ISSN 1064-2307, Journal of Computer and Systems Sciences International, 2006, Vol. 45, No. 6, pp. 943–949. © Pleiades Publishing, Inc, 2006.
Original Russian Text © A.A. Lazarev, 2006, published in Izvestiya Akademii Nauk. Teoriya i Sistemy Upravleniya, 2006, No. 6, pp. 103–110.



# The Pareto-Optimal Set of the NP-Hard Problem of Minimization of the Maximum Lateness for a Single Machine

**A. A. Lazarev**

*Computing Center, Russian Academy of Sciences, ul. Vavilova 40, GSP-1, Moscow, 119991 Russia*

Received May 18, 2006

**Abstract**—The classical problem of scheduling theory that is NP-hard in the strong sense $1|r_j|L_{\max}$ is considered. New properties of optimal schedules are found. A polynomially resolved case of the problem is selected, when the release times ($r_j$), the processing time ($p_j$), and due dates of completion of processing ($d_j$) of jobs satisfy the constraints $d_1 \le \ldots \le d_n$ and $d_1 - r_1 - p_1 \ge \ldots \ge d_n - r_n - p_n$. An algorithm of run time $O(n^3 \log n)$ finds Pareto-optimal sets of schedules according to the criteria $L_{\max}$ and $C_{\max}$ that contains no more than $n$ variants.

**DOI:** 10.1134/S1064230706060098

## 0. INTRODUCTION

We consider the following problem of scheduling theory. On a machine starting from the moment $t$ (the moment of machine release), it is necessary to process jobs of the set $N = \{1, 2, \ldots, n\}$. Simultaneous processing and interrupts in processing jobs are prohibited. For jobs of the set $N$, we denote by $r_j$ the minimum possible moment of the beginning of processing, by $p_j > 0$ the processing time, and by $d_j$ the due date of completion of the job $j \in N$.

Each permutation $\pi$ of the jobs of the set $N$ uniquely determines an *early schedule* such that artificial idle times of the machine are eliminated. In an early schedule, each job $j \in N$ begins to be processed immediately after the completion of processing of the preceding job in the corresponding schedule. If the time of completion of processing of the preceding job is less than the release time for processing ($r_j$) of the current job, then the beginning of the processing of the job $j$ is delayed up to the release time. In what follows, all considered schedules are supposed to be early. The set of possible early schedules corresponding to the set $N$ is denoted by $\Pi(N)$. Note that the number of schedules $\pi \in \Pi(N)$ is equal to the number of permutations of $n$ elements, i.e., $|\Pi(N)| = n!$.

By $c_j(\pi)$, we denote the *moment of completion* of processing the job $j \in N$ in the schedule $\pi \in \Pi(N)$. If $\pi = (j_1, j_2, \ldots, j_n)$ then we have

$$c_{j_1}(\pi) = \max\{r_{j_1}t\} + p_{j_1},$$

$$c_{j_k}(\pi) = \max\{c_{j_{k-1}}(\pi), r_{j_k}\} + p_{j_k}, \ k = 2, \ldots, n.$$

The dependence $L_j(\pi) = c_j(\pi) - d_j$ reflects the *lateness* of the jobs $j \in N$ in the schedule $\pi$. *The maximum lateness* of jobs of the set $N$ under the schedule $\pi$ is determined as

$$L_{\max}(\pi) = \max_{j \in N}\{c_j(\pi) - d_j\}.$$

Let us denote the moment of completion of processing all jobs of the set $N$ under the schedule $\pi$ by

$$C_{\max}(\pi) = \max_{j \in N} c_j(\pi).$$

In the case when the schedule $\pi$ is considered from the moment $t' > t$, we use the notation $L_{\max}(\pi, t')$, $C_{\max}(\pi, t')$ and $c_j(\pi, t')$.

The problem consists in finding an *optimal* schedule $\pi^* \in \Pi(N)$ that corresponds to the minimum value of the goal function

$$L_{\max}(\pi^*) = \max_{j \in N} L_j(\pi) = \max_{j \in N}\{c_j(\pi) - d_j\}. \quad (0.1)$$

In the commonly accepted notation of scheduling theory introduced by Graham and others [1], this problem is written as $1|r_j|L_{\max}$. Intensive work for its solution has been conducted since the early 1950s. Lenstra and others [2] showed that the general case of problem $1|r_j|L_{\max}$ is NP-hard in the strong sense.

A series of polynomially resolved cases of the problem have been found, beginning with the Jackson result [3] for the variant $r_j = 0$, ®$j \in N$, when the schedule for which the jobs are arranged in nondecreasing order of the due dates of completion of processing (by the EDD rule) is a solution. This schedule is also optimal for the case when the release times and the due dates are coordinated ($r_i \le r_j \Leftrightarrow d_i \le d_j$, $\forall i, j \in N$).

Potts [4] presented an iterative version of the extended Jackson rule (IJ) and proved that $L_{\max}(IJ)/L^*_{\max} \le \frac{3}{2}$. Hall and Shmoys [5] modified the iterative version and developed algorithms (MIJ) that guarantee that $L_{\max}(MIJ)/L^*_{\max} \le \frac{4}{3}$. They also proposed two approximation schemes that guarantee that ε-approximate solution can be found for $O(n\log n + n(1/\varepsilon)^{O(1/\varepsilon^2)}$ and $O((n/\varepsilon)^{O(1/\varepsilon)})$ operations. Mastrolilli

[6] developed an improved and approximation scheme that can be executed for the time $O(n + (1/\varepsilon)^{O(1/\varepsilon)})$.

Special cases $1|$ prec; $r_j|C_{\max}$, $1|$prec; $p_j = p$; $r_j|L_{\max}$ and $1|$prec; $r_j$; pmtn$|L_{\max}$ with constraints of precedence in processing were considered in papers by Lawler [7], Simons [8], Baker [9], and others. Hoogeven [10] substantiated a polynomial algorithm (with the run time of $O(n^2 \log n)$ operations) for a special case in which the parameters of jobs satisfy the constraints $d_j - p_j - A \le r_j \le d_j - A$ for a certain constant $A$, ®$j \in N$. A pseudo-polynomial algorithm for the NP-hard case, when the release times are agreeable to the due dates ($d_1 \le \ldots \le d_n$ и $r_1 \ge \ldots \ge r_n$), was developed by Lazarev and Shul'gina [11, 12]. The run time of the algorithm is $O(nP(n + p_{\max}))$, where $P = \sum_{j \in N} p_j$ and $p_{\max} = \max_{j \in N} p_j$. The results presented in this paper give a generalization of the solutions obtained in [13].

