DOI: 10.20535/kpisn.2021.1.211935
UDC 519.161+519.687.1+004.023

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

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

Background. In preemptive job scheduling, which is a part of the flow-shop sequencing tasks, one of the most crucial goals is to obtain a schedule whose total tardiness would be minimal. Total 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 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 tardiness is 25 % greater than the minimum or even worse. Therefore, this heuristic needs a corrective branch which would further try to minimize total tardiness under certain conditions.
Objective. The goal is to ascertain what is to be corrected in the heuristic so that the total tardiness value could be obtained lesser. 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. In this scheduling problem, the inaccuracy of finding the minimal total tardiness has the strongest negative impact, so this is almost the worst case, which defines the accuracy limit of the heuristic and positively serves just as the principle of minimax guaranteeing decreasing losses in the worst conditions.
Methods. The heuristic sorts maximal decisive ratios by release dates, where the scheduling preference is given to the earliest job. To achieve the said goal, three other sorting approaches are presented and a computational study is carried out with applying each of the four heuristic approaches to minimize total tardiness. For this, two series of 266000 and 1064000 scheduling problems are generated.
Results. The earliest-job sorting ensures a heuristically minimal total tardiness value 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 it has been registered just a one scheduling problem (out of 31914 cases followed by a sole “win” of a heuristic version) whose minimal total tardiness is achieved by the earliest-job sorting.
Conclusions. The best heuristic version is that one which uses the sorting approach with minimizing remaining processing periods. This, however, is confirmed only for the case where jobs do not have any priorities. The case when jobs have their priority weights is to be yet analyzed.
Keywords: preemptive 1-machine job scheduling; total tardiness; heuristic; sorting approach; remaining processing periods; remaining available periods.

Introduction

Flow-shop sequencing optimization is a very important problem whose solutions determine high economic and human-resource performance in multistep processes of assembling, building, dispatching, manufacturing, etc. [1, 2]. In preemptive job scheduling, which is a part of the flow-shop sequencing tasks, one of the most crucial goals is to obtain a schedule whose total tardiness would be minimal [3, 4]. Total tardiness minimization is commonly reduced to solving a combinatorial problem. In particular, the well-known Boolean linear programming model [5, 6] is used for that. Nevertheless, struggling to compute a schedule with the exactly minimal total tardiness may become very resource-consuming (implying processor clock speed, memory space, and time of computations) even for a few jobs divided into more than 10 processing periods [6]. Thus, as the number of jobs and the numbers of their processing periods increase, the exact solution of the total tardiness minimization problem may become practically intractable [5, 7, 8]. The problem of tractability can be slightly stretched and strengthened by using an optimal substitute for infinity in the Boolean linear programming model [9] and re-arranging jobs for either job ascending order input or job descending order input [6, 10, 11]. However, these methods allow decreasing computation time only on average, after solving long series of thousands of job scheduling problems.

Along with obtaining an exact solution, there are a lot of heuristics allowing to find an approximate solution, which often coincides with the exact one and ensures thus the minimal total tardiness also [3, 8, 11, 12]. In general, the heuristics operate with the remaining available period [8, 12], remaining slack [3, 8, 11], and remaining processing period [5, 8, 11, 12]. A heuristic, in which the decisive ratio is the reciprocal of the maximum of a pair of the remaining processing period (RPP) and remaining available period (RAP) [8], is closely the best one [12, 13]. The accuracy of this heuristic (henceforward, let it be named the RPP-RAP heuristic) was studied in [8] on a pattern of tight-tardy progressive idling-free single machine preemptive (TPIF1MP) scheduling [11]. In the TPIF1MP scheduling, where the release dates are given in ascending order starting from 1 to the number of jobs, the inaccuracy of finding the minimal total tardiness has the strongest negative impact. Therefore, this is almost the worst case, which defines the accuracy limit of the RPP-RAP heuristic (and other heuristics as well) and positively serves just as the principle of minimax guaranteeing decreasing losses in the worst conditions (the maximum of unfavorable states) [8, 13, 14].

Article [8] ascertained that the RPP-RAP heuristic schedules 2 jobs always with the minimal total tardiness. In scheduling 3 to 7 jobs whose lengths (processing periods) are randomly generated within an integer interval from 2 to 5, the risk of missing the minimal total tardiness is just 1.5 % to 3.2 %. Moreover, it is expected that scheduling 12 and more jobs (divided into 2 to 5 processing periods) has at the most the same risk or even lower. In general, the RPP-RAP heuristic produces about 92 % schedules [8] whose total tardiness is exactly minimal, i.e. the Boolean linear programming model in about 92 % of the cases is needless.

In spite of such a promising performance, the RPP-RAP heuristic may produce schedules of a few jobs whose total tardiness is 25 % greater than the minimum or even worse. For instance, a problem of scheduling 4 jobs divided into 7, 7, 11, and 7 processing periods, respectively, whose due dates are 14, 20, 4, 13 by the progressive release dates 1, 2, 3, 4, has the total tardiness minimum of 29, whereas the RPP-RAP heuristic produces a schedule whose total tardiness is 37 (i.e., the relative gap [7, 8, 15] here is 27.5862 %, which is obviously unacceptable). Therefore, this heuristic needs a corrective branch which would further try to minimize total tardiness under certain conditions.

