DOI: https://doi.org/10.20535/kpisn.2021.2.227037
UDC 519.161+519.687.1+004.023

V.V. Romanuke*
Polish Naval Academy, Gdynia, Poland
*corresponding author: romanukevadimv@gmail.com

# SORTING APPROACHES IN THE HEURISTIC BASED ON REMAINING AVAILABLE AND PROCESSING PERIODS TO MINIMIZE TOTAL WEIGHTED TARDINESS IN PROGRESSIVE IDLING-FREE 1-MACHINE PREEMPTIVE SCHEDULING

**Background.** In preemptive job scheduling, total weighted tardiness minimization is commonly reduced to solving a combinatorial problem, which becomes practically intractable as the number of jobs and the numbers of their processing periods increase. To cope with this challenge, heuristics are used. A heuristic, in which the decisive ratio is the weighted reciprocal of the maximum of a pair of the remaining processing period and remaining available period, is closely the best one. However, the heuristic may produce schedules of a few jobs whose total weighted tardiness is enormously huge compared to the real minimum. Therefore, this heuristic needs further improvements, one of which already exists for jobs without priority weights with a sorting approach where remaining processing periods are minimized. Three other sorting approaches still can outperform it, but such exceptions are quite rare.
**Objective.** The goal is to determine the influence of the four sorting approaches and try to select the best one in the case where jobs have their priority weights. The heuristic will be applied to tight-tardy progressive idling-free 1-machine preemptive scheduling, where the release dates are given in ascending order starting from 1 to the number of jobs, and the due dates are tightly set after the release dates.
**Methods.** To achieve the said goal, a computational study is carried out with applying each of the four heuristic approaches to minimize total weighted tardiness. For this, two series of 4151500 scheduling problems are generated. In the solution of a scheduling problem, a sorting approach can "win" solely or "win" in a group of approaches, producing the heuristically minimal total weighted tardiness. In each series, the distributions of sole-and-group "wins" are ascertained.
**Results.** The sole "wins" and non-whole-group "wins" are rare: the four sorting approaches produce schedules with the same total weighted tardiness in over 98.39 % of scheduling problems. Although the influence of these approaches is different, it is therefore not really significant. Each of the sorting approaches has heavy disadvantages leading sometimes to gigantic inaccuracies, although they occur rarely. When the inaccuracy occurs to be more than 30 %, this implies that 3 to 9 jobs are scheduled.
**Conclusions.** Unlike the case when jobs do not have their priority weights, it is impossible to select the best sorting approach for the case with job priority weights. Instead, a hyper-heuristic comprising the sorting approaches (i. e., the whole group, where each sorting is applied) may be constructed. If a parallelization can be used to process two or even four sorting routines simultaneously, the computation time will not be significantly affected.
**Keywords:** preemptive 1-machine job scheduling; total weighted tardiness; heuristic; sorting approach; remaining processing periods; remaining available periods.

## Introduction

Minimization of total weighted tardiness (TWT) in job schedules is an important task which aims at reducing costs of production delays [1]. This task is commonly reduced to solving a combinatorial problem which becomes practically intractable as volumes of jobs increase (i. e., as the number of jobs and/or the numbers of their processing periods increase) [2, 3]. The tractability can be slightly stretched and strengthened by using an optimal substitute for infinity in respective integer linear programming models [4, 5] and re-arranging jobs for either job ascending order input or job descending order input [6, 7]. After solving long series of thousands of job scheduling problems, these methods can really decrease computation time on average, but the impact on the tractability is too tiny [8].

The tractability problem is resolved by applying heuristics which allow finding schedules whose TWT is approximately minimal [9, 10]. However, the heuristically minimal TWT is not always the exact one. So, the tradeoff here is accuracy versus speed (i. e., computation time). A heuristic, in which the decisive ratio is the weighted reciprocal of the maximum of a pair of the remaining processing period (RPP) and remaining available period (RAP) [3], is closely the best one [11]. The accuracy of this heuristic (henceforward, let it be named the RPP-RAP heuristic) was studied in [3] on a pattern of tight-tardy progressive idling-free single machine preemptive (TPIF1MP) scheduling [8]. In general, the RPP-RAP heuristic produces about 92 % schedules [3]

whose TWT is exactly minimal, i. e. an integer linear programming model in about 92 % of the cases is needless. Besides, the heuristic schedules 2 jobs always with the exactly minimal TWT. However, the RPP-RAP heuristic may produce schedules of a few jobs whose TWT is enormously huge compared to the real minimum of TWT. For instance, a problem of scheduling 4 jobs divided into 6, 4, 5, and 5 processing periods with their respective priority weights 10, 4, 9, 8, whose due dates are 11, 5, 15, 18 by the progressive release dates 1, 2, 3, 4, respectively, has the TWT minimum of 16, whereas the RPP-RAP heuristic produces a schedule whose TWT is 36 (i. e., the relative gap [2, 3, 12] here is 125 %, which is obviously unacceptable). Therefore, this heuristic needs further improvements. One of such improvements is suggested in article [13] which considers minimization of total tardiness (i. e., without job priority weights) in TPIF1MP scheduling, where the release dates are given in ascending order starting from 1 to the number of jobs, and the due dates are tightly set after the release dates. Whereas the RPP-RAP heuristic sorts maximal decisive ratios by release dates, where the scheduling preference is given to the earliest job (this is the earliest-job sorting), three other sorting approaches are presented in [13]. In accordance with article [13], the earliest-job sorting ensures a heuristically minimal total tardiness in more than 97.6 % of scheduling problems, but it fails to minimize total tardiness in no less than 2.2 % of the cases. Nevertheless, a sorting approach with minimizing remaining processing periods produces a heuristically minimal total tardiness for almost any scheduling problem. If an exception occurs, this sorting approach "loses" to the other sorting approaches very little. Moreover, the exceptions are quite rare as just a one scheduling problem was registered (out of 31914 cases followed by a sole "win" of a heuristic version) whose minimal total tardiness was achieved by the earliest-job sorting.

**Problem statement**

In order to improve the RPP-RAP heuristic, the goal is to determine the influence of the four sorting approaches and try to select the best one. For this, a few series of TPIF1MP scheduling problems will be generated to minimize TWT, in which the distribution of sole "wins" is to be ascertained. The distribution of group "wins" is to be ascertained also. The possibility of the best sorting approach selection will be discussed and the corresponding conclusions on it will be made.

**The earliest-job sorting and three other sorting approaches**

Given $N$ jobs to be scheduled, $N \in \mathbb{N} \setminus \{1\}$, with their respective processing periods [14, 15]

$$\mathbf{H} = [H_n]_{1\times N} \in \mathbb{N}^N, \tag{1}$$

priority weights

$$\mathbf{W} = [w_n]_{1\times N} \in \mathbb{N}^N, \tag{2}$$

release dates

$$\mathbf{R} = [r_n]_{1\times N} \in \mathbb{N}^N, \tag{3}$$

and due dates

$$\mathbf{D} = [d_n]_{1\times N} \in \mathbb{N}^N, \tag{4}$$

the RPP-RAP heuristic originally based on the earliest-job sorting builds stepwise a schedule $\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times T}$, where $T = \sum_{n=1}^{N} H_n$, as time $t$ progresses [2, 3, 8, 16, 17]. The RPP of the job which is currently scheduled is updated, but all the starting RPPs (before the scheduling starts) are set at processing periods (1):

$$q_n = H_n \quad \forall n = \overline{1,\ N}. \tag{5}$$

For every set of available jobs

$$A(t) = \{i \in \{\overline{1,\ N}\} : r_i \leqslant t \text{ and } q_i > 0\} \subset \{\overline{1,\ N}\} \tag{6}$$

the respective RAPs are

$$b_i = \max\{0,\ d_i - t + 1\} \quad \forall i \in A(t). \tag{7}$$

Then a subset

$$A^*(t) = \arg\max_{i\in A(t)} \frac{w_i}{\max\{q_i,\ b_i\}} \tag{8}$$

is determined. If $|A^*(t)| = 1$, then

$$\tilde{s}_t = i^* \quad \text{by} \quad A^*(t) = \{i^*\} \subset A(t) \subset \{\overline{1,\ N}\} \tag{9}$$

and the RPP of job $i^*$ is updated as

$$q_{i^*}^{(\text{obs})} = q_{i^*} \quad \text{and} \quad q_{i^*} = q_{i^*}^{(\text{obs})} - 1; \tag{10}$$

otherwise

$$A^*(t) = \{i_l^*\}_{l=1}^{L} \subset A(t) \subset \{\overline{1,\ N}\} \quad \text{by} \quad L > 1, \tag{11}$$

whence

$$\tilde{s}_t = i_1^* \tag{12}$$

and the RPP of job $i_1^*$ is updated as

$$q_{i_1^*}^{(\text{obs})} = q_{i_1^*} \quad \text{and} \quad q_{i_1^*} = q_{i_1^*}^{(\text{obs})} - 1. \tag{13}$$

Assignment (12) executed by condition (11) implies that the earliest job is preferred to be scheduled [18] when there are two or more maximal decisive ratios in subset (8). Thus, job $n$ is completed after moment $\tilde{\theta}(n; H_n)$ if

$$\tilde{s}_{\tilde{\theta}(n;\, h_n)} = n \quad \forall\, h_n = \overline{1,\, H_n} \quad \text{by} \quad \tilde{\theta}(n; h_n) \in \{\overline{1,\, T}\}$$
$$\text{and} \quad \tilde{\theta}(n; h_n) < \tilde{\theta}(n; h_n + 1) \quad \text{for} \quad h_n = \overline{1,\, H_n - 1}$$

in schedule $\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times T}$ returned by the heuristic. Finally, amount

$$\tilde{\vartheta}(N) = \sum_{n=1}^{N} w_n \cdot \max\{0,\, \tilde{\theta}(n; H_n) - d_n\} \tag{14}$$

is an approximately minimal TWT corresponding to this schedule [3, 8, 18]. So, in the case when (11) is true, the RPP-RAP heuristic sorts maximal decisive ratios in subset (8) by release dates, where the scheduling preference is given to the earliest job (whose release date is the least). However, along with this earliest-job sorting, the other three approaches to sorting might be used: the RPP-or-due-date sorting, the min-RPP sorting, and random sorting [13].