Together with the problem $1|r_j|L_{\max}$, we consider in this paper the problem of constructing a Pareto-optimal set of schedules according to the criteria $C_{\max}$ and $L_{\max}$. We formulate an algorithm for constructing the set of schedules $\Phi(N, t) = \{\pi'_1, \pi'_2, \ldots, \pi'_m\}$ for which we have

$$C_{\max}(\pi'_1) < C_{\max}(\pi'_2) < \ldots < C_{\max}(\pi'_m),$$

$$L_{\max}\pi'_1) > L_{\max}\pi'_2 > \ldots > L_{\max}(\pi'_m).$$

## 1. PROPERTIES OF PROBLEMS

Let us denote the fact that the job $i$ precedes the job $j$ in the schedule $\pi$ by $(i \longrightarrow j)_\pi$. We also introduce the notation

$$r_j = \max\{r_j, t\}, \forall j \in N, \tag{1.1}$$

$$r(N, t) = \min_{j \in N}\{r(t)\}. \tag{1.2}$$

In the cases when it is clear with which set we deal, we simply write $r(N, t)$ instead of $r(t)$.

Assume that $\pi$ is the schedule of processing jobs of a certain subset from $N$, for defining which we use the notation $\{\pi\}$ in what follows.

It is supposed that the parameters of jobs satisfy the constraints

$$d_1 \le \ldots \le d_n,\ d_1 - r_1 - p_1 \ge \ldots \ge d_n - r_n - p_n. \tag{1.3}$$

For example, the case when $d_j = r_j + p_j + z$, $j = 1, \ldots, n$, where $z$ is a constant, corresponds to these constraints, i.e., when all jobs have the same time margin (resource) for their due date.

Assume that $|N| > 1$ and $t$ is the time when the machine is released. Let us select from the set $N$ two jobs $f = f(N, t)$ and $s = s(N, t)$ as follows:

$$f(N, t) = \arg\min_{j \in N}\{d_j | r_j(t) = r(N, t)\} \tag{1.4}$$

$$s(N, t) = \arg\min_{j \in N \backslash f}\{d_j | r_j(t) = r(N \backslash f, t)\} \tag{1.5}$$

where $f = f(N, t)$. If $N = \{i\}$, then we set $f(N, t) := i$ and $s(N, t) := 0$, ®$t$. We also define $d_0 = +\infty$, $f(\emptyset, t) = 0$ and $s(\emptyset, t) = 0$, $\forall t$. For jobs $f$ and $s$, the following properties are valid.

**Lemma 1.** If, for jobs of the set $N$, (1.3) holds, then, for any schedule $\pi \in \Pi(N)$ for all $j \in N \backslash \{f\}$ for which $(j \longrightarrow f)_\pi$, we have

$$L_j(\pi) < L_f(\pi) \tag{1.6}$$

and for all $j \in N \backslash \{f, s\}$ satisfying the condition $(j \longrightarrow s)_\pi$, we have

$$L_j(\pi) < L_s(\pi), \tag{1.7}$$

where $f = f(N, t)$ and $s = s(N, t)$.

**Proof.** For all jobs $j$ such that$(j \longrightarrow f)_\pi$, we have $c_j(\pi) < c_f(\pi)$. If $d_j \ge d_f$, then we obviously have

$$L_j(\pi) = c_j(\pi) - d_j < c_f(\pi) - d_f = L_f(\pi),$$

therefore, (1.6) holds.

Assume that for the job $j \in N$, $(j \longrightarrow f)_\pi$ we have $d_j < d_f$. Then, we have $r_j > r_f$. If $r_j \le r_f$, then $r_j(t) \le r_f(t)$ and $r_f(t) = r(t)$ as follows from (1.1) and (1.4). Then, $r_f(t) = r(t)$ and $d_j < d_f$, but it contradicts the definition of job $f$ (1.4). Therefore, we have $r_j > r_f$. It is obvious that $c_j(\pi) - p_j < c_f(\pi) - p_f$, and, since $r_j > r_f$, we have

$$c_j(\pi) - p_j - r_j < c_f(\pi) - p_f - r_f,$$

$$c_j(\pi) - c_f(\pi) < p_j + r_j + p_f - r_f.$$

If $d_j < d_f$, then (1.3) implies that either $d_j - r_j - p_j \ge d_f - r_f - p_f$ or $d_j - d_f \ge r_j + p_j - r_f - p_f$; therefore, $c_j(\pi) - c_f(\pi) < p_j + r_j - p_f - r_f \le d_j - d_f$. This implies that $L_j(\pi, t) < L_f(\pi, t)$ for all jobs $j$, $(j \longrightarrow f)_\pi$.

Let us prove inequality (1.7). For all jobs $j$ satisfying condition $(j \longrightarrow s)_\pi$, we have $c_j(\pi) < c_s(\pi)$. If $d_j \ge d_s$, then $L_j(\pi, t) = c_j(\pi) - d_j < c_s(\pi) - d_s = L_s(\pi, t)$; therefore, (1.7) holds.

Assume that, for job $j \in N \backslash \{f\}$, $(j \longrightarrow s)_\pi$, $d_j < d_s$is valid; then, we have $r_j > r_s$. In fact, if we suppose that $r_j \le r_s$, then $r_j(t) \le r_s(t)$, which follows from (1.1). Moreover, for the job $s$, we have $r_s(t) \ge r(t)$ according to definitions (1.2) and (1.5). If $r_s(t) = r(t)$, then, for $j$ and $s$, we can write $r_j(t) = r_s(t) = r(t)$ and $d_j < d_s$, which contradicts definition (1.5) of the job $s(N, t)$. If $r_s(t) > r(t)$, i.e.,

$r_s > r(t)$, then there is no job $i \in N\backslash\{f, s\}$ such that $r_s > r_i > r(t)$. Hence, for jobs $j$ and $s$, we obtain $r_j(t) = r_s(t)$ and $d_j < d_s$, which contradicts definition (1.5) of the job $s(N, t)$. Therefore, we have $r_j > r_s$.

Since $c_j(\pi) \leq c_s(\pi) - p_s$ and $p_j >$ , we have $c_j(\pi) - p_j < c_s(\pi) - p_s$, and, since $r_j > r_s$, we have $c_j(\pi) - p_j - r_j < c_s(\pi) - p_s - r_s$ and

$$c_j(\pi) - c_s(\pi) < p_j + r_j + p_s - r_s. \qquad (1.8)$$

Since $d_j < d_s$, (1.3) implies that either $d_j - r_j - p_j \geq d_s - r_s - p_s$or

$$c_j(\pi) - c_s(\pi) < p_j + r_j + p_s - r_s \leq d_j - d_s. \qquad (1.9)$$

This implies that $L_j(\pi) < L_s(\pi)$ for all jobs $j \in N\backslash\{f\}$, $(j \longrightarrow s)_\pi$.

