ISSN 1990-4789, Journal of Applied and Industrial Mathematics, 2007, Vol. 1, No. 4, pp. 468–480. © Pleiades Publishing, Ltd., 2007.
Original Russian Text © A.A. Lazarev, R.R. Sadykov, S.V. Sevastyanov, 2006, published in Diskretnyi Analiz i Issledovanie Operatsii, Ser. 2, 2006, Vol. 13, No. 1, pp. 57–76.

A Scheme of Approximation Solution of Problem $1 \mid r_j \mid L_{\max}$

A. A. Lazarev[1*], R. R. Sadykov[1], and S. V. Sevastyanov[2]**

[1] *Kazan State University, Chair of Economical Cybernetics, ul. Kremlevskaya 18, Kazan, 420008 Russia*

[2] *Sobolev Institute of Mathematics, pr. Akad. Koptyuga 4, Novosibirsk, 630090 Russia*

Received April 10, 2005; in final form, November 13, 2005

Abstract—The strongly NP-hard scheduling problem of minimizing the maximum lateness on one machine subject to job release dates is under study. We present a general scheme of approximation solution of the problem which is based on searching for a given problem instance another instance, closest to the original in some metric and belonging to a known polynomially solvable class of instances. For a few concrete variants of the scheme (using different polynomially solvable classes of instances) some analytic formulas are found that make it possible, given a problem instance, to compute easily an upper bound on the absolute error of the solution obtained by a chosen scheme.

DOI: 10.1134/S1990478907040102

INTRODUCTION

A family of jobs $N = \{1, 2, \ldots, n\}$ must be processed on one machine. Preemption and simultaneous processing of several jobs at a time are not allowed. For each $j \in N$ its release date r_j (the minimum possible starting time of the job), the processing time $p_j \geqslant 0$, and a due date d_j for its completion time are specified.

A schedule S is defined by a family $S = \{s_j \mid j \in N\}$ of the job starting times. A schedule S is called *feasible* if $s_j(S) \geqslant r_j$ for all $j \in N$ and the intervals $(s_{j'}(S), s_{j'}(S) + p_{j'})$ and $(s_{j''}(S), s_{j''}(S) + p_{j''})$ do not overlap for any two jobs $j' \neq j''$. The completion time of $j \in N$ in S is denoted by $c_j(S)$. Clearly, $c_j(S) = s_j(S) + p_j$. The *lateness* $c_j(S) - d_j$ of job $j \in N$ in S is denoted by $L_j(S)$, while $L_{\max}(S) = \max_{j \in N} L_j(S)$ stands for the maximum job lateness in S. The problem is to find a feasible schedule S^* providing the minimum value to the maximum job lateness.

This problem is usually denoted by $\langle 1 \mid r_j \mid L_{\max} \rangle$ [5]. Algorithms for its solution can be used for solving other scheduling problems, for instance, job shop problem [1] and the problems of minimizing the weighted number of late jobs on a single machine [14].

Intensive work on elaborating the methods of solution for this problem is performed since the early 50s of the last century. As shown in [13], Problem $\langle 1 \mid r_j \mid L_{\max} \rangle$ is strongly NP-hard. A series of polynomial time solvable cases of the problem was found, starting with the earliest result of [8] for the case $r_j = 0,\ j \in N$, where the sequence defined by the EDD-rule ("the job with the earliest due date is the first") turned out optimal. Problems $\langle 1 \mid \text{prec},\ r_j \mid L_{\max} \rangle$, $\langle 1 \mid \text{prec},\ p_j = p,\ r_j \mid L_{\max} \rangle$, and $\langle 1 \mid \text{prec},\ r_j,\ \text{pmtn} \mid L_{\max} \rangle$ with precedence constraints for job processing were considered in [2, 12, 18]. An algorithm with polynomial running time $O(n^2 \log n)$ was proposed in [7] for the special case with the property $\max_i(d_i - r_i - p_i) \leqslant \min_i(d_i - r_i)$. Another pseudo-polynomial time algorithm for the NP-hard case when the release dates and due dates are numbered in the opposite order ($d_1 \leqslant \cdots \leqslant d_n$ and $r_1 \geqslant \cdots \geqslant r_n$) was designed by Lazarev and Shulgina in [11].

The most frequently used exact method of the Branch-and-Bound type is the algorithm due to Carlier [4]. The method proved to be good enough for large scale instances. Other exact methods for this problem were considered in [3, 9, 15].