The RPP-or-due-date sorting is executed as follows. If the RAP of job $i^*$ is $b_{i^*} > 0$, then

$$\lambda_{i^*} = \frac{w_{i^*}}{q_{i^*}} \quad \text{by} \quad i^* \in \{i_l^*\}_{l=1}^{L}, \tag{15}$$

else

$$\lambda_{i^*} = \frac{w_{i^*}}{d_{i^*}} \quad \text{by} \quad i^* \in \{i_l^*\}_{l=1}^{L}. \tag{16}$$

Subsequently, a subset

$$F^{**}(t) = \arg\min_{i^* \in A^*(t)} \lambda_{i^*} = \{i_l^{**}\}_{l=1}^{L^*} \tag{17}$$

is found and job $i_1^{**}$ is scheduled:

$$\tilde{s}_t = i_1^{**}, \tag{18}$$

whereupon the RPP of job $i_1^{**}$ is updated as

$$q_{i_1^{**}}^{(\text{obs})} = q_{i_1^{**}} \quad \text{and} \quad q_{i_1^{**}} = q_{i_1^{**}}^{(\text{obs})} - 1. \tag{19}$$

For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times 5} = [3 \quad 5 \quad 4 \quad 4 \quad 3], \tag{20}$$

priority weights

$$\mathbf{W} = [w_n]_{1\times 5} = [6 \quad 6 \quad 7 \quad 9 \quad 3], \tag{21}$$

by release dates $r_n = n \quad \forall\, n = \overline{1,\, 5}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 5} = [2 \quad 12 \quad 19 \quad 7 \quad 7] \tag{22}$$

is solved by the RPP-RAP heuristic using the RPP-or-due-date sorting which produces schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 19} = [1 \quad 1 \quad 1 \quad 4 \quad 4 \quad 4 \quad 4 \quad 2 \quad 2$$
$$2 \quad 2 \quad 2 \quad 5 \quad 5 \quad 5 \quad 3 \quad 3 \quad 3 \quad 3] \tag{23}$$

whose TWT is

$$\tilde{\vartheta}(5) = \sum_{n=1}^{5} w_n \cdot \max\{0,\, \tilde{\theta}(n; H_n) - d_n\} =$$
$$6 \cdot \max\{0,\, 3 - 2\} + 6 \cdot \max\{0,\, 12 - 12\} +$$
$$7 \cdot \max\{0,\, 19 - 19\} +$$
$$9 \cdot \max\{0,\, 7 - 7\} + 3 \cdot \max\{0,\, 15 - 7\} = 30. \tag{24}$$

The RPP-RAP heuristic using the earliest-job sorting produces schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 19} = [1 \quad 1 \quad 1 \quad 4 \quad 4 \quad 4 \quad 4 \quad 2 \quad 2$$
$$2 \quad 2 \quad 2 \quad 3 \quad 3 \quad 3 \quad 3 \quad 5 \quad 5 \quad 5] \tag{25}$$

whose TWT is greater by 12 units (i. e., the in-heuristic gap here is 40 %, implying the inaccuracy with respect to the approach producing a lesser value of TWT, although, generally speaking, that TWT is not guaranteed to be the real minimum of TWT):

$$\tilde{\vartheta}(5) = \sum_{n=1}^{5} w_n \cdot \max\{0,\, \tilde{\theta}(n; H_n) - d_n\} =$$
$$6 \cdot \max\{0,\, 3 - 2\} + 6 \cdot \max\{0,\, 12 - 12\} +$$
$$7 \cdot \max\{0,\, 16 - 19\} + 9 \cdot \max\{0,\, 7 - 7\} +$$
$$3 \cdot \max\{0,\, 19 - 7\} = 42. \tag{26}$$

This is the example of that the RPP-or-due-date sorting can outperform the earliest-job sorting at least by 40 %.

The min-RPP sorting consists in scheduling a job whose RPP is minimal. A subset

$$H^{**}(t) = \arg\min_{i^* \in A^*(t)} q_{i^*} = \{i_l^{**}\}_{l=1}^{L^*} \tag{27}$$

is found, and job $i_1^{**}$ is scheduled as (18) by (19) if $|H^{**}(t)| = L^* = 1$. Otherwise, if $|H^{**}(t)| = L^* > 1$, a subset

$$H^{***}(t) = \arg \min_{i^{**} \in H^{**}(t)} q_{i^{**}} = \{i_l^{***}\}_{l=1}^{L^{**}} \qquad (28)$$

is found, and job $i_1^{***}$ is scheduled:

$$\tilde{s}_t = i_1^{***}, \qquad (29)$$

whereupon the RPP of job $i_1^{***}$ is updated as

$$q_{i_1^{***}}^{(\text{obs})} = q_{i_1^{***}} \quad \text{and} \quad q_{i_1^{***}} = q_{i_1^{***}}^{(\text{obs})} - 1. \qquad (30)$$

For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times 5} = [5 \quad 5 \quad 3 \quad 4 \quad 3], \qquad (31)$$

priority weights

$$\mathbf{W} = [w_n]_{1\times 5} = [9 \quad 5 \quad 5 \quad 6 \quad 4], \qquad (32)$$

by release dates $r_n = n \quad \forall n = \overline{1, 5}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 5} = [10 \quad 5 \quad 7 \quad 2 \quad 7] \qquad (33)$$

is solved by the RPP-RAP heuristic using the min-RPP sorting which produces schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 20} = [1 \quad 1 \quad 1 \quad 4 \quad 4 \quad 4 \quad 4 \quad 1 \quad 1 \quad 3$$
$$3 \quad 3 \quad 5 \quad 5 \quad 5 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2] \qquad (34)$$

whose TWT is

$$\tilde{\vartheta}(5) = \sum_{n=1}^{5} w_n \cdot \max\{0, \tilde{\theta}(n; H_n) - d_n\} =$$
$$9 \cdot \max\{0, 9 - 10\} + 5 \cdot \max\{0, 20 - 5\} +$$
$$5 \cdot \max\{0, 12 - 7\} + 6 \cdot \max\{0, 7 - 2\} +$$
$$4 \cdot \max\{0, 15 - 7\} = 162. \qquad (35)$$

It is noteworthy that the RPP-RAP heuristic using the earliest-job sorting produces the same schedule (34), whereas the RPP-or-due-date sorting for job lengths (31), priority weights (32), and due dates (33) produces schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 20} = [1 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 1 \quad 1 \quad 1 \quad 1$$
$$3 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4 \quad 5 \quad 5 \quad 5] \qquad (36)$$

whose TWT is greater by 15 units (i. e., the in-heuristic gap here is 9.26 %):

$$\tilde{\vartheta}(5) = \sum_{n=1}^{5} w_n \cdot \max\{0, \tilde{\theta}(n; H_n) - d_n\} =$$
$$9 \cdot \max\{0, 10 - 10\} + 5 \cdot \max\{0, 6 - 5\} +$$
$$5 \cdot \max\{0, 13 - 7\} + 6 \cdot \max\{0, 17 - 2\} +$$
$$4 \cdot \max\{0, 20 - 7\} = 177. \qquad (37)$$

This is the example of that the min-RPP sorting can outperform the RPP-or-due-date sorting, and the latter, unlike the example with job lengths (20), priority weights (21), and due dates (22), can be outperformed by the earliest-job sorting.

Finally, the random sorting consists in just a random selection of a job from subset (8). Thus, in the case when (11) is true, number $m$ is randomly selected from subset $\{\overline{1, L}\}$ and job $i_m^*$ is scheduled:

$$\tilde{s}_t = i_m^*, \qquad (38)$$

whereupon the RPP of job $i_m^*$ is updated as

$$q_{i_m^*}^{(\text{obs})} = q_{i_m^*} \quad \text{and} \quad q_{i_m^*} = q_{i_m^*}^{(\text{obs})} - 1. \qquad (39)$$

However, if a schedule is built by using the random sorting by (38), (39), instead of (12) by (11), then, generally speaking, the heuristic produces random schedules and TWT. Nevertheless, the random sorting can outperform each of the three above-described sorting approaches. For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times 4} = [4 \quad 3 \quad 4 \quad 5], \qquad (40)$$

priority weights

$$\mathbf{W} = [w_n]_{1\times 4} = [7 \quad 3 \quad 10 \quad 2], \qquad (41)$$

by release dates $r_n = n \quad \forall n = \overline{1, 4}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 4} = [8 \quad 3 \quad 11 \quad 9] \qquad (42)$$

is solved by the random-sorting RPP-RAP heuristic producing two different schedules, one of which is

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 16} = [1 \quad 2 \quad 2 \quad 2 \quad 1 \quad 1 \quad 1 \quad 3$$
$$3 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4 \quad 4] \qquad (43)$$

whose TWT is

$$\tilde{\vartheta}(4) = \sum_{n=1}^{4} w_n \cdot \max\{0, \tilde{\theta}(n; H_n) - d_n\} =$$
$$7 \cdot \max\{0, 7 - 8\} + 3 \cdot \max\{0, 4 - 3\} +$$
$$10 \cdot \max\{0, 11 - 11\} + 2 \cdot \max\{0, 16 - 9\} = 17. \qquad (44)$$

The other schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 16} = [1 \quad 1 \quad 1 \quad 1 \quad 3 \quad 3 \quad 3 \quad 3 \quad 2 \quad 2 \quad 2 \quad 4 \quad 4 \quad 4 \quad 4 \quad 4], \qquad (45)$$

whose probability is equal to the probability of schedule (43), is much worse because its TWT is 21 units greater:

$$\tilde{\vartheta}(4) = \sum_{n=1}^{4} w_n \cdot \max\{0, \tilde{\theta}(n; H_n) - d_n\} = 7 \cdot \max\{0, 4-8\} + 3 \cdot \max\{0, 11-3\} + 10 \cdot \max\{0, 8-11\} + 2 \cdot \max\{0, 16-9\} = 38. \qquad (46)$$

At the same time, the RPP-RAP heuristic using the earliest-job sorting and the min-RPP sorting also produces schedule (45). However, the RPP-or-due-date sorting produces schedule (43). So, the random sorting here outperforms (by 21 units of TWT) the earliest-job sorting and the min-RPP sorting at 50 % rate. In another example, where

$$\mathbf{H} = [H_n]_{1\times 7} = [4 \quad 3 \quad 5 \quad 5 \quad 6 \quad 6 \quad 5], \qquad (47)$$

priority weights

$$\mathbf{W} = [w_n]_{1\times 7} = [2 \quad 8 \quad 5 \quad 6 \quad 10 \quad 2 \quad 1], \qquad (48)$$

by release dates $r_n = n \quad \forall n = \overline{1, 4}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 7} = [7 \quad 12 \quad 3 \quad 6 \quad 7 \quad 15 \quad 9], \qquad (49)$$

the random-sorting RPP-RAP heuristic produces a few schedules, one of which

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 34} = [1 \quad 2 \quad 3 \quad 3 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 2 \quad 2 \quad 3 \quad 3 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4 \quad 1 \quad 1 \quad 1 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7], \qquad (50)$$

has TWT

$$\tilde{\vartheta}(7) = \sum_{n=1}^{7} w_n \cdot \max\{0, \tilde{\theta}(n; H_n) - d_n\} = 2 \cdot \max\{0, 23-7\} + 8 \cdot \max\{0, 12-12\} + 5 \cdot \max\{0, 15-3\} + 6 \cdot \max\{0, 20-6\} + 10 \cdot \max\{0, 10-7\} + 2 \cdot \max\{0, 29-15\} + 1 \cdot \max\{0, 34-9\} = 259. \qquad (51)$$

The other schedules by the random sorting have $\tilde{\vartheta}(7) = 269$. The same TWT is produced by the earliest-job sorting, the RPP-or-due-date sorting, and the min-RPP sorting. Despite the random sorting produces more than two schedules, they have only two possible TWT (whose probabilities are equal): $\tilde{\vartheta}(7) = 259$ and $\tilde{\vartheta}(7) = 269$. Thus, the random sorting here outperforms (by 10 units of TWT) the other three sorting approaches at 50 % rate.