**Theorem 1.** If, for all jobs of the subset $N' \subseteq N$, (1.3) holds, then for any moment $t' \geq t$ and every early schedule $\pi \in \Pi(N')$ there is $\pi' \in \Pi(N')$ such that

$$\begin{aligned} &L_{\max}(\pi', t') \leq L_{\max}(\pi, t'), \\ &C_{\max}(\pi', t') \leq C_{\max}(\pi, t') \end{aligned} \qquad (1.10)$$

and, under the schedule $\pi'$, the job $f = f(N', t')$ is processed first or $s = s(N', t')$. If $d_f \leq d_s$, then, under the schedule $\pi'$, the job $f$ is processed first.

**Proof.** Let $\pi = (\pi_1, f, \pi_2, s, \pi_3)$, where $\pi_1$, $\pi_2$, and $\pi_3$ are partial schedules of $\pi$. Let us construct the schedule $\pi' = (f, \pi_1, \pi_2, s, \pi_3)$. From definitions (1.1), (1.2), and (1.4), we obtain $r_f(t') \leq r_j(t')$, $j \in N'$. This implies that $C_{\max}((f, \pi_1), t') \leq C_{\max}((\pi_1, f), t')$ and

$$C_{\max}(\pi', t') \leq C_{\max}(\pi, t'), \qquad (1.11)$$

$$L_j(\pi', t') \leq L_j(\pi, t'),\ \forall j \in \{(\pi_2, s, \pi_3)\}. \qquad (1.12)$$

It follows from Lemma 1 that

$$L_j(\pi', t') < L_s(\pi', t'),\ \forall j \in \{\pi_1\} \cup \{\pi_2\}. \qquad (1.13)$$

It is obvious that, for job $f$, we have

$$L_f(\pi', t') \leq L_f(\pi, t'). \qquad (1.14)$$

From (1.11)–(1.14), we arrive at $C_{\max}(\pi', t') \leq C_{\max}(\pi, t')$ and $L_{\max}(\pi', t') \leq L_{\max}(\pi, t')$. Let $\pi = (\pi_1, s, \pi_2, f, \pi_3)$, i.e., the job $s$ is processed before the job $f$. We construct a schedule $\pi' = (s, \pi_1, \pi_2, f, \pi_3)$. Then, the proof can be repeated as for $f$. The first part of the theorem is proven.

Suppose that $d_f \leq d_s$ and the schedule $\pi = (\pi_1, s, \pi_2, f, \pi_3)$. Let us construct a schedule $\pi' = (f, \pi_{11}, \pi_{12}, \pi_3)$ where $\pi_{11}$, and $\pi_{12}$ are schedules from the jobs of the sets $\{j : j \in \{(\pi_1, s, \pi_2)\}, d_j < d_f\}$ and $\{j : j \in \{(\pi_1, s, \pi_2)\}, d_j \geq d_f\}$. The jobs in $\pi_{11}$ and $\pi_{12}$ are arranged in nondecreasing order of release times $r_j$. It follows from $d_s \geq d_f$ that $s \in \{\pi_{12}\}$.

For each job $j \in \{\pi_{11}\}$, we have $d_j < d_f$. From (1.3), we obtain $d_j - r_j - p_j \geq d_f - r_f - p_f$. This implies that $r_j + p_j < r_f + p_f$, $\forall j \in \{\pi_{11}\}$, and $C_{\max}((f, \pi_{11}), t') = r_f(t') + p_f + \sum_{j \in \pi_{11}} p_j$. Since the jobs of the schedule $\{\pi_{12}\}$ are arranged in nondecreasing order of release times, we have $C_{\max}((f, \pi_{11}, \pi_{12}), t') \leq C_{\max}((\pi_1, s, \pi_2, f), t')$. As a result, we have

$$C_{\max}(\pi', t') \leq C_{\max}(\pi, t'), \qquad (1.15)$$

$$L_j(\pi', t') \leq L_j(\pi, t'),\ \forall j \in \{\pi_3\}. \qquad (1.16)$$

The job $j \in \{\pi_{12}\}$ satisfies the inequalities $d_j \geq d_f$ and $c_j(\pi', t') \leq c_f(\pi, t')$. Hence, we have

$$L_j(\pi', t') \leq L_f(\pi, t'),\ \forall j \in \{\pi_{12}\}. \qquad (1.17)$$

Since $s \in \{\pi_{12}\}$, we have

$$L_s(\pi', t') \leq L_f(\pi, t'). \qquad (1.18)$$

It follows from Lemma 1 that

$$L_j(\pi', t') \leq L_s(\pi, t'),\ \forall j \in \{\pi_{11}\}. \qquad (1.19)$$

Moreover, it is obvious that

$$L_f(\pi', t') \leq L_f(\pi, t'). \qquad (1.20)$$

It follows from (1.15)–(1.20) that $C_{\max}(\pi', t') \leq C_{\max}(\pi, t')$ and $L_{\max}(\pi', t') \leq L_{\max}(\pi, t')$, which we wanted to prove.

We call a schedule$\pi' \in \Pi(N)$ *efficient* it there is no schedule $\pi \in \Pi(N)$ such that $L_{\max}(\pi) \leq L_{\max}(\pi')$ and $C_{\max}(\pi) \leq C_{\max}(\pi')$ and at least one inequality is strict. Thus, when, for the jobs of the set $N$, constraints (1.3) hold, there is an efficient schedule $\pi'$ for which either the job $f = f(N, t)$ or the job $s = s(N, t)$ is first processed. Moreover, if $d_f \leq d_s$, then there exists an optimal schedule $\pi'$ with top-priority processing of the job $f$.