Problem statement

As the RPP-RAP heuristic has “weak” places, the goal is to ascertain which they are and try to correct them. For this, a few series of TPIF1MP scheduling problems will be generated, in which the “weak” places are to be analyzed. The “weakness” will be treated with respect to using RPP and RAP in an alternative way allowing to obtain a total tardiness lesser than that by the RPP-RAP heuristic. Eventually, the corrected and updated RPP-RAP heuristic (henceforward, let it be named the CU-RPP-RAP heuristic) will be discussed and the corresponding conclusions on it will be made.

The RPP-RAP heuristic and its corrections

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

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

release dates

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

and due dates

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

the RPP-RAP heuristic 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 [7, 8, 11]. Before the start, RPPs are

$$q_n = H_n \quad \forall\, n = \overline{1, N} . \tag{4}$$

For every set of available jobs

$$A(t) = \{i \in \{\overline{1, N}\} : r_i \leq t \text{ and } q_i > 0\} \subset \{\overline{1, N}\} \tag{5}$$

the respective RAPs are

$$b_i = \max\{0, d_i - t + 1\} \quad \forall\, i \in A(t) . \tag{6}$$

Then a subset

$$A^*(t) = \arg\max_{i \in A(t)} (\max\{q_i, b_i\})^{-1} \tag{7}$$

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

$$\tilde{s}_t = i^* \text{ by } A^*(t) = \{i^*\} \subset A(t) \subset \{\overline{1, N}\} \tag{8}$$

and the RPP of job i^* is updated as

$$q_{i^*}^{(\text{obs})} = q_{i^*} \text{ and } q_{i^*} = q_{i^*}^{(\text{obs})} - 1; \tag{9}$$

otherwise

$$A^*(t) = \{i_l^*\}_{l=1}^{L} \subset A(t) \subset \{\overline{1, N}\} \text{ by } L > 1, \tag{10}$$

whence

$$\tilde{s}_t = i_1^* \tag{11}$$

and the RPP of job i_1^* is updated as

$$q_{i_1^*}^{(\text{obs})} = q_{i_1^*} \text{ and } q_{i_1^*} = q_{i_1^*}^{(\text{obs})} - 1. \tag{12}$$

Assignment (11) executed by condition (10) implies that the earliest job is preferred to be scheduled [16] when there are two or more maximal decisive ratios in subset (7). Thus, job n is completed after moment $\tilde{\theta}(n; H_n)$ if

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

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

$$\tilde{\vartheta}(N) = \sum_{n=1}^{N} \max\{0, \tilde{\theta}(n; H_n) - d_n\} \tag{13}$$

is an approximately minimal total tardiness that corresponds to this schedule [8, 11, 16].

Thus, in the case when (10) is true, the RPP-RAP heuristic sorts maximal decisive ratios in subset (7) 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, other approaches to sorting might be used. They consist in sorting by RPPs and due dates. Consider them as follows.

If the RAP of job i^* is $b_{i^*} > 0$, then

$$\lambda_{i^*} = q_{i^*}^{-1} \text{ by } i^* \in \{i_l^*\}_{l=1}^{L}, \tag{14}$$

else

$$\lambda_{i^*} = d_{i^*}^{-1} \text{ by } i^* \in \{i_l^*\}_{l=1}^{L}. \tag{15}$$

Subsequently, a subset

$$F^{**}(t) = \arg\min_{i^* \in A^*(t)} \lambda_{i^*} = \{i_l^{**}\}_{l=1}^{L^*} \tag{16}$$

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

$$\tilde{s}_t = i_1^{**}, \tag{17}$$

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

$$q_{i_1^{**}}^{(\text{obs})} = q_{i_1^{**}} \text{ and } q_{i_1^{**}} = q_{i_1^{**}}^{(\text{obs})} - 1. \tag{18}$$

So, instead of (11) by (10), this approach corrects the RPP-RAP heuristic by scheduling the job with its either maximal RPP or maximal due date depending on the RAP. Henceforward, let it be named the RPP-or-due-date sorting. For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times 6} = [6 \;\; 2 \;\; 5 \;\; 3 \;\; 4 \;\; 4] \tag{19}$$

by $r_n = n \;\; \forall\, n = \overline{1, 6}$ and due dates

$$\mathbf{D} = [d_n]_{1\times 6} = [1 \;\; 6 \;\; 10 \;\; 4 \;\; 8 \;\; 4] \tag{20}$$

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

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 24} = [1 \;\; 2 \;\; 2 \;\; 4 \;\; 4 \;\; 4 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 6 \;\; 6 \;\; 6 \;\; 6 \;\; 3 \;\; 3 \;\; 3 \;\; 3 \;\; 3 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 1] \tag{21}$$

whose total tardiness is

$$\tilde{\vartheta}(6) = \sum_{n=1}^{6} \max\{0, \tilde{\theta}(n; H_n) - d_n\} =$$
$$\max\{0, 24-1\} + \max\{0, 3-6\} +$$
$$\max\{0, 19-10\} + \max\{0, 6-4\} +$$
$$\max\{0, 10-8\} + \max\{0, 14-4\} = 46. \tag{22}$$