### Generation of TPIF1MP scheduling problems

As there is no an obvious "leader" among the above-described four sorting approaches, the possibility of such a "leadership" (implying the capability to produce heuristically minimal TWT more often than others) should be ascertained via a statistical analysis of the performance. In TPIF1MP scheduling, the release dates for (3) can be given in ascending order as follows:

$$r_n = n \quad \forall n = \overline{1, N}. \qquad (52)$$

The due dates for (4) are generated as [8, 18]

$$d_n = r_n + H_n - 1 + b_n \quad \forall n = \overline{1, N}. \qquad (53)$$

by the respective random due date shift [7, 8]

$$b_n = \psi(H_n \cdot \zeta) \quad \text{for} \quad n = \overline{1, N} \qquad (54)$$

with a pseudorandom number $\zeta$ drawn from the standard normal distribution (with zero mean and unit variance), and function $\psi(\xi)$ returning the integer part of number $\xi$ (e. g., see [7, 8]). The job lengths for (1) are generated as [3]

$$H_n = \psi(A\upsilon + 2) \ \text{ for } \ n = \overline{1, N} \ \text{ by } \ A = \overline{2, 20} \qquad (55)$$

with a pseudorandom number $\upsilon$ [19, 20] drawn from the standard uniform distribution on the open interval (0; 1). So, the job length is randomly generated between 2 and $A - 1$ [3]. The job priority weights for (2) are generated similarly to (55):

$$w_n = \psi(b\upsilon + 1) \quad \text{for} \quad n = \overline{1, N} \qquad (56)$$

by a weight amplitude factor

$$b \in \{\{10u\}_{u=1}^{10}, \{100z\}_{z=2}^{10}\}. \qquad (57)$$

Once a vector of job lengths (1) is generated, due date shifts (54) are generated until

$$d_n \geqslant 1 \quad \forall n = \overline{1, N}. \qquad (58)$$

and TWT is not 0 (i. e., the due dates are not very great, so at least one tardy job would exist).

**Computational study**

A first series of TPIF1MP scheduling problems is generated by (52)–(58), where 500 scheduling problems are generated for every $N = \overline{3, 25}$ (owing to that the schedules of 2 jobs by the RPP-RAP heuristic always have the exactly minimal TWT) and $A = \overline{2, 20}$ and (57). So, altogether 4151500 scheduling problems are generated in the first series. Each scheduling problem is solved by the RPP-RAP heuristic using the four sorting approaches: the earliest-job sorting (which initially constitutes the RPP-RAP heuristic itself) by (11)–(13), the RPP-or-due-date sorting by (15)–(19), the min-RPP sorting by (27) with (18) by (19) or (28)–(30), and the random sorting by (38), (39).

Table 1 presents the number and percentage of generated scheduling problems whose TWT has been revealed to be minimal for the given sorting approach, whereas the other three sorting approaches have produced greater values of TWT (i. e., the heuristic with the given sorting approach "has won"). The most "winning" sorting approach is the min-RPP sorting covered nearly a half of all sole "wins". Despite its rather probabilistic nature, the random sorting has had almost a quarter of all sole "wins".

***Table* 1.** The number of sole "wins" in the first series

| Sorting approach | Earliest-job sorting | RPP-or-due-date sorting | Min-RPP sorting | Random sorting |
|---|---|---|---|---|
| "Wins" | 4245 | 3216 | 10914 | 5761 |
| Percentage of "wins" | 17.6 | 13.3 | 45.2 | 23.9 |

In the case of TPIF1MP scheduling problems without priority weights, sole "wins" constitute about 2.39 % to 2.44 % of the volume of generated problems [13]. Here, in the first series, the percentage is just 0.5814 % (for 24136 sole "wins" out of 4151500 generated scheduling problems). Nevertheless, this is the average value. The sole "wins" percentage decreases as the weight amplitude factor increases (Fig. 1). Meanwhile, the interrelationship among the percentages of "wins" given in Table 1 remains almost the same (Fig. 2).

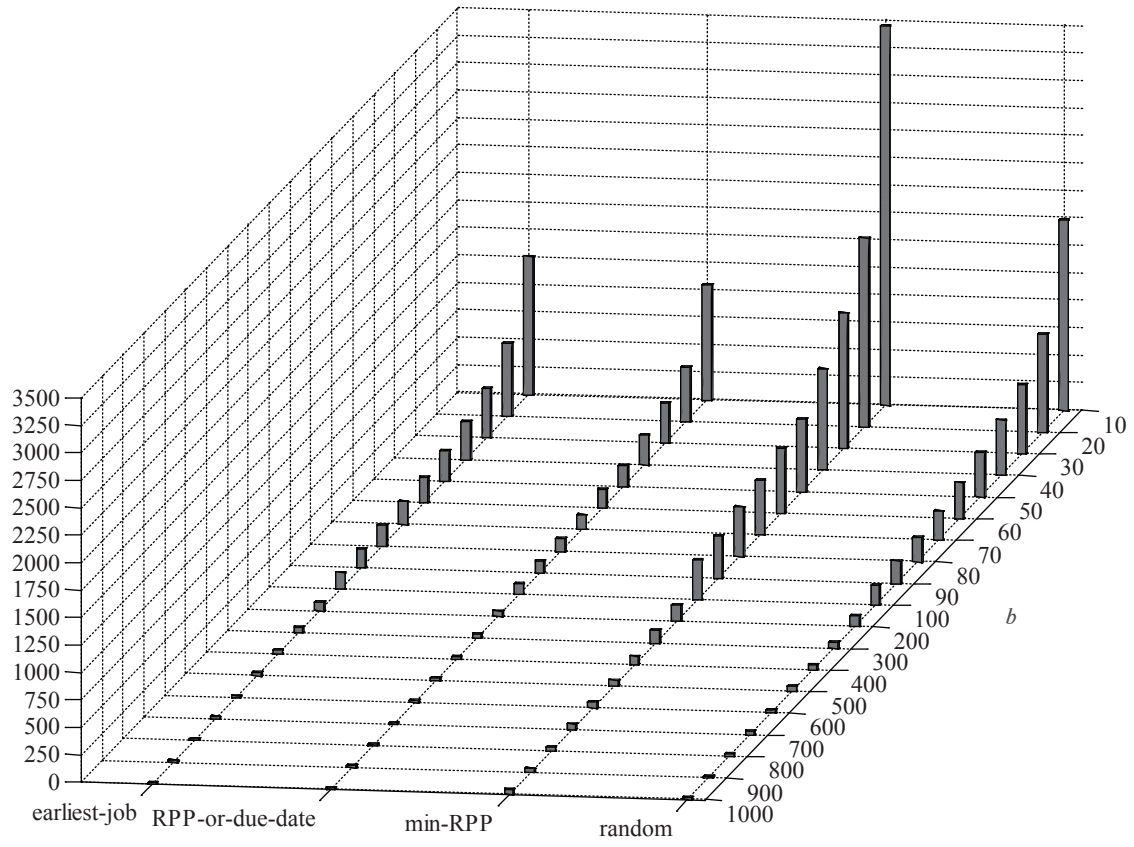


Fig. 1. The number of sole "wins" in the first series versus the weight amplitude factor and the sorting approach type

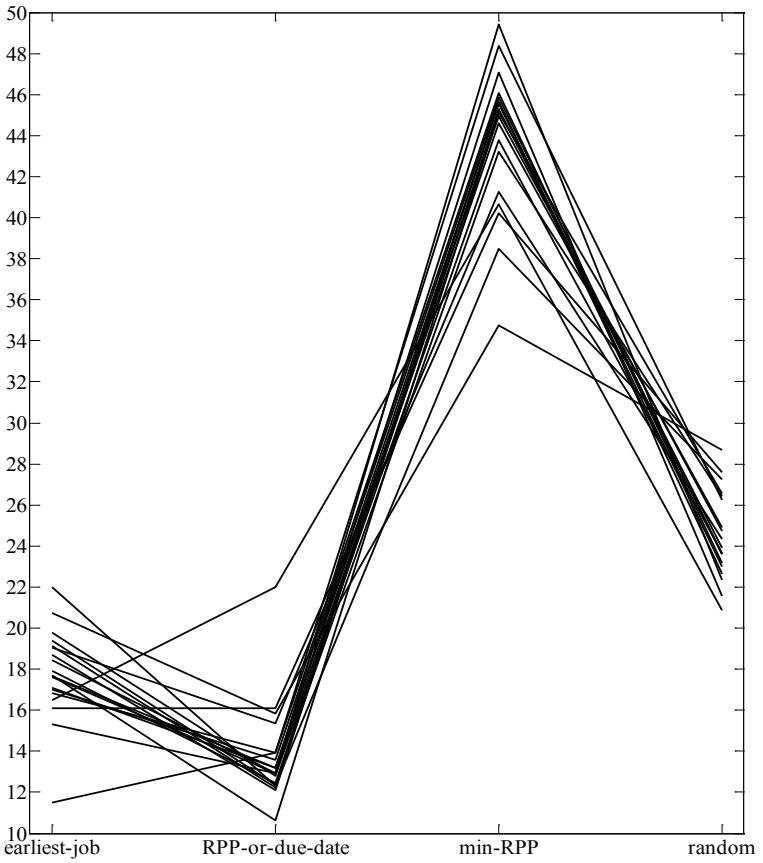


Fig. 2. The percentage of "wins" in the first series versus the sorting approach type by varying the weight amplitude factor

Obviously, group "wins", where two to four sorting approaches have produced the same (minimal) TWT, dominate. They constitute 99.4186 % (for 4127364 group "wins" out of 4151500 generated scheduling problems) of the volume of generated problems, where 11 "winning" groups have been revealed (in fact, this is the total set of groups; there cannot be other groups). The distribution of the number of group "wins" is presented in Table 2 where the group membership is highlighted with a gray color. The highest and dominating percentage (98.397 %) of group "wins" has been revealed to be for the group consisting of all four sorting approaches. As the weight amplitude factor increases, this whole-group "wins" percentage increases (Fig. 3, where dotted markers are used), whereas the 10 numbers of non-whole-group "wins" decrease (Fig. 4). The polyline of the interrelationship among the percentages of non-whole-group "wins" almost vanishes as the weight amplitude factor becomes sufficiently great (Fig. 5).