[*] E-mail: Alexandr.Lazarev@ksu.ru
[**] E-mail: seva@math.nsc.ru

Some of the published papers describe approximation algorithms with ratio performance guarantees. Potts [17] presented an iterative version of the extended Jackson's rule (IJ) and proved its performance ratio $L_{\max}(S_{IJ})/L^*_{\max} \leqslant 3/2$. Hall and Shmoys [6] proposed a modified iterative version (MIJ) with ratio performance guarantee $L_{\max}(S_{MIJ})/L^*_{\max} \leqslant 4/3$. They also presented two approximation schemes that for any $\varepsilon > 0$ guaranteed an ε-approximation in time $O(n \log n + n(1/\varepsilon)^{O(1/\varepsilon^2)})$ and $O((n/\varepsilon)^{O(1/\varepsilon)})$ respectively. Mastrolilli [14] designed an improved approximation scheme with running time $O(n + (1/\varepsilon)^{O(1/\varepsilon)})$. There are known also several polynomial time algorithms with absolute performance guarantee

$$L_{\max}(S) - L^*_{\max} \leqslant p_{\max} \doteq \max_{j \in N} p_j.$$

In this paper we propose an original approach to searching for approximation solutions with worst case absolute performance guarantees. The core idea of our approach consists in finding for a given instance A another instance C (with the same number of jobs) which, first, belongs to a known polynomially solvable class of instances and, second, is the nearest to instance A (in a certain metric $\rho(A, C)$) among all instances of the class. Having applied the polynomial time algorithm to instance C, we find its optimal sequence of jobs and apply it to the original instance A as an approximation solution with an absolute error at most the distance $\rho(A, C)$ between instances A and C.

Thus, the article presents an effective combination of the two classical approaches to solving NP-hard problems: 1) designing approximation algorithms and 2) searching for efficiently solvable special cases. Up to our knowledge, such a combination of these two approaches is proposed for the first time.

The scheme of the paper is as follows: Section 1 introduces the basic notation and definitions. In Section 2 a formula is derived which estimates the absolute variation of the optimum under a given modification of such problem parameters as release dates and due dates (provided that job processing times are fixed). Section 3 describes a general scheme of searching for approximation solution for a given problem instance. Subsections 3.1 and 3.2 address certain variants of the scheme based on two different polynomial time solvable cases of the problem. In both variants, an absolute error of the solution obtained is estimated in terms of the job parameters of a given instance. Finally, in the conclusion we formulate the main results of the article and propose further promising research directions.

1. NOTATION AND DEFINITIONS OF BASIC NOTIONS

In this section we introduce some basic notation and definitions to be used in the paper.

$L^A_j(S)$ and $c^A_j(S)$ will stand for the lateness and completion time of job $j \in N$ in S for a given instance A with problem parameters $\{r^A_j, p^A_j, d^A_j\}$, $j \in N$. Respectively, $L^A(S) = \max_{j \in N} L^A_j(S)$ will denote the maximum job lateness in a schedule S for a given instance A.

Definition 1. Given a problem instance A, each permutation π of jobs in N uniquely defines the *early schedule* S^A_π. In an early schedule each job $j \in N$ starts processing at the earliest possible time: either at its release date r^A_j or right after the completion of the previous job in the corresponding sequence. Thus, for an arbitrary permutation $\pi = (j_1, \ldots, j_n)$, the corresponding early schedule $S^A_\pi = \{s_j \mid j \in N\}$ is defined as:

$$s_{j_1} = r^A_{j_1}, \quad s_{j_k} = \max\{s_{j_{k-1}} + p^A_{j_{k-1}}, r^A_{j_k}\}, \quad k = 2, \ldots, n.$$

In all our constructions the early schedules play an exclusive role, since the optimal schedule of any instance is contained in the set of early schedules.

π^A and S^A will denote the optimal sequence and optimal schedule of an instance A. For optimal schedules, we deal only with early schedules, assuming thereby that $S^A = S^A_{\pi^A}$.

$\Pi(N)$ will denote the set of all permutations of jobs in N.

Definition 2. Given an instance A with job set N, we say that an instance B on the same job set *inherits* a parameter x from A, if $x^B_j = x^A_j$ for $j \in N$.