Let us determine $\Omega(N, t)$ as a subset of $\Pi(N)$. The schedule $\pi = (i_1, i_2, \ldots, i_n)$ belongs to $\Omega(N, t)$ if the job $i_k$, $k = 1, 2, \ldots, n$ is chosen from $f_k = f(N_{k-1}, c_{i_{k-1}})$ and $s_k = s(N_{k-1}, c_{i_{k-1}})$, where $N_{k-1} = N\backslash\{i_1, i_2, \ldots, i_{k-1}\}$, $c_{i_{k-1}} = c_{i_{k-1}}(\pi)$ and $N_0 = N$, $c_{i_0} = t$. For $d_{f_k} \leq d_{s_k}$, we have $i_k = f_k$; if $d_{f_k} > d_{s_k}$, then either $i_k = f_k$ or $i_k = s_k$. It is obvious that the set of schedules $\Omega(N, t)$ contains no more than $2^n$ schedules. For the example when

$$\begin{cases} n = 2m,\ t \leq r_1 < r_2 < \ldots < r_n, \\ r_{2i-1} < r_{2i} + p_{2i} < r_{2i-1} + p_{2i-1},\ 1 \leq i \leq m, \\ r_{2i} + p_{2i} + p_{2i-1} \leq r_{2i+1},\ 1 \leq i \leq m-1, \\ d_j = r_j + p_j,\ 1 \leq j \leq n, \end{cases} \qquad (1.21)$$

the set $\Omega(N, t)$ contains $2^m$ schedules.

**Theorem 2.** If, for the jobs of the subset $N' \subseteq N$, $|N'| = n'$, (1.3) is valid, then, for any time $t' \geq t$ and schedule $\pi \in \Pi(N')$, there exists a schedule $\pi' \in \Omega(N', t')$ such that $L_{\max}(\pi', t') \leq L_{\max}(\pi, t')$ and $C_{\max}(\pi', t') \leq C_{\max}(\pi, t')$

**Proof.** Let $\pi = (j_1, j_2, \dots, j_{n'})$ be an arbitrary schedule. We denote the first $l$ jobs of the schedule $\pi$ by $\pi_l$, $l = 0, 1, 2, \dots, n'$, where $\pi_0$ is an empty schedule and $\bar{\pi}_l = (j_{l+1}, \dots, j_{n'})$. Then, $\pi = (\pi_l, \bar{\pi}_l)$. Let us introduce $N_l = N' \backslash \{\pi_l\}$ and $C_l = C_{\max}(\pi_l, t')$. Suppose that, for a certain $l$, $0 \le l < n'$, $\pi_l$ is the greatest initial partial schedule of a certain schedule from $\Omega(N', t')$. If $j_1 \ne f(N', t')$ and $j_1 \ne s(N', t')$, then $\pi_l = \pi_0$, $l = 0$; i.e., the greatest partial schedule is empty. Assume that $f = f(N_l, C_l)$ and $s = s(N_l, C_l)$. If $d_f > d_s$, then $j_{l+1} \ne f$ and $j_{l+1} \ne s$; on the contrary, if $d_f \le d_s$, then $j_{l+1} \ne f$, since $\pi_{l+1}$ is not the initial schedule of a certain schedule from $\Omega(N', t')$.

By Theorem 1 for jobs of the set $\{\bar{\pi}_l\}$, $\bar{\pi}_l \in \Pi(N_l)$ from the time $C_l$, there exists a schedule $\bar{\pi}_l'$ for which we have $L_{\max}(\bar{\pi}_l', C_l) \le L_{\max}(\bar{\pi}_l, C_l)$ and $C_{\max}(\bar{\pi}_l', C_l) \le C_{\max}(\bar{\pi}_l, C_l)$, and $[\bar{\pi}_l]_1 = f$ or $s$. Moreover, for $d_f \le d_s$, we have $[\bar{\pi}_l]_1 = f$, where $[\sigma]_k$ is the job at the $k$th place in the schedule $\sigma$. This implies $L_{\max}((\pi_l, \bar{\pi}_l'), t') \le L_{\max}((\pi_l, \bar{\pi}_l), t')$ and $C_{\max}((\pi_l, \bar{\pi}_l'), t') \le C_{\max}((\pi_l, \bar{\pi}_l), t')$.

Let us denote $\pi' = (\pi_l, \bar{\pi}_l')$. We obtain a schedule $\pi'$ such that the processing of the first $l$ jobs coincides with the processing of the first $l$ jobs of a certain schedule from the set $\Omega(N', t')$ and $L_{\max}(\pi', t') \le L_{\max}(\pi, t')$ and $C_{\max}(\pi', t') \le C_{\max}(\pi, t')$.

After less than $n'$ sequential transformations (since the length of the schedule $n' \le n$) of the initial arbitrary chosen schedule $\pi$, we arrive at the schedule $\pi' \in \Omega(N', t')$ that provides that $L_{\max}(\pi', t') \le L_{\max}(\pi, t')$ and $C_{\max}(\pi', t') \le C_{\max}(\pi, t')$, which we wanted to prove.

Let us form the following partial schedule $\omega(N, \mathrm{t}) = (i_1, i_2, \dots, i_l)$. For each job $i_k$, $k = 1, 2, \dots, l$, we have $i_k = f_k$ and $d_{f_k} \le d_{s_k}$, where $f_k = f(N_{k-1}, C_{k-1})$ and $s_k = s(N_{k-1}, C_{k-1})$. For $f = f(N_l, C_l)$ and $s = s(N_l, C_l)$, the inequality $d_f > d_s$ holds. If $d_f > d_s$ for $f = f(N, t)$ and $s = s(N, t)$, then $\omega(N, t) = \emptyset$. Thus, $\omega(N, t)$ is the "maximum" schedule such that a job ($f$) is chosen uniquely at the recurrent place of the schedule in its construction. With the help of algorithm 1, for jobs of the set $N$ from the time $t$, the schedule $\omega(N, t)$ can be constructed.

**Algorithm 1.** Let us set $\omega = \emptyset$ in advance.

We find the jobs $f = f(N, t)$ and $s = s(N, t)$. If $d_f \le d_s$, then $\omega = (\omega, f)$; otherwise, the algorithm terminates the operation. We set $N = N \backslash \{f\}$ and $t = r_f(t) + p_f$ and repeat operations for the next step.

**Lemma 2.** The run time of algorithm 1 for finding the schedule $\omega(N, t)$ for any $n$ and $t$ is no more than $O(n \log n)$ operations.

**Proof.** At each iteration step of algorithm 1, we find two jobs, $f = f(N, t)$ and $s = s(N, t)$. If the jobs are arranged according to the release times $r_j$ (and, correspondingly, the time $r(t)$ is found for $O(1)$ operations), then, to find the two jobs ($f$ and $s$), $O(\log n)$ operations are required. In total, there are no more than $n$ iteration steps. Thus, to construct the schedule $\omega(N, t)$, $O(n \log n)$ operations are required.

**Lemma 3.** If, for jobs of the set $N$, (1.3) holds, then any schedule $\pi \in \Omega(N, t)$ begins with the schedule $\omega(N, t)$.