*Table* 2. The number of group "wins" in the first series

| Group of sorting Approaches which produce the same value of TWT | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Earliest-job sorting | ■ | ■ | ■ | | | | ■ | ■ | ■ | | ■ |
| | RPP-or-due-date sorting | ■ | | | ■ | ■ | | ■ | ■ | | ■ | ■ |
| | Min-RPP sorting | | ■ | | ■ | | ■ | ■ | | ■ | ■ | ■ |
| | Random sorting | | | ■ | | ■ | ■ | | ■ | ■ | ■ | ■ |
| Group number, # | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| Group "wins" | | 3926 | 8107 | 3734 | 7729 | 3040 | 9712 | 10966 | 4006 | 7679 | 7250 | 4061215 |
| Percentage of "wins" | | 0.095 | 0.196 | 0.091 | 0.187 | 0.074 | 0.235 | 0.266 | 0.097 | 0.186 | 0.176 | 98.397 |

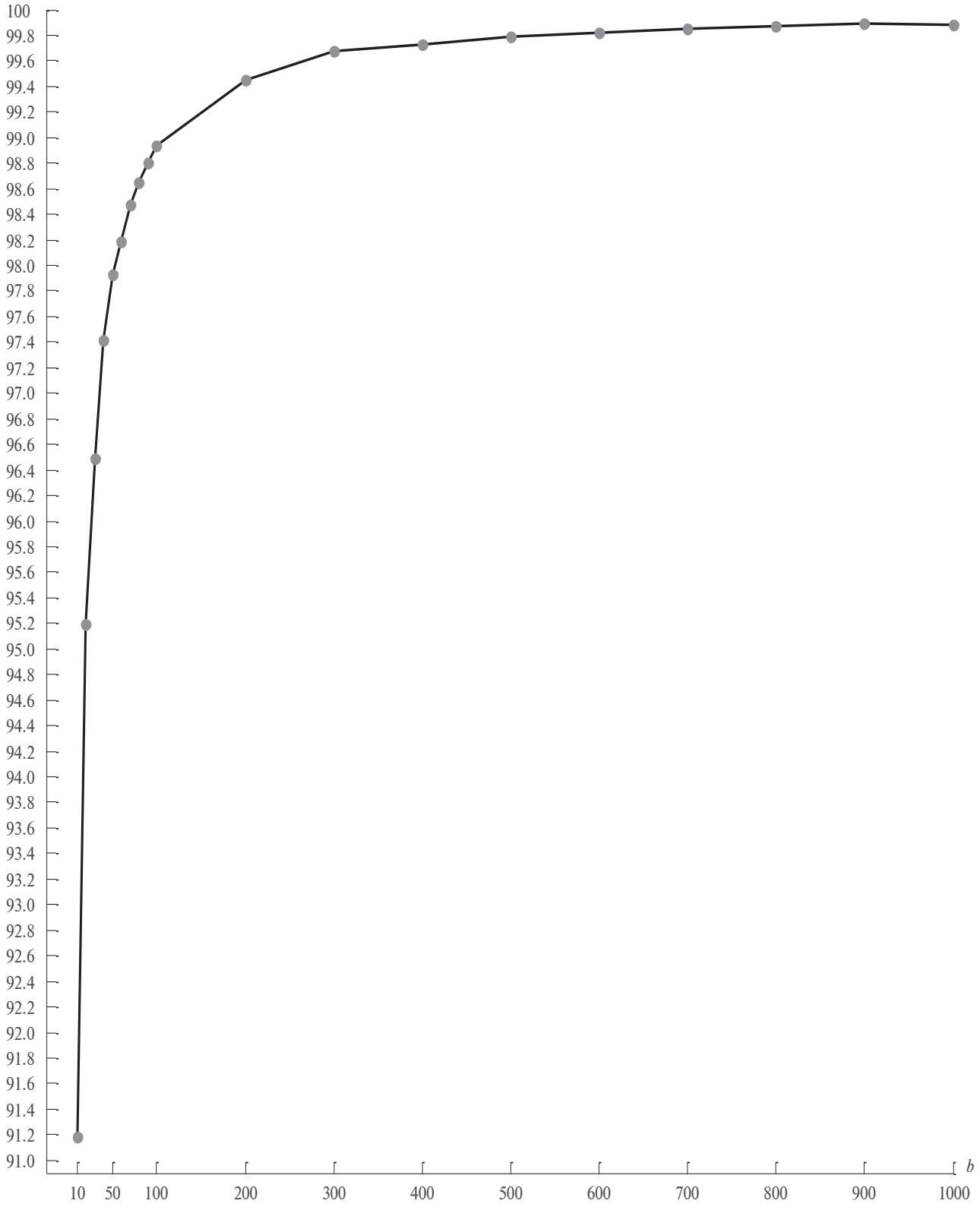


Fig. 3. The whole-group “wins” percentage in the first series versus the weight amplitude factor

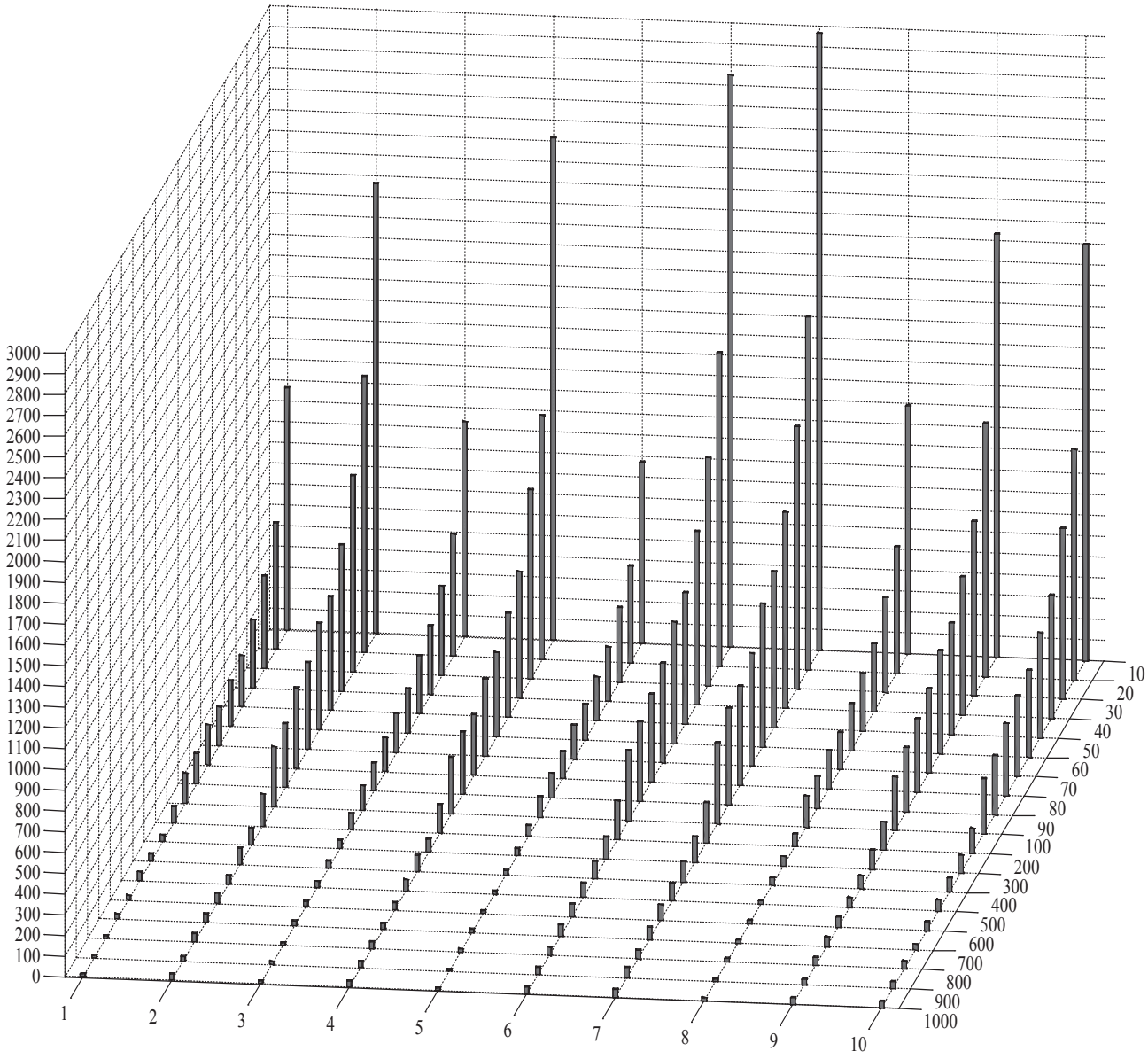


Fig. 4. The 10 numbers of non-whole-group "wins" in the first series versus the weight amplitude factor

Unlike the case of TPIF1MP scheduling problems without priority weights [13], here the min-RPP sorting is not the incontestable "winner" because it does not belong to every non-whole group (i. e., to groups ##1–10). However, it is worth to note that it has had 51443 non-whole-group "wins" in the first series, whereas the earliest-job sorting, the RPP-or-due-date sorting, and the random sorting have had 38418, 36917, and 35421 non-whole-group "wins", respectively. So, the min-RPP sorting does have nonetheless some advantage, which might be taken into account when a preference to a single sorting approach is to be given (because applying simultaneously two or more sorting approaches to solving a scheduling problem will slow down the process of obtaining a solution, whichever short computation time is).