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

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 24} = [1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 1 \;\; 2 \;\; 2 \;\; 4 \;\; 4 \;\; 4 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 6 \;\; 6 \;\; 6 \;\; 6 \;\; 3 \;\; 3 \;\; 3 \;\; 3 \;\; 3] \tag{23}$$

whose total tardiness is greater by 4 units:

$$\tilde{\vartheta}(6) = \sum_{n=1}^{6} \max\{0, \tilde{\theta}(n; H_n) - d_n\} =$$
$$\max\{0, 6-1\} + \max\{0, 8-6\} +$$
$$\max\{0, 24-10\} + \max\{0, 11-4\} +$$
$$\max\{0, 15-8\} + \max\{0, 19-4\} = 50. \tag{24}$$

This is the example of that the RPP-or-due-date sorting can outperform the earliest-job sorting.

An approach based on sorting only RPPs in ascending order can be used as well: a subset

$$H^{**}(t) = \arg\min_{i^* \in A^*(t)} q_{i^*} = \{i_l^{**}\}_{l=1}^{L^*} \tag{25}$$

is found, and job i_1^{**} is scheduled as (17) by (18). Henceforward, let it be named the min-RPP sorting. For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times4} = [2 \quad 3 \quad 2 \quad 2] \qquad (26)$$

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

$$\mathbf{D} = [d_n]_{1\times4} = [2 \quad 6 \quad 6 \quad 5] \qquad (27)$$

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

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times9} = [1 \quad 1 \quad 3 \quad 4 \quad 4 \quad 3 \quad 2 \quad 2 \quad 2] \qquad (28)$$

whose total tardiness is

$$\tilde{\vartheta}(4) = \sum_{n=1}^{4} \max\{0,\ \tilde{\theta}(n;\ H_n) - d_n\} =$$
$$\max\{0,\ 2-2\} + \max\{0,\ 9-6\} +$$
$$\max\{0,\ 6-6\} + \max\{0,\ 5-5\} = 3. \qquad (29)$$

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

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times9} = [1 \quad 1 \quad 2 \quad 4 \quad 4 \quad 2 \quad 2 \quad 3 \quad 3] \qquad (30)$$

whose total tardiness is greater by 1 unit:

$$\tilde{\vartheta}(4) = \sum_{n=1}^{4} \max\{0,\ \tilde{\theta}(n;\ H_n) - d_n\} =$$
$$\max\{0,\ 2-2\} + \max\{0,\ 7-6\} +$$
$$\max\{0,\ 9-6\} + \max\{0,\ 5-5\} = 4. \qquad (31)$$

This is the example of that the min-RPP sorting can outperform the earliest-job sorting as well as the RPP-or-due-date sorting can. It is worth noting that the RPP-or-due-date sorting for job lengths (26) and due dates (27) also produces schedule (30).

Finally, an approach based on just a random selection of a job from subset (7) will be tried. Thus, in the case when (10) is true, number m is randomly selected from subset $\{\overline{1,\ L}\}$ and job i_m^* is scheduled:

$$\tilde{s}_t = i_m^*, \qquad (32)$$

whereupon the RPP of job i_m^* is updated as

$$q_{i_m^*}^{(\text{obs})} = q_{i_m^*} \quad \text{and} \quad q_{i_m^*} = q_{i_m^*}^{(\text{obs})} - 1. \qquad (33)$$

However, if a schedule is built by using the random sorting by (32), (33), instead of (11) by (10), then such a corrected RPP-RAP heuristic will produce, generally speaking, random schedules and total tardiness values. Nevertheless, the random sorting can outperform the earliest-job sorting. For example, a TPIF1MP scheduling problem with

$$\mathbf{H} = [H_n]_{1\times12} =$$
$$[7 \quad 9 \quad 13 \quad 8 \quad 11 \quad 16 \quad 17 \quad 4 \quad 12 \quad 5 \quad 3 \quad 13] \qquad (34)$$

by $r_n = n \quad \forall\, n = \overline{1, 12}$ and due dates

$$\mathbf{D} = [d_n]_{1\times12} =$$
$$[11 \quad 11 \quad 16 \quad 12 \quad 9 \quad 1 \quad 5 \quad 16 \quad 19 \quad 18 \quad 12 \quad 31] \qquad (35)$$

