285**	5	2.958016	2.857993
20	9	2(10%)	8(40%)	10	6.562489	2.721417	10	2.95476	1.920063
30	23	3(10%)	3(10%)	1	4.68668	4.658469	5	7.173906	**6.670326**
30	23	3(10%)	3(10%)	10	**14.67459**	4.20408	10	7.522324	6.514499
30	20	3(10%)	6(20%)	1	1.956536	1.891303	5	4.045457	3.091913
30	20	3(10%)	6(20%)	10	11.69048	2.700793	10	4.815396	4.711968
50	39	5(10%)	5(10%)	1	4.054842	3.206457	5	4.998765	4.641194
50	39	5(10%)	5(10%)	10	14.34459	5.541194	10	**9.474605**	4.726287
50	34	5(10%)	10(20%)	1	1.519173	1.399192	5	1.89226	1.560108
50	34	5(10%)	10(20%)	10	7.431458	**6.250765**	10	5.11061	3.123643
100	79	10(10%)	10(10%)	1	2.65005	1.922146	5	3.370896	2.897564
100	79	10(10%)	10(10%)	10	9.051175	4.640042	10	5.122501	4.563323
100	69	10(10%)	20(20%)	1	1.869466	1.80168	5	**1.774118**	1.53655
100	69	10(10%)	10(20%)	10	7.190554	5.998655	10	2.60963	2.440866
					6.138297	**3.021218**		**4.841651**	**3.646327**

6 Concluding remarks

While an optimal sequencing rule for the deterministic problem $1||\sum w_iC_i$ has been known since 1956 due to [Smith (1956)], its uncertain version continues to attract the attention of the OR researchers since problem $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$ is commonly applicable in many multiple-resource scheduling systems, where only one of the resources is the bottleneck and uncertain.

We introduced the notion of an optimality box of a job permutation and derived an algorithm for constructing the optimality box for a job permutation $\pi_k \in S$, which runs in $O(n \log n)$ time. We presented useful properties of the optimality box as well as computational results for finding a permutation with the largest perimeter of the optimality box. By the computational results, it is shown that the permutation with the largest perimeter of the optimality box is close to an optimal permutation, which can be determined after completing the jobs when their processing times became known.

For future work, we suggest to develop algorithms for finding a job permutation with the largest volume of the optimality box. Such algorithms will be compared with the best known algorithms developed for the hard problems $1|p_i^L \le p_i \le p_i^U|\sum w_iC_i$.

The proven properties of the optimality boxes may be also used to increase the efficiency and effectiveness of the algorithms available for solving the robust (minmax regret) uncertain single machine scheduling problems. The proven properties of the optimality box and the strong dominance relation on the set of jobs may be used to improve the robust approach for single-machine scheduling as it was done in [Pereira (2016)] by using the dominance relation determined on the job set.

The results that we obtained may be generalized to the problem $1|prec, p_i^L \le p_i \le p_i^U|\sum w_iC_i$, where precedence constraints are given among the set of jobs. If the deterministic problem $1|prec|\sum w_iC_i$ is polynomially solvable for a particular type of precedence constraints (e.g., it is defined by an in-tree, an out-tree or a series-parallel graph), then the above results may be used for the uncertain problem $1|prec, p_i^L \le p_i \le p_i^U|\sum w_iC_i$. The developed approach may be applied to single-machine problems $1|prec, p_i^L \le p_i \le p_i^U|\gamma$ with other criteria like $\gamma = L_{max}$.

Acknowledgement

The first and second authors acknowledge the research grant from the Ministry of Science and Technology, Taiwan.

References

[Allahverdi and Aydilek (2010)] Allahverdi, A., and H. Aydilek. 2010. "Heuristics for two-machine flowshop scheduling problem to minimize makespan with bounded processing times." *International Journal of Production Research* 48: 6367–6385.

[Allahverdi, Aydilek, and Aydilek (2014)] Allahverdi, A., H. Aydilek, and A. Aydilek. 2014. "Single machine scheduling problem with interval processing times to minimize mean weighted completion times." *Computers & Operations Research* 51: 200–207.