Definition 3. An *instance* $Q = \{(r_j^Q, p_j^Q, d_j^Q) | j \in N\}$ is *inverse* to an instance

$$P = \{(r_j^P, p_j^P, d_j^P) \mid j \in N\},$$

if for all $j \in N$ we have

$$r_j^Q = -d_j^P, \qquad p_j^Q = p_j^P, \qquad d_j^Q = -r_j^P.$$

The permutation $\pi' = (i_n, i_{n-1}, \ldots, i_1)$ is called *inverse* to a permutation $\pi = (i_1, \ldots, i_n)$. *The schedule* $S' = \{s'_j \mid j \in N\}$ is called *inverse* to a schedule $S = \{s_j \mid j \in N\}$, if $s'_j = -s_j - p_j$ for all $j \in N$.

It is easily seen that the notion of inversion is symmetric, i.e., if an object X is inverse to Y then Y is inverse to X. In particular, the processing intervals I_j and I'_j of job j in two mutually inverse schedules S and S' are symmetric with respect to the origin.

Definition 4. Given an instance $V = \{(r_j^V, p_j^V, d_j^V) \mid j \in N\}$, its feasible schedule S is called *fully feasible*, if each job $j \in N$ is processed within its due interval $[r_j^V, d_j^V]$.

Furthermore, $\Delta = \Delta(V, S)$ will denote the minimum amount (possibly, negative) that should be added to all due dates of jobs of instance V so as given feasible schedule S became fully feasible. Clearly, $\Delta(V, S) = L^V(S)$.

Definition 5. Given instances A and B, we define the following functions:

$$\rho_d(A, B) = \max_{j \in N} \{d_j^A - d_j^B\} + \max_{j \in N} \{d_j^B - d_j^A\}$$

$$\rho_r(A, B) = \max_{j \in N} \{r_j^A - r_j^B\} + \max_{j \in N} \{r_j^B - r_j^A\}$$

$$\rho(A, B) = \rho_d(A, B) + \rho_r(A, B).$$

It can be easily checked that $\rho(A, B)$ meets all metric properties, and so it can be used for measuring a *distance* between instances A and B.

2. THE ABSOLUTE ERROR OF AN APPROXIMATE SOLUTION

Lemma 1. *Suppose that an instance B inherits from an instance A release dates and processing times of jobs. Then for any feasible (for both instances) schedule S we have*

$$L^B(S) - L^A(S) \leqslant \max_{j \in N} \{d_j^A - d_j^B\}. \tag{1}$$

Proof. Given $i \in N$, we have

$$L^A(S) + \max_{j \in N} \{d_j^A - d_j^B\} \geqslant c_i(S) - d_i^A + d_i^A - d_i^B = c_i(S) - d_i^B.$$

Therefore,

$$L^A(S) + \max_{j \in N} \{d_j^A - d_j^B\} \geqslant \max_{i \in N} (c_i(S) - d_i^B) = L^B(S).$$

Lemma 1 is proved. □

Lemma 2. *Suppose that an instance B inherits from an instance A release dates and processing times of jobs. Then*

$$0 \leqslant L^A(S^A_{\pi^B}) - L^A(S^A) \leqslant \rho_d(A, B).$$

Proof. Substituting S^A for S in (1), we obtain

$$L^A(S^A) + \max_{j\in N}\{d_j^A - d_j^B\} \geqslant L^B(S^A) = L^B(S^B_{\pi^A}). \tag{2}$$

By trading the places of notations of A and B in (1), we obtain

$$L^B(S^B) + \max_{j\in N}\{d_j^B - d_j^A\} \geqslant L^A(S^A_{\pi^B}). \tag{3}$$

By the definition of S^B, we have

$$L^B(S^B_{\pi^A}) \geqslant L^B(S^B). \tag{4}$$

Now it follows from (2)–(4) that

$$L^A(S^A) + \max_{j\in N}\{d_j^A - d_j^B\} \geqslant L^A(S^A_{\pi^B}) - \max_{j\in N}\{d_j^B - d_j^A\}$$

or $L^A(S^A) + \rho_d(A,B) \geqslant L^A(S^A_{\pi^B}) \geqslant L^A(S^A)$. Lemma 2 is proved. □

Since $\rho_d(A,B) = \rho_d(B,A)$, by trading the places of A and B, we obtain from Lemma 2

Corollary 1. $\rho_d(A,B) \geqslant L^B(S^B_{\pi^A}) - L^B(S^B) \geqslant 0$.