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

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times118} = [1 \quad 2 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 8 \quad 8$$
$$11 \quad 11 \quad 11 \quad 8 \quad 8 \quad 10 \quad 10 \quad 10 \quad 10 \quad 10 \quad 2 \quad 2$$
$$2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 2 \quad 4 \quad 4 \quad 4 \quad 4 \quad 4 \quad 4$$
$$4 \quad 4 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5 \quad 5$$
$$5 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9 \quad 9$$
$$9 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 3$$
$$3 \quad 3 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12 \quad 12$$
$$12 \quad 12 \quad 12 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6$$
$$6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 6 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7$$
$$7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7 \quad 7] \qquad (36)$$

whose total tardiness is

$$\tilde{\vartheta}(12) = \sum_{n=1}^{12} \max\{0,\ \tilde{\theta}(n;\ H_n) - d_n\} =$$
$$\max\{0,\ 8-11\} + \max\{0,\ 28-11\} + \max\{0,\ 72-16\} +$$
$$\max\{0,\ 36-12\} + \max\{0,\ 47-9\} + \max\{0,\ 101-1\} +$$
$$\max\{0,\ 118-5\} + \max\{0,\ 15-16\} + \max\{0,\ 59-19\} +$$
$$\max\{0,\ 20-18\} + \max\{0,\ 13-12\} +$$
$$\max\{0,\ 85-31\} = 445. \qquad (37)$$

The other schedules have total tardiness values of 446, 447, 448, and 451. Thus, the random-sorting RPP-RAP heuristic produces here five possible total tardiness values whose probabilities are about 0.083, 0.177, 0.25, 0.334, and 0.156, respectively. It is worth noting that the RPP-or-due-date sorting for job lengths (34) and due dates (35) produces a schedule whose total tardiness is $\tilde{\vartheta}(12) = 451$, and the min-RPP sorting produces schedule

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 118} = [1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 4\ \ 4\ \ 4$$
$$11\ \ 11\ \ 11\ \ 8\ \ 8\ \ 8\ \ 8\ \ 4\ \ 4\ \ 4\ \ 4\ \ 4$$
$$10\ \ 10\ \ 10\ \ 10\ \ 10\ \ 2\ \ 2\ \ 2\ \ 2\ \ 2\ \ 2\ \ 2$$
$$2\ \ 2\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5\ \ 5$$
$$5\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9\ \ 9$$
$$9\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3$$
$$3\ \ 3\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12\ \ 12$$
$$12\ \ 12\ \ 12\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6$$
$$6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 6\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7$$
$$7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7\ \ 7] \quad (38)$$

whose total tardiness is 2 units greater than (37):

$$\tilde{\vartheta}(12) = \sum_{n=1}^{12} \max\{0,\ \tilde{\theta}(n;\ H_n) - d_n\} =$$
$$\max\{0,\ 7-11\} + \max\{0,\ 36-11\} + \max\{0,\ 72-16\} +$$
$$\max\{0,\ 22-12\} + \max\{0,\ 47-9\} + \max\{0,\ 101-1\} +$$
$$\max\{0,\ 118-5\} + \max\{0,\ 17-16\} + \max\{0,\ 59-19\} +$$
$$\max\{0,\ 27-18\} + \max\{0,\ 13-12\} +$$
$$\max\{0,\ 85-31\} = 447. \quad (39)$$

The RPP-RAP heuristic using the earliest-job sorting also produces schedule (38) which is outperformed by schedule (36). Consequently, each of the three described alternative approaches to sorting can outperform the earliest-job sorting. Along with that, one should remember that the random-sorting RPP-RAP heuristic may be outperformed as well in the same scheduling problem. So, stating that the random sorting produces a schedule or schedules with a definite total tardiness must be followed with a respective (approximate) probability.

Generation of TPIF1MP scheduling problems

In TPIF1MP scheduling, the release dates can be given in ascending order as follows:

$$r_n = n \quad \forall\, n = \overline{1,\ N}. \quad (40)$$

The due dates are generated as [11, 16]

$$d_n = r_n + H_n - 1 + b_n \quad \forall\, n = \overline{1,\ N} \quad (41)$$

by the respective random due date shift [10, 11]

$$b_n = \psi(H_n \cdot \zeta) \ \text{ for } \ n = \overline{1,\ N} \quad (42)$$

with a pseudorandom number ζ drawn from the standard normal distribution (with zero mean and unit variance), and function $\psi(\xi)$ returning the integer part of number ξ (e.g., see [10, 11]). The job lengths are generated as [8]

$$H_n = \psi(A\upsilon + 2) \ \text{ for } \ n = \overline{1,\ N} \ \text{ by } \ A = \overline{2,\ 20} \quad (43)$$

with a pseudorandom number υ [17] drawn from the standard uniform distribution on the open interval (0; 1). So, the job length is randomly generated between 2 and $A-1$ [8]. Once a vector of job lengths (1) is generated, due date shifts (42) are generated until

$$d_n \geq 1 \quad \forall\, n = \overline{1,\ N} \quad (44)$$

and the total tardiness value 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 (40) — (44), where 1000 scheduling problems are generated for every $N = \overline{2,\ 15}$ and $A = \overline{2,\ 20}$. So, altogether 266000 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 (10) — (12), the RPP-or-due-date sorting by (14) — (18), the min-RPP sorting by (25) with (17) by (18), and the random sorting by (32), (33).