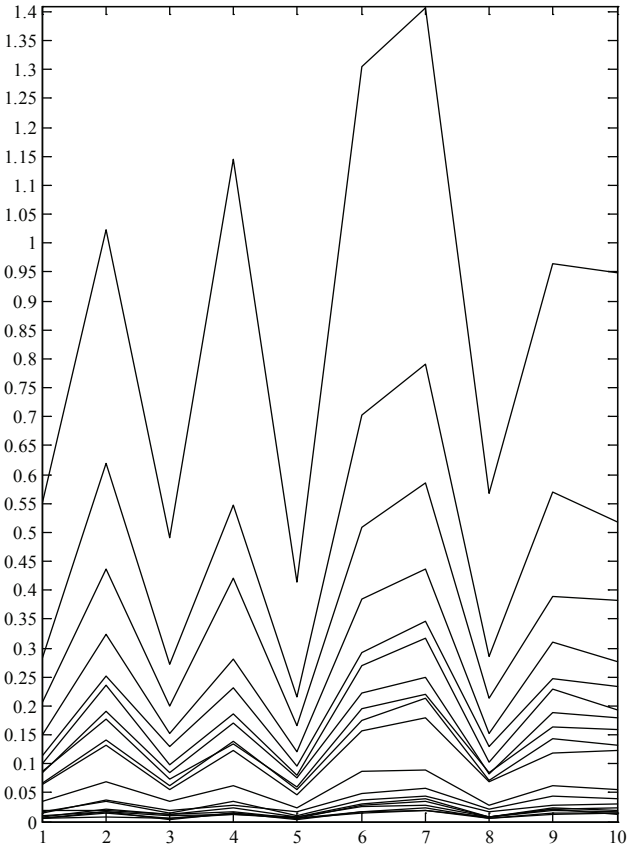


Fig. 5. The percentage of non-whole-group "wins" in the first series versus the group number type by varying the weight amplitude factor

To get convinced that the obtained results are statistically reliable and repeatable, a second series of TPIF1MP scheduling problems is analogously generated by (52)–(58). Table 3 presents the number of sole "wins" in the second series along with the relative difference from the first series. Sole "wins" constitute 0.5836 % of the volume of generated problems in the second series. The difference between the series does not exceed 7.5 %. This is why the barred plot of the number of sole "wins" versus the weight amplitude factor (Fig. 6) is very similar to that for the first series (Fig. 1). The difference between the barred plots in Figs. 1 and 6 is tiny, indeed. Besides, the interrelationship among the percentages of "wins" given in Table 3 remains almost the same (Fig. 7) by just nearly repeating the plot in Fig. 2. The difference between the polylines in Figs. 2 and 7 is not that tiny, but the shape of those two polyline bunches is the same.

***Table* 3.** The number of sole "wins" in the second series

| Sorting approach | Earliest-job sorting | RPP-or-due-date sorting | Min-RPP sorting | Random sorting |
|---|---|---|---|---|
| "Wins" | 3927 | 3273 | 11168 | 5862 |
| Percentage of "wins" | 16.2 | 13.5 | 46.1 | 24.2 |
| Difference (%) from the first series | 7.49 | 1.77 | 2.33 | 1.75 |

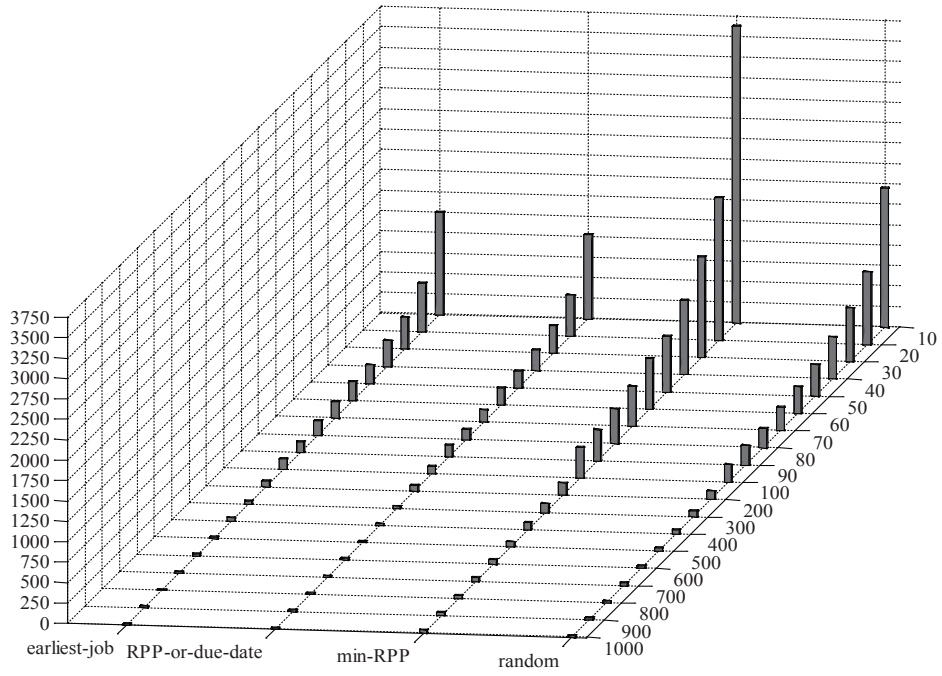


Fig. 6. The number of sole "wins" in the second series versus the weight amplitude factor and the sorting approach type

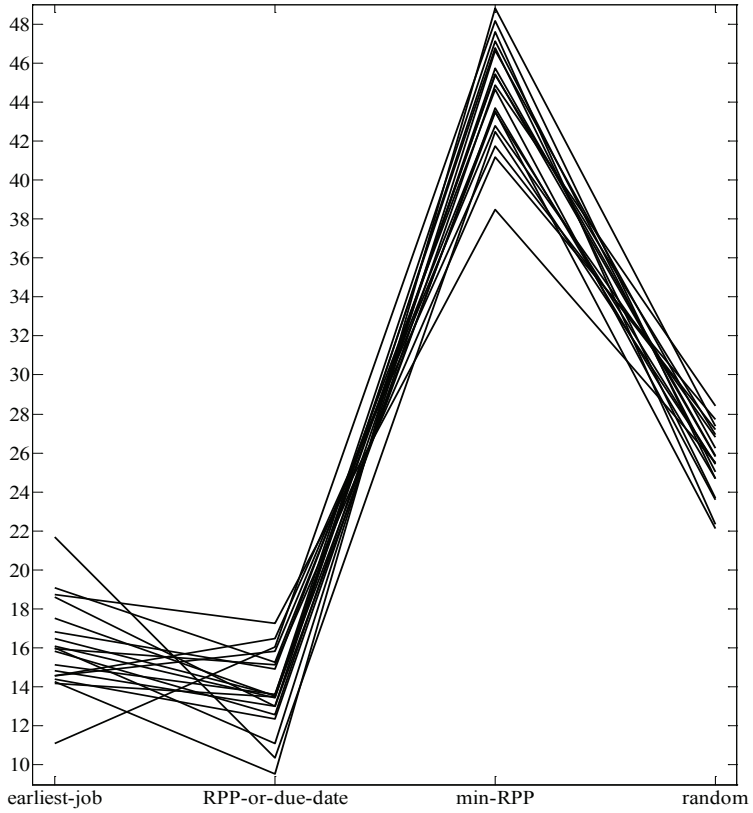


Fig. 7. The percentage of "wins" in the second series versus the sorting approach type by varying the weight amplitude factor

Group "wins" constitute 99.4164 % (for 4127270 group "wins" out of 4151500 generated scheduling problems) of the volume of generated problems in the second series. The distribution of the number of group "wins" is presented in Table 4 along with the relative difference from the first series. The difference between the series does not exceed 2.5 %, where the difference for whole-group "wins" is just 0.01 %. As the weight amplitude factor increases, this whole-group "wins" percentage increases (Fig. 8) similarly to that in the first series, as well as the 10 numbers of non-whole-group "wins" decrease (Fig. 9). The difference between the barred plots in Figs. 4 and 9 is tiny, indeed. The polyline of the interrelationship among the percentages of non-whole-group "wins" similarly vanishes as the weight amplitude factor becomes sufficiently great (Fig. 10). Even the peaks and cavities in Figs. 5 and 10 are very resembling that additionally confirms the repeatability of the first series.

***Table* 4.** The number of group "wins" in the second series

| Group of sorting approaches which produce the same value of TWT | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Earliest-job sorting | ■ | ■ | ■ | | | | ■ | ■ | ■ | | ■ |
| | RPP-or-due-date sorting | ■ | | | ■ | ■ | | ■ | ■ | | ■ | ■ |
| | Min-RPP sorting | | ■ | | ■ | | ■ | ■ | | ■ | ■ | ■ |
| | Random sorting | | | ■ | | ■ | ■ | | ■ | ■ | ■ | ■ |
| Group number, # | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| Group "wins" | | 3931 | 8119 | 3768 | 7919 | 2969 | 9835 | 10860 | 3932 | 7703 | 7292 | 4060942 |
| Percentage of "wins" | | 0.095 | 0.197 | 0.091 | 0.192 | 0.072 | 0.238 | 0.263 | 0.95 | 0.187 | 0.177 | 98.393 |
| Difference (%) from the first series | | 0.13 | 0.15 | 0.91 | 2.46 | 2.34 | 1.27 | 0.97 | 1.85 | 0.31 | 0.58 | 0.01 |