**Proof.** If $\omega(N, t) = \emptyset$, i.e., $d_f > d_s$, where $f = f(N, t)$ and $s = s(N, t)$, the assertion of the lemma holds, since any schedule stars from the empty one.

Let $\omega(N, t) = (i_1, i_2, \dots, i_l)$ and, consequently, for each $i_k$, $k = 1, 2, \dots, l$, we have $i_k = f_k$ and $d_{f_k} \le d_{s_k}$, where $f_k = f(N_{k-1}, C_{k-1})$ and $s_k = s(N_{k-1}, C_{k-1})$. For $f = f(N_l, C_l)$ and $s = s(N_l, C_l$ we have $d_f > d_s$. As can be seen from the definition of the set of schedules $\Omega(N, t)$, all schedules of this subset begin with the partial schedule $\omega(N, t)$.

Let us use the following notation: $\omega^1(N, t) = (f, \omega(N', t'))$ and $\omega^2(N, t) = (s, \omega(N'', t''))$, where $f = f(N, t)$, $s = s(N, t)$, $N' = N \backslash \{f\}$, $N'' = N \backslash \{s\}$, $t' = r_f(t) + p_f$, and $t'' = r_s(t) + p_s$. It is obvious that the algorithm for finding $\omega^1$ (as well as $\omega^2$) is executed for $O(n \log n)$ operations, the same as the algorithm for constructing $\omega(N, t)$.

**Corollary from Lemma 3.** If for jobs of the set $N$ (1.3) is valid, then each schedule $\pi \in \Omega(N, t)$ begins with either $\omega^1(N, t)$ or $\omega^2(N, t)$.

**Theorem 3.** If the jobs of the set $N$ satisfy (1.3), then, for any schedule $\pi \in \Omega(N, t)$, $(i \longrightarrow j)_\pi$ holds for any $i \in \{\omega^1(N, t)\}$ and $j \in N \backslash \{\omega^1(N, t)\}$.

**Proof.** In the case $\{\omega^1(N, t)\} = N$, the assertion of the theorem is obviously true. Let $\{\omega^1(N, t)\} \ne N$. Below in the text of proof, we use the notation $\omega^1 = \omega^1(N, t)$.

If $f = f(N, t)$ and $s = s(N, t)$ are such that $d_f \le d_s$, then all schedules from the set $\Omega(N, \mathrm{t})$ begin with the partial schedule $\omega(N, t) = \omega^1$; therefore, the assertion of the theorem is also true.

Consider the case $d_f > d_s$. All schedules of the set $\Omega(N, t)$ that begins with the job $f$ have the partial schedule $\omega(N, t) = \omega^1$. Let us take an arbitrary schedule $\pi \in \Omega(N, t)$ with the job $s$at the first place, $[\pi]_1 = s$ and the schedule $|\omega^1| = l$, $l < n$ that contains $l$ jobs. Assume that $\pi_l = (j_1, j_2, \dots, j_l)$ is a partial schedule of length $l$ of the schedule $\pi$, $j_1 = s$. Let us prove that $\{\pi_l\} = \{\omega^1\}$. Suppose the contrary, which means that there exists a job $j \in \{\pi_l\}$, but $j \notin \{\omega^1\}$.

Assume that $(j \longrightarrow f)_\pi$. If $d_j < d_f$, then (1.3) implies that $d_j - r_j - p_j \ge d_f - r_f - p_f$; therefore, $r_j + p_j < r_f + p_f$. Hence, the job $j$ is involved in the schedule $\omega^1$ by the definition of $\omega(N, t)$ and $\omega^1$, but, by assumption, $j \notin \{\omega^1\}$. If $d_j \ge d_f$, then the fact that $\pi \in \Omega(N, t)$ implies that $(f \longrightarrow j)_\pi$, but this contradicts $(j \longrightarrow f)_\pi$.

Let $(f \longrightarrow j)_\pi$. Then, for each job $i \in \{\omega^1\}$ such that $i \notin \{\pi_l\}$, we have $r_i < r_i + p_i \le C_{\max}(\omega^1) < r_{s_{l+1}} \le r_j$ since $j \notin \{\omega^1\}$, where $s_{l+1} = s(\mathrm{N} \backslash \{\omega^1\}, C_{\max}(\omega^1))$. The jobs

$s_{l+1}$ and $j$ were not sequenced in the schedule $\omega^1$; therefore, $C_{\max}(\pi_1) < r_{s_{l+1}} \le r_j$. Moreover, $d_i \le d_j$. If we have $d_i > d_j$, then $r_i + p_i \ge r_j + p_j$, but $r_i + p_i < r_j$ is valid. Hence, $i \longrightarrow j)_{\pi_l}$ since $\pi = (\pi_l, \bar{\pi}_l) \in \Omega(N, t)$ but the assumption that $i \notin \{\pi_l\}$ and $j \in \{\pi_l\}$ is violated.

Therefore, our supposition is not true; therefore, $\{\omega^1\} = \{\pi_l\}$, which we wanted to prove.

Thus, the processing of jobs of the set $\{\omega^1(N, t)\}$ precedes the processing of the jobs of the set $N \backslash \{\omega^1(N, t)\}$ for any schedule from the set $\Omega(N, t)$.

## 2. MAKESPAN PROBLEM UNDER A CONSTRAINT ON THE MAXIMUM LATENESS

Let us formulate problem 1 $1|d_i \le d_j, d_i - r_i - p_i \ge d_j - r_j - p_j; L_{\max} \le y|C_{\max}$, which consists in finding, for a certain real number $y$, a schedule $\theta$ with $C_{\max}(\theta) = \min\{C_{\max}(\pi)\colon L_{\max}(\pi) \le y\}$. If $L_{\max}(\pi) > y$ for any $\pi \in \Pi(N)$, then $\theta = \emptyset$.

**Algorithm 2.** Let us set $\theta = \omega(N, t)$. If $L_{\max}(\pi) > y$, then $\theta = \emptyset$ and the algorithm terminates the operation; otherwise, continue to construct the schedule $\theta$.

Find $N' = N \backslash \{\theta\}$ and $t' := C_{\max}(\theta)$. If $N' = \emptyset$, then the algorithms is terminated. Otherwise,

if $L_{\max}(\omega^1, t') \le y$, where $\omega^1 = \omega^1(N', t')$, then $\theta = (\theta, \omega^1)$ and go to the next step;

if $L_{\max}(\omega^1, t') > y$ and $L_{\max}(\omega^2, t') \le y$ where $\omega^2 = \omega^2(N', t')$, then $\theta = (\theta, \omega^2)$ and go to the next step;

if $L_{\max}(\omega^1, t') > y$ and $L_{\max}(\omega^2, t') > y$, the set $\theta = \emptyset$ and the algorithm terminates the operation.