Lemma 3. *Let V and W be mutually inverse instances with job set N, and let π and π' be mutually inverse permutations from $\Pi(N)$. Then $L^V(S^V_\pi) = L^W(S^W_{\pi'})$.*

Proof. Let $\Delta = \Delta(V, S^V_\pi)$. Since S^V_π is fully feasible for $V(\Delta)$; the schedule S', inverse to S^V_π, is fully feasible for the instance W' inverse to $V(\Delta)$. This means that $L^{W'}(S') \leqslant 0$. Note that the instance W' differs from the instance W (inverse to V) in that all r_j are decreased by Δ. So, if we shift a schedule S' to the right by Δ, the schedule S'' obtained will be feasible for W, with $L^W(S'') \leqslant \Delta = L^V(S^V_\pi)$. Since the jobs are sequenced in S'' by π', we have $L^W(S^W_{\pi'}) \leqslant L^W(S'') \leqslant L^V(S^V_\pi)$.

By trading the places of V and W, as well as π and π', we will obtain the inverse inequality $L^V(S^V_\pi) \leqslant L^W(S^W_{\pi'})$. This implies the equality $L^V(S^V_\pi) = L^W(S^W_{\pi'})$. Lemma 3 is proved. □

Corollary 2. *If a permutation π is optimal for an instance V then the inverse permutation π' is optimal for the instance inverse to V.*

Lemma 4. *Suppose that an instance C inherits from an instance B processing times and due dates. Then*

$$0 \leqslant L^B(S^B_{\pi^C}) - L^B(S^B) \leqslant \rho_r(B,C).$$

Proof. Consider two instances E and F with parameters $r_j^E = -d_j^B$, $p_j^E = p_j^B$, $d_j^E = -r_j^B$ and $r_j^F = -d_j^C$, $p_j^F = p_j^C$, $d_j^F = -r_j^C$ inverse to B and C. Let π^E and π^F be the permutations inverse to π^B and π^C. Then by Corollary 2 the permutations π^E and π^F are optimal for the instances E and F. By Lemma 2,

$$\rho_d(E,F) \geqslant L^E(S^E_{\pi^F}) - L^E(S^E) \geqslant 0.$$

By Lemma 3, we have $L^B(S^B) = L^E(S^E)$ and $L^B(S^B_{\pi^C}) = L^E(S^E_{\pi^F})$. Hence,

$$\rho_d(E,F) \geqslant L^B(S^B_{\pi^C}) - L^B(S^B) \geqslant 0. \tag{5}$$

But we have

$$\rho_d(E,F) = \max_{j\in N}\{d_j^E - d_j^F\} + \max_{j\in N}\{d_j^F - d_j^E\} = \max_{j\in N}\{r_j^C - r_j^B\} + \max_{j\in N}\{r_j^B - r_j^C\} = \rho_r(B,C),$$

which together with (5) implies Lemma 4. □

Theorem 1. *Suppose that an instance C inherits the job processing times from A. Then*

$$0 \leqslant L^A(S^A_{\pi^C}) - L^A(S^A) \leqslant \rho(A, C).$$

Proof. The inequality $0 \leqslant L^A(S^A_{\pi^C}) - L^A(S^A)$ follows directly from the optimality of S^A for A. Let us prove that $L^A(S^A_{\pi^C}) - L^A(S^A) \leqslant \rho(A, C)$.

Consider the instance B with job parameters $r^B_j = r^A_j$, $p^B_j = p^A_j = p^C_j$, $d^B_j = d^C_j$ and its optimal schedule S^B. By Lemma 4,

$$L^B(S^B_{\pi^C}) - L^B(S^B) \leqslant \rho_r(B, C) = \rho_r(A, C). \tag{6}$$

By the optimality of the sequence π^B for B, we have

$$L^B(S^B) \leqslant L^B(S^B_{\pi^A}). \tag{7}$$

By Lemma 1, for the instances A and B and the schedules $S^A_{\pi^C} = S^B_{\pi^C}$ and $S^B_{\pi^A} = S^A_{\pi^A} = S^A$ we obtain

$$L^A(S^A_{\pi^C}) - L^B(S^B_{\pi^C}) \leqslant \max_{j \in N}\{d^B_j - d^A_j\}, \tag{8}$$

$$L^B(S^B_{\pi^A}) - L^A(S^A) \leqslant \max_{j \in N}\{d^A_j - d^B_j\}. \tag{9}$$

Summing up (6)–(9), we obtain

$$L^A(S^A_{\pi^C}) - L^A(S^A) \leqslant \rho_r(A, C) + \rho_d(A, B) = \rho_r(A, C) + \rho_d(A, C) = \rho(A, C).$$

Theorem 1 is proved. □

Corollary 3. $0 \leqslant L^C(S^C_{\pi^A}) - L^C(S^C) \leqslant \rho(A, C)$.