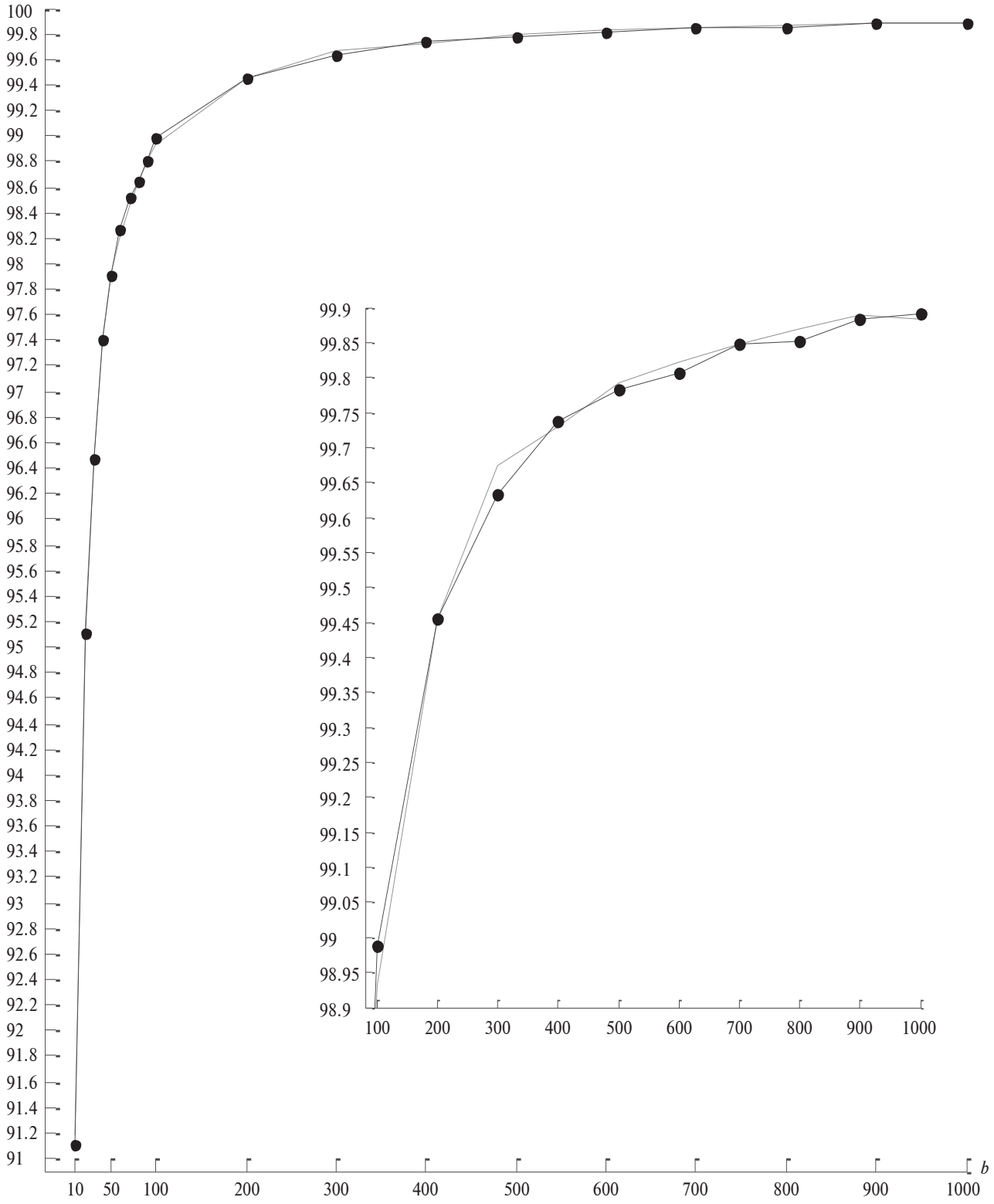


Fig. 8. The whole-group “wins” percentage in the second series versus the weight amplitude factor (the first series is shown also, without dotted markers)

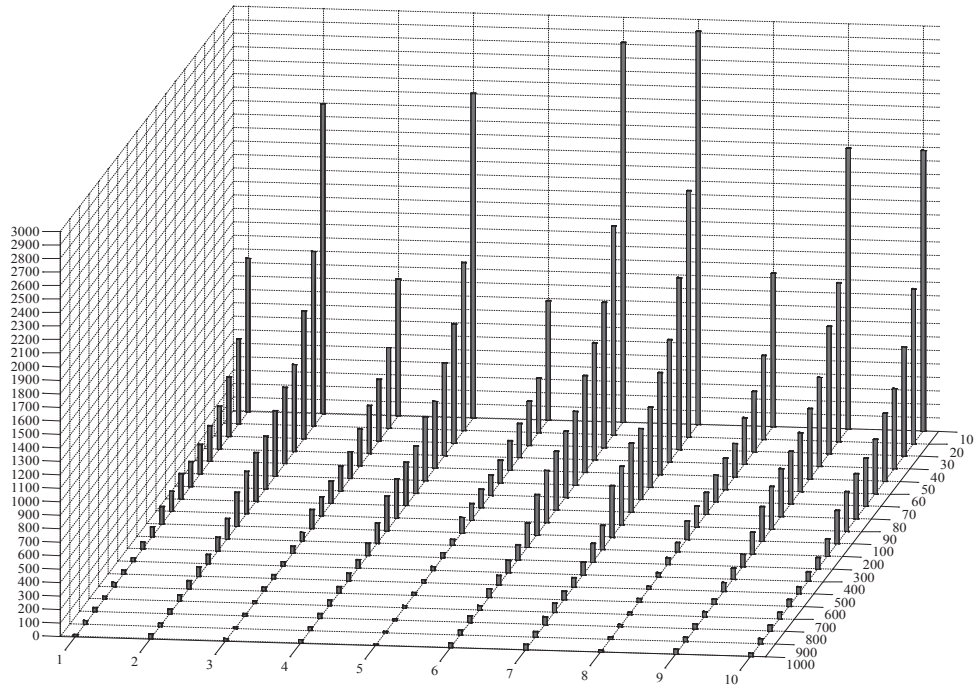


Fig. 9. The 10 numbers of non-whole-group "wins" in the second series versus the weight amplitude factor

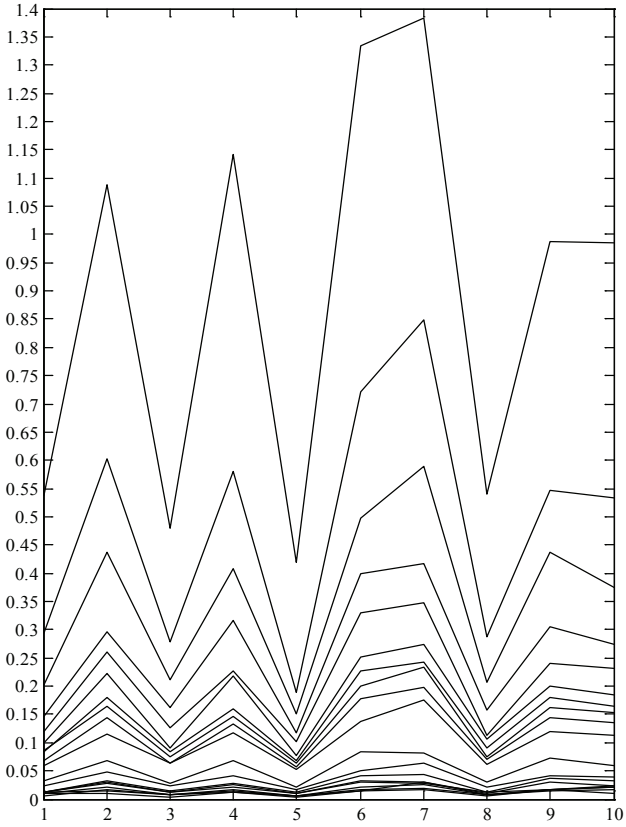


Fig. 10. The percentage of non-whole-group "wins" in the second series versus the group number type by varying the weight amplitude factor

Finally, it is worth to note that, in the second series, the earliest-job sorting, the RPP-or-due-date sorting, the min-RPP sorting, and the random sorting have had 38313, 36903, 51728, and 35499 non-whole-group "wins", respectively. These numbers differ from those in the first series by no more than 0.56 %, so it is valid (along with the above-mentioned similarities between the barred plots and polyline bunches) to affirm the series repeatable and statistically reliable. The repeatability, however, does not imply itself that it will be so easy to select the best sorting approach in order to make the RPP-RAP heuristic more accurate (or, in other words, to improve the RPP-RAP heuristic).

**Discussion**

The min-RPP sorting appears to have an advantage but it is hard to select this approach as the best. Indeed, there are examples of the scheduling problem in which the min-RPP sorting fails to be as accurate as other sorting approaches are. Moreover, the inaccuracy (with respect to the approach producing a lesser value of TWT) can be just gigantic. For instance, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times 3} = [9 \quad 16 \quad 3], \tag{59}$$

priority weights

$$\mathbf{W} = [w_n]_{1\times 3} = [24 \quad 63 \quad 13], \tag{60}$$

by release dates $r_n = n \quad \forall\, n = \overline{1,\, 3}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 4} = [23 \quad 36 \quad 4] \tag{61}$$

is solved by the min-RPP heuristic producing schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 28} = [1 \quad 2 \quad 3 \quad 3 \quad 3 \quad 2 \quad 2 \quad 2 \quad 2$$
$$2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2$$
$$2 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1] \tag{62}$$

whose TWT is

$$\tilde{\vartheta}(3) = \sum_{n=1}^{3} w_n \cdot \max\{0,\, \tilde{\theta}(n;\, H_n) - d_n\} =$$
$$24 \cdot \max\{0,\, 28 - 23\} + 63 \cdot \max\{0,\, 20 - 36\} +$$
$$13 \cdot \max\{0,\, 5 - 4\} = 133. \tag{63}$$

However, both the earliest-job sorting and the RPP-or-due-date sorting produce schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 28} = [1 \quad 2 \quad 3 \quad 3 \quad 3 \quad 2 \quad 2 \quad 2 \quad 2$$
$$2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 1 \quad 1 \quad 1 \quad 1$$
$$1 \quad 1 \quad 1 \quad 1 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2] \tag{64}$$

whose TWT is 120 units lesser:

$$\tilde{\vartheta}(3) = \sum_{n=1}^{3} w_n \cdot \max\{0,\, \tilde{\theta}(n;\, H_n) - d_n\} =$$
$$24 \cdot \max\{0,\, 23 - 23\} + 63 \cdot \max\{0,\, 28 - 36\} +$$
$$13 \cdot \max\{0,\, 5 - 4\} = 13. \tag{65}$$

The in-heuristic gap here is more than 923 % (!), which defies the min-RPP sorting best-selection outright. Such shocking counterexamples exist for the earliest-job sorting and the random sorting also. The RPP-or-due-date sorting, however, produce less shocking counterexamples. Table 5 contains a numerical description of the shocking counterexamples along with ordinary counterexamples in the first series, in which the inaccuracy of the sorting approach (with respect to the approach producing a lesser value of TWT) is that gigantic. The minimal and maximal number of jobs, at which the instances with TWT exceeding the in-heuristic gap have been registered, are given also. The similar results in the second series are given in Table 6.