**Lemma 4.** The run time of algorithm 2 does not exceed $O(n^2 \log n)$ operations.

**Proof.** At each iteration step of the main step of algorithm 2, the schedules $\omega^1$ are found, and, if necessary, the schedules $\omega^2$ are also found for $O(n \log n)$ operations. Since $\omega^1$ and $\omega^2$ consists of at least one job, at each iteration step of the algorithm, one or several jobs are added to the schedule $\theta$ or we set $\theta = \emptyset$ and terminate the work. Therefore, the total number of algorithm steps is no less than $n$. Thus, algorithm 2 is executed in $O(n^2 \log n)$ operations.

Denote by $\theta(N, t, y)$ the schedule constructed by algorithm 2 since the time $t$ from jobs of the set $N$ with a maximum lateness no greater than $y$. If $N = \emptyset$, then $\theta(\emptyset, t, y) = \emptyset$ for any $t$ and $y$.

**Theorem 4.** Assume that, for jobs of the set $N$, (1.3) holds. If with the help of algorithm 2 the schedule $\theta(N, t, y) = \emptyset$ is constructed, then $C_{\max}(\theta) = \min\{C_{\max}(\pi)\colon L_{\max}(\pi) \le y, \pi \in \Pi(N)\}$. If, as a result of operation of algorithm 2, the schedule is not formed, i.e., $\theta(N, t, y) = \emptyset$, then $L_{\max}(\pi) > y$ for each $\pi \in \Pi(N)$.

**Proof.** In the situation when for the schedule $\pi \in \Pi(N)$ holds, there exists a schedule $\pi' \in \Omega(N, t)$ such that $L_{\max}(\pi') \le L_{\max}(\pi) \le y$ and $C_{\max}(\pi') \le C_{\max}(\pi)$ by Theorem 2. Therefore, the desired schedule $\theta$ is found among the schedules of the set $\Omega(N, t)$. By Lemma 3, all schedules of the set $\Omega(N, t)$ begin with $\omega(N, t)$. Let us take $\theta_0 = \omega(N, t)$.

After $k$, $k \ge 0$ main steps of algorithm 2, we obtain the schedule $\theta_k$ and $N' = N \backslash \{\theta_k\}$ and $t' = C_{\max}(\theta_k)$. Suppose that there exists a makespan schedule $\theta$ beginning with $\theta_k$. By Theorem 2, the schedule $\theta_k$ can be optimally extended among the schedules of the set $\Omega(N', t')$.

Let $\theta_{k+1} = (\theta_k, \omega^1(N', t'))$, i.e., $L_{\max}(\theta_{k+1}) \le y$. Under the schedule $\omega^1$, $\omega^1 = \omega^1(N', t')$, there are no artificial idle times of the machine, and all schedules from the set $\Omega(N', t')$ begin with jobs of the set $\{\omega^1(N', t')\}$ by Theorem 3. Therefore, $\omega^1(N', t')$ is makespan ($C_{\max}$) among all extensions of the partial schedule $\theta_k$ that are admissible in the maximum lateness ($L_{\max}$).

At the recurrent step of the algorithm, $\theta_{k+1} = (\theta_k, \omega^2(N', t'))$, i.e., $L_{\max}(\omega^1, t') > y$ and $L_{\max}(\omega^2, t') \le y$. All the schedules of the set $\Omega(N', t')$ begin with the schedule $\omega^1(N', t')$ or $\omega^2(N', t')$. Since $L_{\max}(\omega^1, t') > y$, there exists only one appropriate extension $\omega^2(N', t')$.

Thus, at each main step of the algorithm, we choose the makespan extension of the partial schedule $\theta_k$ among all extensions admissible in the maximum lateness. After no more than $n$ main steps of the algorithm, the desired schedule will be constructed.

Assume that, after the *(k + 1)*th step of the algorithm, $L_{\max}(\omega^1, t') > y$ and $L_{\max}(\omega^2, t') > y$. If the schedule $\theta$ existed, i.e., $\theta \neq \emptyset$, then $\theta$ would begin with $\theta_k$. Then, for any schedule $\pi \in \Pi(N', t')$, there exists a schedule $\pi' \in \Omega(N', t')$ such that either $L_{\max}(\pi, t') \ge L_{\max}(\pi', t') \ge L_{\max}(\omega^1, t') > y$ or $L_{\max}(\pi, t') \ge L_{\max}(\pi', t') \ge L_{\max}(\omega^2, t') > y$. Therefore, $\theta = \emptyset$.

Repeating our considerations as many times as the main step of algorithm 2 was executed (no more than $n$ times), we arrive at the validity of the assertion of the theorem.

## 3. AN ALGORITHM FOR CONSTRUCTING THE SET OF PARETO-SCHEDULES ACCORDING TO THE CRITERIA $C_{\max}$ AND $L_{\max}$

In what follows, we present an algorithm for constructing the set of Pareto-schedules $\Phi(N, t) = \{\pi'_1, \pi'_2, \ldots, \pi'_m\}$ according to the criteria $C_{\max}$ and $L_{\max}$ such that

$$C_{\max}(\pi'_1) < C_{\max}(\pi'_2) < \ldots < C_{\max}(\pi'_m),)$$

$$L_{\max}(\pi'_1) < L_{\max}(\pi'_2) > \ldots > L_{\max}(\pi'_m).$$

The schedule $\pi'_m$ is a solution to problem $1|r_j|\,L_{\max}$ under the condition that (1.3) holds.