Table 1 presents the number of generated scheduling problems whose total tardiness has been revealed to be minimal for the given sorting approach, whereas the other three sorting approaches have produced greater total tardiness values (i.e., the heuristic with the given sorting approach "has won"). Amazingly enough, neither the earliest-job sorting, nor the RPP-or-due-date sorting "has won" at all, whereas the min-RPP sorting "has won" 6486 times; the random sorting has only one "win" with an approximate probability of 0.083 for job lengths

***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"	0	0	6486	1

$$\mathbf{H} = [H_n]_{1\times 12} = [2 \;\; 6 \;\; 4 \;\; 8 \;\; 8 \;\; 5 \;\; 3 \;\; 7 \;\; 2 \;\; 3 \;\; 8 \;\; 6] \quad (45)$$

and due dates

$$\mathbf{D} = [d_n]_{1\times 12} = [4 \;\; 4 \;\; 8 \;\; 20 \;\; 32 \;\; 1 \;\; 12 \;\; 12 \;\; 9 \;\; 10 \;\; 32 \;\; 20], \quad (46)$$

where

$$\tilde{\mathbf{S}} = [\tilde{s}_t]_{1\times 62} = [1 \;\; 1 \;\; 2 \;\; 3 \;\; 3 \;\; 3 \;\; 3 \;\; 7 \;\; 9 \;\; 9 \;\; 7 \;\; 7 \;\; 10 \;\; 10 \;\; 10 \;\; 2 \;\; 2 \;\; 2 \;\; 2 \;\; 2 \;\; 6 \;\; 6 \;\; 6 \;\; 6 \;\; 6 \;\; 12 \;\; 12 \;\; 12 \;\; 12 \;\; 12 \;\; 12 \;\; 8 \;\; 8 \;\; 8 \;\; 8 \;\; 8 \;\; 8 \;\; 8 \;\; 4 \;\; 4 \;\; 4 \;\; 4 \;\; 4 \;\; 4 \;\; 4 \;\; 4 \;\; 11 \;\; 11 \;\; 11 \;\; 11 \;\; 11 \;\; 11 \;\; 11 \;\; 11 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 5 \;\; 5] \quad (47)$$

is that "winning" schedule whose total tardiness is

$$\tilde{\vartheta}(12) = \sum_{n=1}^{12} \max\{0, \tilde{\theta}(n; H_n) - d_n\} = \max\{0, 2-4\} + \max\{0, 20-4\} + \max\{0, 7-8\} + \max\{0, 46-20\} + \max\{0, 62-32\} + \max\{0, 25-1\} + \max\{0, 12-12\} + \max\{0, 38-12\} + \max\{0, 10-9\} + \max\{0, 15-10\} + \max\{0, 54-32\} + \max\{0, 31-20\} = 161. \quad (48)$$

The other two total tardiness values of 162 and 163 by the random sorting for (45), (46) have probabilities of 0.5 and 0.417, respectively. The min-RPP sorting for (45), (46) produces a schedule whose total tardiness is 162 being just 0.6211 % worse than the random-sorting 8.3 % probable schedule (47).

The 6487 sole "wins" in the first series constitute just 2.4387 % of the series volume. The remaining 97.5613 % are group "wins", where two to four sorting approaches have produced the same (minimal) total tardiness. Table 2 presents the distribution of the number of group "wins", where seven "winning" groups have been disclosed. The highest and dominating percentage (93.4836 %) of group "wins" has been revealed to be for the group consisting of all four sorting approaches. Moreover, every group contains the min-RPP sorting which is the incontestable "winner" in the consideration of sole "wins" (see Table 1). This implies that the min-RPP sorting itself is capable of producing a total tardiness value, which either is minimal among the total tardiness values by the other three sorting approaches or is not greater (this is the 97.5613 % of the 266000 cases in the first series) than these values.

To get convinced that the obtained results are statistically reliable, a second series of TPIF1MP scheduling problems is analogously generated by (40) – (44), where 4000 scheduling problems are generated for every $N = \overline{2, 15}$ and $A = \overline{2, 20}$. Table 3 presents the number of generated scheduling problems (among those 1064000 generated scheduling problems) whose total tardiness has been revealed to be minimal for the given sorting approach, whereas the other three sorting approaches have produced greater total tardiness values. As in the case of the first series, here the number of sole "wins" constitute just a small percentage (which is 2.3901 %) of the series volume, whereas the min-RPP sorting "has won" almost everywhere, expect for three scheduling problems. Now the random sorting "has won" two times for scheduling problems with

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

Group of sorting approaches which produce the same minimal total tardiness	Earliest-job sorting	–	–	Earliest-job sorting	Earliest-job sorting	–	Earliest-job sorting
	–	RPP-or-due-date sorting	–	RPP-or-due-date sorting	–	RPP-or-due-date sorting	RPP-or-due-date sorting
	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting
	–	–	Random sorting	–	Random sorting	Random sorting	Random sorting
Group "wins"	5222	587	4831	662	4961	648	242602
Percentage of "wins"	2.0122	0.2262	1.8616	0.2551	1.9117	0.2497	93.4836

$$\mathbf{H} = [H_n]_{1\times13} = [15\ \ 15\ \ 11\ \ 3\ \ 10\ \ 13\ \ 12\ \ 19\ \ 4\ \ 8\ \ 11\ \ 9\ \ 17], \qquad (49)$$