***Table* 5.** The number of ordinary and shocking counterexamples per sorting approach in the first series

| In-heuristic gap, % | Earliest-job sorting | | | RPP-or-due-date sorting | | | Min-RPP sorting | | | Random sorting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Number of instances | Number of jobs | | Number of instances | Number of jobs | | Number of instances | Number of jobs | | Number of instances | Number of jobs | |
| | | min | max | | min | max | | min | max | | min | max |
| 10 | 840 | 3 | 18 | 637 | 3 | 23 | 645 | 3 | 16 | 1047 | 3 | 23 |
| 20 | 373 | 3 | 9 | 120 | 3 | 11 | 329 | 3 | 8 | 442 | 3 | 9 |
| 30 | 236 | 3 | 8 | 57 | 3 | 9 | 217 | 3 | 8 | 280 | 3 | 9 |
| 40 | 168 | 3 | 7 | 33 | 3 | 6 | 148 | 3 | 6 | 205 | 3 | 7 |
| 50 | 134 | 3 | 6 | 21 | 3 | 5 | 119 | 3 | 6 | 159 | 3 | 6 |
| 60 | 100 | 3 | 6 | 14 | 3 | 4 | 89 | 3 | 6 | 121 | 3 | 6 |
| 70 | 78 | 3 | 5 | 13 | 3 | 4 | 71 | 3 | 6 | 101 | 3 | 6 |
| 80 | 68 | 3 | 5 | 9 | 3 | 4 | 60 | 3 | 5 | 87 | 3 | 6 |
| 90 | 57 | 3 | 5 | 6 | 3 | 3 | 51 | 3 | 5 | 77 | 3 | 5 |
| 100 | 45 | 3 | 5 | 6 | 3 | 3 | 39 | 3 | 5 | 59 | 3 | 5 |
| 200 | 16 | 3 | 5 | 1 | 3 | 3 | 16 | 3 | 4 | 24 | 3 | 5 |
| 300 | 6 | 3 | 4 | 1 | 3 | 3 | 7 | 3 | 4 | 11 | 3 | 3 |
| 400 | 3 | 3 | 3 | 0 | – | – | 3 | 3 | 3 | 8 | 3 | 3 |
| 500 | 3 | 3 | 3 | 0 | – | – | 3 | 3 | 3 | 6 | 3 | 3 |
| 600 | 1 | 3 | 3 | 0 | – | – | 1 | 3 | 3 | 2 | 3 | 3 |
| 700 | 1 | 3 | 3 | 0 | – | – | 1 | 3 | 3 | 2 | 3 | 3 |
| 800 | 0 | – | – | 0 | – | – | 1 | 3 | 3 | 1 | 3 | 3 |

***Table* 6.** The number of ordinary and shocking counterexamples per sorting approach in the second series

| In-heuristic gap, % | Earliest-job sorting | | | RPP-or-due-date sorting | | | Min-RPP sorting | | | Random sorting | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Number of instances | Number of jobs | | Number of instances | Number of jobs | | Number of instances | Number of jobs | | Number of instances | Number of jobs | |
| | | min | max | | min | max | | min | max | | min | max |
| 10 | 834 | 3 | 21 | 587 | 3 | 25 | 641 | 3 | 20 | 1011 | 3 | 25 |
| 20 | 408 | 3 | 9 | 139 | 3 | 11 | 325 | 3 | 10 | 447 | 3 | 9 |
| 30 | 271 | 3 | 7 | 72 | 3 | 8 | 213 | 3 | 7 | 289 | 3 | 8 |
| 40 | 207 | 3 | 7 | 44 | 3 | 7 | 160 | 3 | 7 | 201 | 3 | 7 |
| 50 | 161 | 3 | 7 | 31 | 3 | 7 | 128 | 3 | 6 | 152 | 3 | 7 |
| 60 | 128 | 3 | 6 | 22 | 3 | 5 | 103 | 3 | 6 | 133 | 3 | 6 |
| 70 | 115 | 3 | 6 | 19 | 3 | 5 | 90 | 3 | 6 | 123 | 3 | 6 |
| 80 | 93 | 3 | 5 | 11 | 3 | 5 | 71 | 3 | 5 | 94 | 3 | 6 |
| 90 | 76 | 3 | 5 | 8 | 3 | 5 | 62 | 3 | 5 | 81 | 3 | 6 |
| 100 | 60 | 3 | 5 | 7 | 3 | 5 | 51 | 3 | 5 | 64 | 3 | 6 |
| 200 | 15 | 3 | 4 | 2 | 3 | 3 | 16 | 3 | 4 | 21 | 3 | 5 |
| 300 | 5 | 3 | 4 | 1 | 3 | 3 | 9 | 3 | 4 | 10 | 3 | 5 |
| 400 | 2 | 3 | 3 | 1 | 3 | 3 | 5 | 3 | 4 | 7 | 3 | 5 |
| 500 | 0 | – | – | 0 | – | – | 4 | 3 | 4 | 3 | 3 | 4 |
| 600 | 0 | – | – | 0 | – | – | 3 | 3 | 4 | 2 | 3 | 4 |
| 700 | 0 | – | – | 0 | – | – | 3 | 3 | 4 | 2 | 3 | 4 |
| 800 | 0 | – | – | 0 | – | – | 3 | 3 | 4 | 1 | 4 | 4 |

It is worth noting that the shocking counterexample with (59)–(65) falls beyond the statistics of Tables 5 and 6. Overall, Tables 5 and 6 allow seeing that the shocking counterexamples occur when 3 to 9 jobs are scheduled (when the in-heuristic gap is more than 30 %). To exclude the occurrence of gigantic inaccuracies, the respective integer linear programming model to exactly minimize TWT should be used instead of the heuristic. This model will be applicable owing to that scheduling up to 10 jobs is practically tractable [2, 3, 15, 21, 22].

It is also seen that the RPP-or-due-date sorting can produce an in-heuristic gap in more than 20 % by scheduling up to 11 jobs. This upper number is 9 for the earliest-job sorting and the random sorting, and is 10 for the min-RPP sorting. A pretty huge in-heuristic gap in more than 10 % can be produced by the min-RPP sorting when up to 20 jobs are scheduled (see Table 6, in which all the maximal numbers of jobs in the first row are greater than those in Table 5). This upper number is 21 for the earliest-job sorting, but it is 25 for the RPP-or-due-date sorting and the random sorting. Scheduling such numbers of jobs is unlikely to be tractable by the respective integer linear programming model.

Therefore, each of the sorting approaches has heavy disadvantages, although they occur rarely. As sole "wins" and non-whole-group "wins" are rare also, it is impossible to select the best sorting approach. Instead, a hyper-heuristic comprising the sorting approaches (i. e., the whole group, where each sorting is applied) may be constructed (although the process of obtaining a solution will be slowed down, unless a parallelization is used) [23, 24].

## Conclusions

Pairwise comparison of Tables 1 and 2 to Tables 3 and 4 allows affirming that it is sufficient to generate 500 scheduling problems for obtaining statistically reliable results. This is confirmed by comparing Figs. 1–5 to Figs. 6–10, which are a deeper dissection of the tables. Figs. 1, 4, 6, 9 also confirm that the four sorting approaches become more indistinguishable as the weight amplitude factor increases.

In minimizing TWT by the RPP-RAP heuristic, the earliest-job sorting, the RPP-or-due-date sorting, the min-RPP sorting, and the random sorting approaches produce schedules with the same TWT in over 98.39 % of TPIF1MP scheduling problems.

Although the influence of these approaches is different, it is therefore not really significant. The RPP-RAP heuristic is truly improved only by applying all the four sorting approaches to solving a scheduling problem. If a parallelization can be used to process two or even four sorting routines simultaneously, the computation time will not be significantly affected.

Based on Tables 5 and 6 with the shocking counterexamples, it is recommended to use the RPP-or-due-date sorting for scheduling up to a few tens of jobs. To reduce the likelihood of gigantic inaccuracy, the volume of a scheduling problem should be as great as possible. An open question is how to recognize a shocking counterexample in a scheduling problem with (1)–(4) beforehand.

**References**


[1] M.L. Pinedo, *Planning and Scheduling in Manufacturing and Services.* New York: Springer-Verlag, 2009, 536 p. doi: 10.1007/978-1-4419-0910-7

[2] V.V. Romanuke, "Accuracy of a heuristic for total weighted completion time minimization in preemptive single machine scheduling problem by no idle time intervals," *KPI Sci. News*, no. 3, pp. 52–62, 2019. doi: 10.20535/kpi-sn.2019.3.164804

[3] V.V. Romanuke, "Minimal total weighted tardiness in tight-tardy single machine preemptive idling-free scheduling," *Appl. Comput. Syst.*, vol. 24, no. 2, pp. 150–160, 2019. doi: 10.2478/acss-2019-0019

[4] M. Batsyn *et al.*, "Online heuristic for the preemptive single machine scheduling problem of minimizing the total weighted completion time," *Optimiz. Method. Softw.*, vol. 29, no. 5, pp. 955–963, 2014. doi: 10.1080/10556788.2013.854360

[5] V.V. Romanuke, "Infinity substitute in exactly minimizing total tardiness in tight-tardy progressive 1-machine scheduling by idling-free preemptions of equal-length jobs," *Bulletin of V. Karazin Kharkiv National Univ. Math. Modell. Inf. Technol. Autom. Control Syst.*, no. 44, pp. 94–101, 2019. doi: 10.26565/2304-6201-2019-44-10

[6] V.V. Romanuke, "Efficient exact minimization of total tardiness in tight-tardy progressive single machine scheduling with idling-free preemptions of equal-length jobs," *KPI Sci. News*, no. 1, pp. 27–39, 2020. doi: 10.20535/kpi-sn.2020.1.180877

[7] V.V. Romanuke, "Job order input for efficient exact minimization of total tardiness in tight-tardy progressive single machine scheduling with idling-free preemptions," *Proc. O.S. Popov ONAT*, vol. 1, no. 1, pp. 19–36, 2020. doi: 10.33243/2518-7139-2020-1-1-19-36

[8] V.V. Romanuke, "Tight-tardy progressive idling-free 1-machine preemptive scheduling by heuristic's efficient job order input," *KPI Sci. News*, no. 3, pp. 32–42, 2020. doi: 10.20535/kpi-sn.2020.3.199850

[9] K.-C. Ying *et al.*, "Sequencing single-machine tardiness problems with sequence dependent setup times using an iterated greedy heuristic," *Exp. Syst. Applicat.*, vol. 36, no. 3, pp. 7087–7092, 2009. doi: 10.1016/j.eswa.2008.08.033

[10] Y.H. Lee *et al.*, "A heuristic to minimize the total weighted tardiness with sequence-dependent setups", *IIE Trans.*, vol. 29, no. 1, pp. 45–52, 1997. doi: 10.1080/07408179708966311

[11] F. Jaramillo and M. Erkoc, "Minimizing total weighted tardiness and overtime costs for single machine preemptive scheduling," *Comput. & Ind. Eng.*, vol. 107, pp. 109–119, 2017. doi: 10.1016/j.cie.2017.03.012