**Algorithm 3.** Set $y := +\infty$, $\pi^* := \omega(N, t)$, $\Phi := \emptyset$, $m := 0$.

Find $N' = N\backslash\{\pi^*\}$ and $t' = C_{max}(\pi^*)$. If $N' = \emptyset$, then $\Phi = \Phi \cup (\pi^*)$ and $m = 1$ and the algorithm terminates operation. Otherwise,

if $L_{\max}(\omega^1, t') \le L_{\max}(\pi^*)$, then $\pi^* := (\pi^*, \omega^1)$, where $\omega^1 = \omega^1(N', t')$ and go to the next step;

in the case $L_{\max}(\omega^1, t') > L_{\max}(\pi^*)$ the following variants are possible:

(1) $L_{\max}(\omega^1, t') < y$, then find $\theta = \theta(N', t', y')$ with the help of algorithm 2, where $y' = L_{\max}(\omega^1, t')$ and test

if $\theta = \emptyset$, then $\pi^* := (\pi^*, \omega^1)$ and go to the next step;

otherwise, set $\pi' := (\pi^*, \theta)$; comparing $C_{\max}(\pi'_m) < C_{max}(\pi')$ and making sure that this inequality is valid, we execute $m := m + 1$, $\pi'_m := \pi'$, $\Phi := \Phi \cup (\pi'_m)$ and $y = L_{\max}(\pi'_m)$; in the opposite case, $\pi'_m := \pi'$ and go to the next step;

(2) $L_{max}(\pi_1, t') \ge y$; find $\omega^2 = \omega^2(N', t')$:

if $L_{max}(\omega^2, t') < y$, then $\pi^* := (\pi^*, \omega^2)$ and go to the next step;

otherwise, $\pi^* := \pi'_m$ and the algorithm terminates operation.

As a result of operation of algorithm 3 for the set of jobs $N$, since the moment $t$, a set of schedules $\Phi(N, t)$ is constructed such that $1 \le |\Phi(N, t)| \le n$. The set $\Phi(N, t)$ of example (1.21) consists of no more than two schedules.

**Lemma 5.** The run time of algorithm 3 does not exceed $O(n^3 \log n)$ operations.

**Proof.** At each iteration of the described step of algorithm 3, the schedule $\omega^1$ and, if necessary, the schedule $\omega^2$ are found for $O(n \log n)$ operations according to Lemma 2, as well as the schedule $\theta$ for $O(n^2 \log n)$ operations. Since $\omega^1$ and $\omega^2$ consist of at least one job, at any iteration step of the algorithm, one or more jobs are added to the schedule $\pi^*$ or the algorithm is terminated at the last reference schedule $\pi'$. Therefore, the total number of iterations is no greater than $n$. Thus, algorithm 3 is executed for no more than $O(n^3 \log n)$ operations.

**Theorem 5.** Suppose that, for jobs of $N$, (1.3) is valid. Then the schedule $\pi^*$ constructed by algorithm 3 is optimal according to the criterion $L_{\max}$. Moreover, for any schedule $\pi \in \Pi(N)$, there is a schedule $\pi' \in \Phi(N, t)$ such that $L_{\max}(\pi') \le L_{\max}(\pi)$ and $C_{\max}(\pi') \le C_{\max}(\pi)$.

**Proof.** By Theorem 2, there exists an optimal (according to $L_{\max}$) schedule that belongs to $\Omega(N, t)$. All schedules of the set $\Omega(N, t)$ begin with the partial schedule $\omega(N, t)$ by Lemma 3.

Let $\pi_0 = \omega(N, t)$. After $k$, $k \ge 0$ main steps of algorithm 3, we have a partial schedule $\pi_k$. Assume that there is an optimal (according to $L_{\max}$) schedule beginning with $\pi_k$. Let us introduce the notation $N' = N\backslash\{\pi_k\}$ and $t' = C_{max}(\pi_k)$.

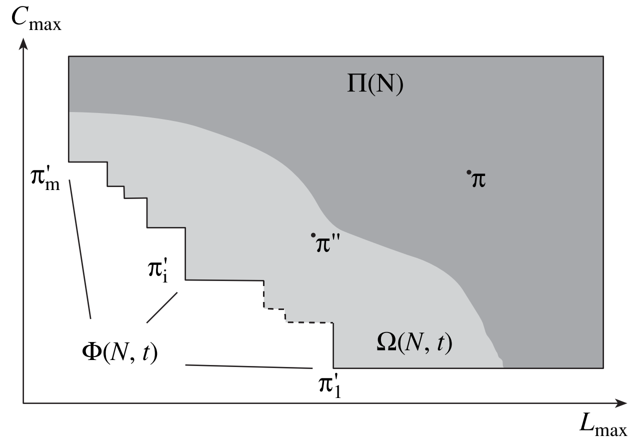
If $\pi_{k+1} = (\pi_k, \omega^1)$, where $\omega^1 = \omega^1(N', t')$, then either $L_{max}(\omega^1, t') \le L_{max}(\pi_k)$ or $L_{max}(\pi_k) < L_{max}(\omega^1, t') < y$ is the current value of the criterion, and maximum lateness will appear at the next steps of algorithm 3, i.e., $\theta(N', t', y') \ne \emptyset$, where $y' = L_{max}(\omega^1, t')$. If $\theta = \theta(N', t', y') \ne \emptyset$, then we improve the current value of the maximum lateness $\pi' = (\pi_k, \theta)$ and $y = L_{max}(\pi') = L_{max}(\omega^1, t')$. The schedule $\pi'$ is appended to the set of schedules $\Phi(N, t)$. Moreover, the processing of jobs of the set $\{\omega^1\}$ precedes the processing of jobs of the set $N' \backslash\{\omega^1\}$,– by Theorem 3. Thus, the schedule $\omega^1$ without artificial idle times of the machine is the best extension for $\pi_k$.

Consider the case $\pi_{k+1} = (\pi_k, \omega^2)$, where $\omega^2 = \omega^2(N', t')$, i.e., $L_{max}(\omega^2, t') < L_{\max}(\pi') \le L_{\max}(\omega^1, t')$ according to algorithm 3. Therefore, the extension $\omega^2$ is better than $\omega^1$. This implies that the partial schedule $\pi_{k+1}$ is part of a certain optimal schedule.

Repeating such considerations no more than $n$ times, we arrive at the optimality (according to $L_{\max}$) of the schedule $\pi^*$. The set of schedules $\Phi(N, t)$ contains no more than $n$ schedules since, at each main step of the algorithm, no more than one schedule is appended to the set $\Phi(N, t)$, and this step is executed no more than $n$ times.