$$\mathbf{D} = [d_n]_{1\times13} = [16\ \ 4\ \ 30\ \ 6\ \ 19\ \ 14\ \ 19\ \ 37\ \ 9\ \ 21\ \ 22\ \ 24\ \ 30] \qquad (50)$$

and

$$\mathbf{H} = [H_n]_{1\times14} = [5\ \ 9\ \ 4\ \ 7\ \ 10\ \ 12\ \ 16\ \ 17\ \ 5\ \ 6\ \ 13\ \ 4\ \ 4\ \ 13], \qquad (51)$$

$$\mathbf{D} = [d_n]_{1\times14} = [14\ \ 4\ \ 10\ \ 13\ \ 4\ \ 46\ \ 31\ \ 26\ \ 14\ \ 15\ \ 18\ \ 18\ \ 18\ \ 51] \qquad (52)$$

producing the respective schedules by $\tilde{\vartheta}(13) = 597$ for (49), (50) and $\tilde{\vartheta}(14) = 437$ for (51), (52), and the earliest-job sorting has one "win" for a scheduling problem with

$$\mathbf{H} = [H_n]_{1\times14} = [8\ \ 2\ \ 11\ \ 9\ \ 3\ \ 10\ \ 11\ \ 7\ \ 13\ \ 5\ \ 7\ \ 10\ \ 8\ \ 7], \qquad (53)$$

$$\mathbf{D} = [d_n]_{1\times14} = [16\ \ 4\ \ 11\ \ 14\ \ 9\ \ 17\ \ 39\ \ 16\ \ 17\ \ 14\ \ 23\ \ 36\ \ 17\ \ 4] \qquad (54)$$

producing the respective schedule by $\tilde{\vartheta}(14) = 453$. It is worth noting that the min-RPP sorting for job lengths (49) and due dates (50) produces a schedule whose total tardiness is $\tilde{\vartheta}(13) = 601$, and $\tilde{\vartheta}(14) = 438$ for job lengths (51) and due dates (52), i.e. the min-RPP sorting "loses" to the random sorting just 0.67 % and 0.2288 %, respectively. Moreover, for job lengths (53) and due dates (54), the min-RPP sorting produces a schedule whose total tardiness is $\tilde{\vartheta}(14) = 454$ "losing" thus 0.2208 % to the earliest-job sorting.

***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"	1	0	25428	2

Table 4 presents the distribution of the number of group "wins", where the same seven "winning" groups have been disclosed in the second series. The group "wins" do not much differ from those in Table 2 (for a working accuracy, see the bottom row of Table 4). This confirms the statistical reliability of the results in Tables 1 and 2. Therefore, the min-RPP sorting is really the incontestable "winner" in minimizing total tardiness of TPIF1MP scheduling problems generated by (40) — (44).

A special case when all the jobs have identical volumes is to be studied also. For this, a series of TPIF1MP scheduling problems of N jobs, each of which is of A processing periods, is generated by (40) — (42) and (44), where 1000 scheduling problems are generated for every $N = \overline{3, 15}$ and $A = \overline{2, 10}$. So, altogether 117000 scheduling problems are generated in this series. There are no sole "wins", and the group "wins" are distributed only among four groups (Table 5). Once again, the highest and dominating percentage (96.2197 %) of group "wins" has been revealed to be for the group consisting of all four sorting approaches. As for the first and second series of non-constant job length scheduling problems, every "winning" group in the series of constant job length scheduling problems contains the min-RPP sorting. This additionally confirms the min-RPP sorting is very efficient.

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

Group of sorting approaches which produce the same minimal total tardiness	Earliest-job sorting	–	–	Earliest-job sorting	Earliest-job sorting	–	Earliest-job sorting
	–	RPP-or-due-date sorting	–	RPP-or-due-date sorting	–	RPP-or-due-date sorting	RPP-or-due-date sorting
	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting
	–	–	Random sorting	–	Random sorting	Random sorting	Random sorting
Group "wins"	20839	2474	18742	2619	19587	2395	971913
Percentage of "wins"	2.0065	0.2382	1.8046	0.2522	1.8860	0.2306	93.5819
Relative difference from the first series, %	0.2851	5.0457	3.1567	1.1576	1.3626	8.2794	0.1051

Table 5. The number of group "wins" in the series of constant job length scheduling problems

Group of sorting approaches which produce the same minimal total tardiness	Earliest-job sorting	Earliest-job sorting	Earliest-job sorting	Earliest-job sorting
	–	RPP-or-due-date sorting	–	RPP-or-due-date sorting
	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting	Min-RPP sorting
	–	–	Random sorting	Random sorting
Group "wins"	1997	216	2210	112577
Percentage of "wins"	1.7068	0.1846	1.8889	96.2197

Discussion

One of the "weak" places in the RPP-RAP heuristic by (4) – (12) is the earliest-job sorting. Despite it ensures a heuristically minimal total tardiness value in more than 97.6 % of TPIF1MP scheduling problems (which is derived from the respective group "wins" in Tables 2 and 4), the earliest-job sorting still fails to minimize total tardiness in no less than 2.2 % of the cases (this is a lower bound of such an estimate). On the other hand, the min-RPP sorting produces a heuristically minimal total tardiness for almost any TPIF1MP scheduling problem, where exceptions like (45) – (54) are quite rare (technically, this has been just 4 occurrences out of 31914 cases followed by sole "wins", i.e. the exception is only 0.0125 % probable). All the more that the min-RPP sorting "loses" to the other sorting approaches very little – it has not been revealed a difference greater than 0.75 %, although there have been registered only 5 such occurrences including the case with job lengths (34) and dues dates (35). Moreover, if to consider non-random sorting approaches, it has been registered just a one TPIF1MP scheduling problem whose minimal total tardiness is achieved by the earliest-job sorting.