[12] M. Tamannaei and M. Rasti-Barzoki, "Mathematical programming and solution approaches for minimizing tardiness and transportation costs in the supply chain scheduling problem," *Comput. & Ind. Eng.*, vol. 127, pp. 643–656, 2019. doi: 10.1016/j.cie.2018.11.003

[13] V.V. Romanuke, "A sorting improvement in the heuristic based on remaining available and processing periods to minimize total tardiness in progressive idling-free 1-machine preemptive scheduling," *KPI Sci. News*, no. 1, pp. 32–41, 2021. doi: 10.20535/kpisn.2021.1.211935

[14] M.L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, Springer International Publishing, 2016, 670 p. doi: 10.1007/978-3-319-26580-3

[15] R.L. Graham *et al.*, "Optimization and approximation in deterministic sequencing and scheduling: A survey," in *Annals of Discrete Mathematics*, vol. 5, 1979, pp. 287–326. doi: 10.1016/S0167-5060(08)70356-X

[16] R. Panneerselvam, "Simple heuristic to minimize total tardiness in a single machine scheduling problem," *Int. J. Advanc. Manuf. Technol.*, vol. 30, no. 7-8, pp. 722–726, 2006. doi: 10.1007/s00170-005-0102-1

[17] Z. Tian *et al.*, "An $O(n^2)$ algorithm for scheduling equal-length preemptive jobs on a single machine to minimize total tardiness," *J. Schedul.*, vol. 9, no. 4, pp. 343–364, 2006. doi: 10.1007/s10951-006-7039-6

[18] V.V. Romanuke, "Heuristic's job order efficiency in tight-tardy progressive idling-free 1-machine preemptive scheduling of equal-length jobs," *KPI Sci. News*, no. 2, pp. 64–73, 2020. doi: 10.20535/kpi-sn.2020.2.181869

[19] R.T. Kneusel, *Random Numbers and Computers*, Springer International Publishing, 2018, 260 p. doi: 10.1007/978-3-319-77697-2

[20] J.M.A. Trejo and C.S. Calude, "A new quantum random number generator certified by value indefiniteness," *Theoretical Comput. Sci.*, vol. 862, pp. 3–13, 2021. doi: 10.1016/j.tcs.2020.08.014

[21] W.-Y. Ku and J.C. Beck, “Mixed integer programming models for job shop scheduling: A computational analysis,” *Comput. & Operations Res.*, vol. 73, pp. 165–173, 2016. doi: 10.1016/j.cor.2016.04.006

[22] D. Kress *et al.*, “An exact solution approach for scheduling cooperative gantry cranes,” *European J. Operational Res.*, vol. 273, no. 1, pp. 82–101, 2019. doi: 10.1016/j.ejor.2018.07.043

[23] H. Chen *et al.*, “A hyper-heuristic based ensemble genetic programming approach for stochastic resource constrained project scheduling problem,” *Exp. Syst. Applicat.*, vol. 167, p. 114174, 2021. doi: 10.1016/j.eswa.2020.114174

[24] H.-B. Song and J. Lin, “A genetic programming hyper-heuristic for the distributed assembly permutation flow-shop scheduling problem with sequence dependent setup times,” *Swarm Evol. Comput.*, vol. 60, p. 100807, 2021. doi: 10.1016/j.swevo.2020.100807

В.В. Романюк

ПІДХОДИ ДО СОРТУВАННЯ В ЕВРИСТИЦІ НА ОСНОВІ ЗАЛИШКОВИХ НАЯВНИХ РЕСУРСІВ І ПЕРІОДІВ ДО ОБРОБКИ ДЛЯ МІНІМІЗАЦІЇ ЗАГАЛЬНОГО ЗВАЖЕНОГО ЗАПІЗНЮВАННЯ У ПОСТУПАЛЬНОМУ ОДНОМАШИННОМУ ПЛАНУВАННІ З ПЕРЕМИКАННЯМИ БЕЗ ПРОСТОЮ

**Проблематика.** У плануванні завдань із перемиканнями мінімізація загального зваженого запізнювання зазвичай зводиться до розв’язання комбінаторної задачі, котра стає практично нерозв’язною, щойно кількість завдань і їх періодів до обробки зростає. Щоб упоратися з цим викликом, використовують евристики. Близькою до найкращої є евристика, у якій вирішальним співвідношенням є зважене обернене значення максимуму пари залишкового періоду до обробки та залишкового наявного ресурсу. Однак ця евристика може виробляти розклади декількох робіт, загальне зважене запізнювання яких є вкрай великим, як порівнювати зі справжнім мінімумом. Тому ця евристика потребує удосконалень, одне з яких уже існує для завдань без ваг пріоритетів, де використовують підхід до сортування з мінімізацією залишкових періодів до обробки. Три інших підходи до сортування все ж можуть показувати навіть кращі результати, однак такі винятки є надзвичайно рідкісними.

**Мета дослідження.** Визначити вплив чотирьох підходів до сортування та спробувати вибрати найкращий для випадку завдань із вагами пріоритетів. Евристика буде застосована до щільного поступального одномашинного планування з перемиканнями без простою, в якому моменти запуску завдань подають у порядку зростання від 1 до кількості завдань, а моменти приймання виконання завдань встановлюють щільно за моментами запуску.

**Методика реалізації.** Проведено обчислювальне дослідження із застосуванням кожного з чотирьох евристичних підходів для мінімізації загального зваженого запізнювання. Для цього згенеровано дві послідовності по 4151500 задач планування. У розв’язку задачі планування підхід до сортування може “виграти” одноосібно чи “виграти” в групі з іншими підходами, продукуючи евристично мінімальне загальне зважене запізнювання. Розподіли одноосібних і групових “перемог” встановлюють для кожної послідовності.

**Результати дослідження.** Одноосібні “перемоги” та “перемоги” неповних груп є рідкісними: 4 підходи до сортування продукують розклади з тим самим загальним зваженим запізнюванням у понад 98,39 % задач планування. Відповідно, хоча вплив цих підходів різний, він не є по-справжньому значущим. Кожен із підходів до сортування має серйозні недоліки, котрі іноді призводять до гігантських неточностей, хоча вони й трапляються нечасто. Коли неточність виходить більше 30 %, це означає, що складаються розклади від 3 до 9 завдань.

**Висновки.** На відміну від випадку, коли завдання не мають ваг пріоритетів, для випадку з вагами пріоритетів неможливо вибрати найкращий підхід до сортування. Натомість можна побудувати гіперевристику, котра буде містити всі ці підходи до сортування (тобто повну групу, де застосовуватимуть кожен підхід). Використання паралелізації для одночасного оброблення двох або навіть чотирьох алгоритмів сортування особливо не вплине на час обчислень.

**Ключові слова:** одномашинне планування завдань з перемиканнями; загальне зважене запізнювання; евристика; підхід до сортування; залишкові періоди до обробки; залишкові наявні ресурси.

В.В. Романюк

ПОДХОДЫ К СОРТИРОВКЕ В ЭВРИСТИКЕ НА ОСНОВЕ ОСТАТОЧНЫХ ИМЕЮЩИХСЯ РЕСУРСОВ И ПЕРИОДОВ К ОБРАБОТКЕ ДЛЯ МИНИМИЗАЦИИ ОБЩЕГО ВЗВЕШЕННОГО ЗАПАЗДЫВАНИЯ В ПРОГРЕССИРУЮЩЕМ ОДНОМАШИННОМ ПЛАНИРОВАНИИ С ПЕРЕКЛЮЧЕНИЯМИ БЕЗ ПРОСТОЯ

**Проблематика.** В планировании заданий с переключениями минимизация общего взвешенного запаздывания обычно сводится к решению комбинаторной задачи, которая становится практически неразрешимой, как только количество заданий и их периодов к обработке возрастает. Чтобы справиться с этим вызовом, используют эвристики. Близкой к наилучшей является эвристика, в которой решающим соотношением является взвешенная обратная величина максимума пары остаточного периода к обработке и остаточного имеющегося ресурса. Однако эта эвристика может производить расписания нескольких заданий, общее взвешенное запаздывание которых является исключительно огромным в сравнении с настоящим минимумом. Поэтому эта эвристика требует улучшений, одно из которых уже существует для заданий без весов приоритетов, где используют подход к сортировке с минимизацией остаточных периодов к обработке. Три других подхода к сортировке всё же могут показывать результаты даже лучше, однако подобные исключения чрезвычайно редки.

**Цель исследования.** Установить влияние четырёх подходов к сортировке и попытаться выбрать наилучший для случая заданий с весами приоритетов. Эвристика будет применена к плотному прогрессирующему одномашинному планированию с переключениями без простоя, в котором моменты запуска заданий подают в порядке возрастания от 1 к количеству заданий, а моменты приёма выполнения заданий устанавливают плотно по моментам запуска.

**Методика реализации.** Проведено вычислительное исследование с применением каждого из четырёх эвристических подходов для минимизации общего взвешенного запаздывания. Для этого сгенерированы две последовательности по 4151500 задач

планирования. В решении задачи планирования подход к сортировке может “выиграть” единолично либо “выиграть” в группе с другими подходами, производя эвристически минимальное общее взвешенное запаздывание. Распределения единоличных и групповых “побед” устанавливают для каждой последовательности.

**Результаты исследования.** Единоличные “победы” и “победы” неполных групп редки: 4 подхода к сортировке производят расписания с тем же общим взвешенным запаздыванием в более чем 98.39 % задачах планирования. Соответственно, хотя влияние этих подходов разное, оно не является по-настоящему значимым. Каждый из подходов к сортировке имеет серьёзные недостатки, которые иногда приводят к гигантским неточностям, хотя они и случаются редко. Когда неточность получается более 30 %, это означает, что составляются расписания от 3 до 9 заданий.

**Выводы.** В отличие от случая, когда задания не имеют весов приоритетов, для случая с весами приоритетов невозможно выбрать наилучший подход к сортировке. Вместо этого можно построить гиперэвристику, вмещающую все эти подходы к сортировке (то есть полную группу, где будут применять каждый подход). Использование параллелизации для одновременной обработки двух или даже четырёх алгоритмов сортировки особо не повлияет на время вычислений.

**Ключевые слова:** одномашинное планирование заданий с переключениями; общее взвешенное запаздывание; эвристика; подход к сортировке; остаточные периоды к обработке; остаточные имеющиеся ресурсы.

Рекомендована Радою
факультету прикладної математики
КПІ ім. Ігоря Сікорського

Надійшла до редакції
17 березня 2021 року

Прийнята до публікації
14 червня 2021 року