[Coffman (1976)] Coffman, E.J. 1976. *Computer and Job-Shop Scheduling Theory.* John Willey & Sons, Hauppauge, New York, USA.

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

[Goren and Sabuncuoglu (2008)] Goren, S., and I. Sabuncuoglu. 2008. "Robustness and stability measures for scheduling: single-machine environment." *IIE Transactions* 40: 66–83.

[Graham et al. (1979)] Graham, R.L., E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnoy Kan. 1979. "Optimization and Approximation in Deterministic Sequencing and Scheduling." *Annals of Discrete Applied Mathematics* 5: 287–326.

[Harikrishnan and Ishii (2005)] Harikrishnan, K.K., and H. Ishii. 2005. "Single machine batch scheduling problem with resource dependent setup and processing time in the presence of fuzzy due date." *Fuzzy Optimization and Decision Making* 4: 141–147.

[Herroelen and Leus (2004)] Herroelen, W., and R. Leus. 2004. "Robust and reactive project scheduling: a review and classification procedures." *International Journal of Production Research* 42 (8): 1599–1620.

[Kasperski and Zielinski (2008)] Kasperski, A., and P. Zielinski. 2008. "A 2-approximation algorithm for interval data minmax regret sequencing problems with total flow time criterion." *Operations Research Letters* 36: 343–344.

[Kasperski and Zielinski(2011)] Kasperski, A., and P. Zielinski. 2011. "Possibilistic minmax regret sequencing problems with fuzzy parameteres." *IEEE Transactions on Fuzzy Systems* 19 (6): 1072–1082.

[Kouvelis, Daniels, and Vairaktarakis (2000)] Kouvelis, P., R.L. Daniels, and G. Vairaktarakis. 2000. "Robust scheduling of a two-machine flow shop with uncertain processing times." *IIE Transactions* 32: 421–432.

[Kouvelis and Yu (1997)] Kouvelis, P., and G. Yu. 1997. *Robust Discrete Optimization and its Application*. Kluwer Academic Publishers, Boston, The USA.

[Lu, Lin, and Ying (2012)] Lu, C.-C., S.-W. Lin, and K.-C. Ying. 2012. "Robust scheduling on a single machine total flow time." *Computers & Operations Research* 39: 1682–1691.

[Özelkan and Duckstein (1999)] Özelkan, E.C., and L. Duckstein. 1999. "Optimal fuzzy counterparts of scheduling rules." *European Journal of Operational Research* 113: 593–609.

[Pereira (2016)] Pereira, J. 2016. "The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective." *Computers & Operations Research* 66: 141–152.

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

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

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

[Sotskov, Egorova, and Lai (2009)] Sotskov, Yu.N., N.M. Egorova, and T.-C. Lai. 2009. "Minimizing total weighted flow time of a set of jobs with interval processing times." *Mathematical and Computer Modelling* 50: 556–573.

[Sotskov and Lai (2012)] Sotskov, Yu.N., and T.-C. Lai. 2012. "Minimizing total weighted flow time under uncertainty using dominance and a stability box." *Computers & Operations Research* 39: 1271–1289.

[Sotskov and Werner (2014)] Sotskov, Yu.N., and F. Werner. 2014. *Sequencing and Scheduling with Inaccurate Data*. Nova Science Publishers, Hauppauge, New York, USA.

[Tadayon and Smith (2015)] Tadayon, B., and J.C. Smith. 2015. "Algorithms and complexity analysis for robust single-machine scheduling problems." *Journal of Scheduling* 18 (6): 575–592.

[Wu (2010)] Wu, H.-C. 2010. "Solving the fuzzy earliness and tardiness in scheduling problems by using genetic algorithms." *Expert Systems with Applications* 37 (7): 4860–4866.

[Yang and Yu (2002)] Yang, J., and G. Yu. 2002. "On the robust single machine scheduling problem." *Journal of Combinatorial Optimization* 6: 17–33.