Apart from the min-RPP sorting, the other "weak" places in the RPP-RAP heuristic are tied to the RPP-or-due-date sorting and the random sorting. The RPP-or-due-date sorting cannot "win" solely, and "wins" of the random sorting are very rare and unreliable. Consequently, these sorting approaches cannot be used in minimizing total tardiness of TPIF1MP scheduling problems. On the contrary, the min-RPP sorting successfully corrects almost any fails of the RPP-RAP heuristic. Therefore, the CU-RPP-RAP heuristic can be defined as the RPP-RAP heuristic by (4) – (10) using the min-RPP sorting approach as (25) with (17) by (18).

Despite the CU-RPP-RAP heuristic is not perfect against the RPP-RAP heuristic (using the earliest-job sorting), it is impossible to figure out which processing periods (1) and due dates (3) should be to have the least total tardiness value produced solely by the earliest-job sorting approach. The task of searching for such a pattern was simplified with setting release dates (2) as (40), but the pattern is not clear. This pattern does possibly exist but it is too tricky if even the simplification has not given a primitive sketch of it. Nevertheless, this does not really matter owing to the highly efficient min-RPP sorting.

Conclusions

To minimize total tardiness in TPIF1MP scheduling by the heuristic based on RAPs and RPPs, it is the best to use the min-RPP sorting instead of the earliest-job sorting to schedule jobs when two or more maximal decisive ratios become equal. It is plausible that the respective CU-RPP-RAP heuristic is not the best itself, but it generally outperforms or equal to the heuristic if the earliest-job sorting, the RPP-or-due-date sorting, and the random sorting are used. Therefore, the CU-RPP-RAP heuristic using the min-RPP sorting is the best considering the RPP-RAP heuristic with the other three sorting approaches in TPIF1MP scheduling. However, the research on the RPP-RAP heuristic should be furthered for the case when jobs have their priority weights, where the sorting improvement is to be yet analyzed.

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

[3] 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

[4] 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

[5] 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

[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, "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

[8] 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

[9] 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 V.N. Karazin Kharkiv National University*, ser. MIA, vol. 44, pp. 94–101, 2019. doi: 10.26565/2304-6201-2019-44-10

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

[11] 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

[12] 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

[13] R.L. Graham *et al.*, "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey," in *Annals of Discrete Mathematics*, vol. 5, P.L. Hammer *et al.*, Eds. B.C., Canada: North-Holland, 1979, pp. 287–326. doi: 10.1016/S0167-5060(08)70356-X

[14] 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

[15] W.-Y. Ku and J. C. Beck, "Mixed Integer Programming models for job shop scheduling: A computational analysis," *Comput. & Operat. Res*, vol. 73, pp. 165–173, 2016. doi: 10.1016/j.cor.2016.04.006

[16] 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

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

В.В. Романюк

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

Проблематика. У плануванні завдань із перемиканнями, яке є частиною задач упорядковування на виробництві, однією з ключових цілей є отримання розкладу, загальне запізнювання якого було б мінімальним. Мінімізація загального запізнювання зазвичай зводиться до розв'язання комбінаторної задачі, що стає практично нерозв'язною, щойно кількість завдань і кількості їх періодів до обробки зростають. Щоб упоратися з цією проблемою, використовують евристики. Близькою до найкращої є евристика, у якій вирішальним співвідношенням є обернене значення максимуму пари залишкового періоду до обробки та залишкового наявного ресурсу. Однак ця евристика може складати розклади декількох робіт, загальне запізнювання яких є на 25 % більшим за мінімум, або й гірше. Тому ця евристика потребує коректувальної гілки, яка б надалі намагалася мінімізувати загальне запізнювання за певних умов.

Мета дослідження. Встановити, що саме має бути скориговано в евристиці так, щоб значення загального запізнювання зменшилось. Евристика буде застосована до щільного поступального одномашинного планування з перемиканнями без простою, у якому моменти запуску завдань подаються у порядку зростання від 1 до кількості завдань, а моменти прийняття виконання завдань встановлюються щільно за моментами запуску. Неточність знаходження мінімального загального запізнювання у цій задачі планування має найбільш негативний вплив, тому вона є майже найгіршим випадком, який визначає межу точності евристики та безпосередньо слугує принципом мінімаксу, гарантуючи зменшення втрат за найгірших умов.

Методика реалізації. Евристика сортує максимальні вирішальні співвідношення за моментами запуску завдань, віддаючи перевагу найбільш раннім завданням. Для досягнення зазначеної мети представляються три інші підходи до сортування та проводиться обчислювальне дослідження із застосуванням кожного з чотирьох евристичних підходів для мінімізації загального запізнювання. Для цього генеруються дві послідовності з 266000 і 1064000 задач планування.