Assume that there exists a schedule $\pi \in \Pi(N)$, $\pi \notin \Phi(N, t)$, such that either $C_{\max}(\pi) \le C_{\max}(\pi')$ and $L_{\max}(\pi) \ge L_{\max}(\pi')$ or $C_{\max}(\pi) \ge C_{\max}(\pi')$ and $L_{\max}(\pi) \le L_{\max}(\pi')$ for each schedule $\pi' \in \Phi(N$, and at least one of the pair of inequalities is strict. Theorem 2 implies that there exists a schedule $\pi'' \in \Omega(N, t)$ such that $L_{\max}(\pi'') \le L_{\max}(\pi)$ and $C_{\max}(\pi'') \le C_{\max}(\pi)$. If $\pi'' \in \Phi(N, t)$, then, obviously, our assumption does not hold. Let $\pi'' \in \Omega(N, t)\backslash\Phi(N, t)$. Algorithm 3 shows that the structure of each schedule $\pi' \in \Phi(N, t)$ can be represented as a sequence of partial schedules $\pi' = (\omega'_0, \omega'_1, \omega'_2, \ldots, \omega'_{k'})$, where $\omega'_0 = \omega(N, t)$, $\omega'_i$ is either $\omega^1(N'_i, C'_i)$ or $\omega^2(N'_i, C'_i)$ and $N'_i = N\backslash\{\omega'_0, \ldots, \omega'_{i-1}\}$, $C'_i = C_{\max}((\omega'_0, \ldots, \omega'_{i-1}), t)$, $i = 1, 2, \ldots, k'$. The schedule $\pi''$ has an analogous structure by the definition of the set $\Omega(N, t)$; i.e., $\pi = (\omega''_0, \omega''_1, \omega''_2, \ldots, \omega'_{k''})$, possibly, $k'' \ne k'$, where $\omega''_0 = \omega'_0 = \omega(N, t)$, $\omega''_i$ is either $\omega^1(N''_i, C''_i)$ or $\omega^2(N''_i, C''_i)$, and $N''_i = N\backslash\{\omega''_0, \ldots, \omega''_{i-1}\}$ and $C''_i = C_{\max}((\omega''_0, \ldots, \omega''_{i-1}), t)$, $i = 1, 2, \ldots, k''$.

Suppose that the first $k$ partial schedules $\pi''$ and $\pi'$ coincide, i.e., $\omega''_i = \omega'_i = \omega_i$, $i = 0, 1, \ldots, k-1$ and $\omega''_k \ne \omega'_k$. Let $y = L_{\max}(\omega_0, \ldots, \omega_{k-1})$. We construct the schedule $\theta$ with the help of algorithm 2, $\theta = \theta(N_k, C_k, y)$. If $\theta = \emptyset$, then, by algorithm 3, $\omega'_k = \omega^1(N_k, C_k)$. Since $\omega''_k \ne \omega'_k$, we have $\omega''_k = \omega^2(N_k, C_k)$. The value of the goal function ($L_{\max}$) is achieved on jobs of the set $N_k$ since $\theta = \emptyset$. The entire structure of algorithm 3 is




The set of Pareto-optimal schedules.

arranged so as to order the jobs as densely as possible before the critical job is met (according to $L_{\max}$); therefore, we extend the schedule $\omega^1$ and after that $C_{\max}(\pi') \le C_{\max}(\pi'')$ and $L_{\max}(\pi') \le L_{\max}(\pi'')$. If $\theta \neq \emptyset$, then, for the schedules $\pi'$ and $\pi''$, we have $C_{\max}(\pi') \le C_{\max}(\pi'')$ and $L_{\max}(\pi') = L_{\max}(\pi'')$. Thus, for any schedule $\pi'' \in \Omega(N, t)\backslash\Phi(N, t)$, there exists a schedule $\pi' \in \Phi(N, t)$ such that $L_{\max}(\pi') \le L_{\max}(\pi'')$ and $C_{\max}(\pi') \le C_{\max}(\pi'')$ . Figure 1 schematically presents the considered schedules.

For the set of schedules $\Phi(N, t) = \{ \pi'_1, \pi'_2, \ldots, \pi'_m \}$, we have

$$C_{\max}(\pi'_1) < C_{\max}(\pi'_2) < \ldots < C_{\max}(\pi'_m),$$

$$L_{\max}(\pi'_1) > L_{\max}(\pi'_2) > \ldots > L_{\max}(\pi'_m).$$

The schedule $\pi'_1$ is makespan (according to $C_{\max}$), while $\pi'_m$ is optimal according to the maximum lateness (according to $L_{\max}$) if the parameters of jobs of the set $N$ satisfy conditions (1.3)

Experimental investigation of algorithm 3 has shown that it is able to construct optimal schedules (according to $L_{\max}$) even for examples that do not satisfy conditions (1.3).

## REFERENCES


1. R. L. Graham, E. L. Lawler, J. K. Lenstra, et al., "Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey," Ann. Discrete Math. **5**, 287–326 (1979).
2. J. K. Lenstra, A. H. G. Rinnooy Kan, and P. Brucker, "Complexity of Machine Scheduling Problems," Ann. Oper. Res. **1**, 343–362 (1975).
3. J. R. Jackson, "Scheduling a Production Line to Minimize Maximum Tardiness," Manage. Sci. **43** (1955).
4. C. N. Potts, "Analysis of a Heuristic for One Machine Sequencing with Release Dates and Delivery Times," Oper. Res. **28**, 1436–1441 (1980).
5. L. A. Hall and D. B. Shmoys, "Jackson'S Rule for One-Machine Scheduling: Making a Good Heuristic Better," Math. Oper. Res. **17**, 22–35 (1992).
6. M. Mastrolilli, "Efficient Approximation Schemes for Scheduling Problems with Release Dates and Delivery Times," J. Scheduling **6**, 521–531 (2003).
7. E. L. Lawler, "Optimal Sequencing of a Single Machine Subject to Precedence Constraints," Manage. Sci. **19**, 544–546 (1973).
8. B. B. Simons, "A Fast Algorithm for Single Processor Scheduling," in *Proceedings of 19th Annual Symposium on Foundations of Computer Science, New York, USA, 1978*, pp. 246–252.
9. K. R. Baker, E. L. Lawler, J. K. Lenstra, et al., "Preemptive Scheduling of a Single Machine to Minimize Maximum Cost Subject to Release Dates and Precedence Constraints," Oper. Res. **31**, 381–386 (1983).
10. J. A. Hoogeveen, "Minimizing Maximum Promptness and Maximum Lateness on a Single Machine," Math. Oper. Res. **21**, 100–114 (1996).
11. A. A. Lazarev and O. N. Shul'gina, "A Pseudo-polynomial Algorithm for Solving the NP-hard Problem of Minimization of Maximum Lateness," in *Proceedings of 11th Baikal Workshop on Optimization Methods and Their Applications, Irkutsk, Russia, 1998*, pp. 163–167 [in Russian].
12. A. A. Lazarev and O. N. Shul'gina, "A Polynomially Resolved Partial Case of the Problem of Minimization of Maximum Lateness," Izv. Vyssh. Uchebn. Zaved., Mat., 11 (2000).
13. A. A. Lazarev, Candidate's Dissertation in Mathematics and Physics (KGU, Kazan, 1989).