Результати дослідження. Підхід із сортуванням найбільш ранніх завдань забезпечує евристично мінімальне значення загального запізнювання у понад 97,6 % задач планування, але цей підхід не може мінімізувати загальне запізнювання у не менш ніж 2,2 % випадків. А втім, підхід із сортуванням за мінімізацією залишкових періодів до обробки виробляє евристично мінімальне загальне запізнювання майже для будь-якої задачі планування. Попри можливі винятки, цей підхід із сортуванням “програє” іншим підходам дуже мало. Крім того такі винятки є надзвичайно рідкісними, оскільки було зафіксовано лише одну задачу планування (з 31914 випадків, відзначених одноособовими “перемогами” евристик), мінімальне загальне запізнювання якої досягається за підходом із сортуванням найбільш ранніх завдань.

Висновки. Найкращою версією евристики є використання підходу із сортуванням за мінімізацією залишкових періодів до обробки. Однак наразі це підтверджується для випадку, в якому завдання не мають жодних пріоритетів. Випадок, за якого завдання мають ваги своїх пріоритетів, ще потрібно проаналізувати.

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

В.В. Романюк

УСОВЕРШЕНСТВОВАНИЕ СОРТИРОВОК В ЭВРИСТИКЕ НА ОСНОВЕ ОСТАТОЧНЫХ ИМЕЮЩИХСЯ РЕСУРСОВ И ПЕРИОДОВ ДО ОБРАБОТКИ ДЛЯ МИНИМИЗАЦИИ ОБЩЕГО ЗАПАЗДЫВАНИЯ В ПРОГРЕССИРУЮЩЕМ ОДНОМАШИННОМ ПЛАНИРОВАНИИ С ПЕРЕКЛЮЧЕНИЯМИ БЕЗ ПРОСТОЯ

Проблематика. В планировании заданий с переключениями, которое является частью задач упорядочения на производстве, одной из ключевых целей является получение расписания, общее запаздывание которого было бы минимальным. Обычно минимизация общего запаздывания сводится к решению комбинаторной задачи, которая становится практически неразрешимой, как только количество заданий и количества их периодов до обработки возрастают. Чтобы решить эту проблему, используют эвристики. Близкой к наилучшей является эвристика, в которой за решающее соотношение берут обратную величину максимума пары остаточного периода до обработки и остаточного имеющегося ресурса. Однако эта эвристика может создавать расписания нескольких заданий, чьё общее запаздывание на 25 % больше минимума, или даже хуже. Поэтому эта эвристика требует корректировочной ветви, которая бы в дальнейшем пыталась минимизировать общее запаздывание в определённых условиях.

Цель исследования. Установить, что именно должно быть скорректировано в эвристике так, чтобы значение общего запаздывания стало меньшим. Эвристика будет применена к плотному прогрессирующему одномашинному планированию с переключениями без простоя, в котором моменты запуска заданий подаются в порядке возрастания от 1 до количества заданий, а моменты приёма выполнения заданий устанавливаются плотно по моментам запуска. Неточность нахождения минимального общего запаздывания в этой задаче планирования имеет наиболее негативное влияние, поэтому она является практически наихудшим случаем, определяющим границу точности эвристики, и непосредственно выступает в качестве принципа минимакса, гарантируя уменьшение потерь в наихудших условиях.

Методика реализации. Эвристика сортирует максимальные решающие соотношения по моментам запуска заданий, отдавая предпочтение наиболее ранним заданиям. Для достижения указанной цели представляются три других подхода к сортировке и проводится вычислительное исследование с применением каждого из четырёх эвристических подходов для минимизации общего запаздывания. Для этого генерируются две последовательности из 266000 и 1064000 задач планирования.

Результаты исследования. Подход с сортировкой наиболее ранних заданий обеспечивает эвристически минимальное значение общего запаздывания в более чем 97,6 % задач планирования, но этот подход не может минимизировать общее запаздывание в не менее чем 2,2 % случаев. Тем не менее, подход с сортировкой по минимизации остаточных периодов к обработке производит эвристически минимальное общее запаздывание почти для любой задачи планирования. Вопреки возможным исключениям, этот подход с сортировкой “проигрывает” другим подходам очень мало. Более того, такие исключения чрезвычайно редки, поскольку было зафиксировано лишь одну задачу планирования (из 31914 случаев, отмеченных единоличными “победами” эвристик), минимальное общее запаздывание которой достигается по подходу с сортировкой наиболее ранних заданий.

Выводы. Наилучшей версией эвристики является использование подхода с сортировкой по минимизации остаточных периодов к обработке. Однако пока это подтверждается для случая, в котором задания не обладают какими-либо приоритетами. Случай, в котором задания наделены весами своих приоритетов, ещё предстоит проанализировать.

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

Рекомендована Радою
факультету прикладної математики
КПІ ім. Ігоря Сікорського

Надійшла до редакції
21 вересня 2020 року

Прийнята до публікації
29 березня 2021 року