The result of Theorem 1 can be applied to Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ in conditions of uncertainty of release and/or due dates of jobs. Suppose that for a given instance A we somehow found an optimal job sequence π_A, after which a new information on the values of the parameters $\{r_j, d_j\}$ was received (while job processing times remained unchanged). Let C denote the new instance. The question arises: What is the quality of the solution obtained by applying the sequence π_A to the instance C? Corollary 3 enables us to bound the absolute error of such solution by the distance ρ between the instances A and C.

Another application of Corollary 3 is related to elaboration of efficient approximation algorithms for the NP-hard problem $\langle 1 \mid r_j \mid L_{\max}\rangle$. The general scheme of the approximate solution based on the algorithms elaborated for special polynomially solvable cases of our problem is considered in the next section together with different variants of application of this scheme.

3. A SCHEME OF APPROXIMATE SOLUTION

The general idea of approximate solution of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ consists in performing the following two steps: At the first step, given an instance A, we find a transformation of its parameters r^A_j, d^A_j such that the obtained instance C with job parameters $r^C_j, p^C_j = p^A_j, d^C_j$ would belong to a given polynomial time solvable class of instances of the original (NP-hard) problem. At the second step, to find a solution for the instance C, we apply the known algorithm capable of solving the class of instances in polynomial time. After that, it remains to apply the job sequence found by the algorithm to constructing an early schedule for instance A.

By Theorem 1, the absolute error of such solution cannot be greater than the distance $\rho(A, C)$ between instances A and C. It is clear that the minimum bound on the error of such solution will be guaranteed in the case when at the first step of this scheme an instance C is found that minimizes the value of $\rho(A, C)$. Thus, given an instance A of the problem under study, its approximate solution reduces to searching for another instance C belonging to a given class of instances $\mathcal{C}$ (known to be polynomially solvable) and being the nearest one to A among all instances in $\mathcal{C}$.

Consider the case when a polynomial time solvable class of instances of our problem is defined by a system of linear inequalities of the following type:

$$\boldsymbol{A} * R^C + \boldsymbol{B} * P^C + \boldsymbol{C} * D^C \leqslant H \tag{10}$$

(under the constraints $p_j^C \geqslant 0$, $j \in N$), where

$$R^C = (r_1^C, \ldots, r_n^C)^T, \qquad P^C = (p_1^C, \ldots, p_n^C)^T, \qquad D^C = (d_1^C, \ldots, d_n^C)^T,$$

$\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ are matrices of size $m \times n$ and $H = (h_1, \ldots, h_m)^T$ is an m-dimensional vector (the upper index ^{T} is used here to denote the transposed matrix). Then to find in such class of instances the particular instance C minimizing the distance $\rho(A, C)$ (for a given instance A), it is sufficient to solve the following linear program: Find

$$\min\,(x_d - y_d + x_r - y_r) \tag{11}$$

under the constraints

$$\begin{aligned} &y_d \leqslant d_j^A - d_j^C \leqslant x_d, \qquad y_r \leqslant r_j^A - r_j^C \leqslant x_r, \\ &p_j^A = p_j^C, \qquad j \in N, \\ &\boldsymbol{A} * R^C + \boldsymbol{B} * P^C + \boldsymbol{C} * D^C \leqslant H. \end{aligned} \tag{11a}$$

Of course, while using in the general reduction scheme some concrete polynomially solvable classes of instances, it is desirable to use most efficient algorithms of minimizing $\rho(A, C)$. To this end, while solving (11a), one should maximally take into account the specificity of the system resulting from the specificity of the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and H.

For instance, in the special case [1], when $d_j - r_j - p_j = 0,\ j \in N$, in the system of linear inequalities (10) the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and H are specified as follows:

$$\boldsymbol{A} = \boldsymbol{B} = (\boldsymbol{I} \oplus (-\boldsymbol{I}))^T, \qquad \boldsymbol{C} = ((-\boldsymbol{I}) \oplus \boldsymbol{I})^T, \qquad H = (h)^T,$$

where $\boldsymbol{I}$ is the identity matrix of size n, while h is the $2n$-dimensional zero vector and $\boldsymbol{A} \oplus \boldsymbol{B}$ denotes the concatenation of a matrix $\boldsymbol{A}$ of size $l \times p$ and a matrix $\boldsymbol{B}$ of size $l \times q$.

For another illustration, let us consider the class $\mathcal{C}$ of instances of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ with $d_j = \delta$, $j \in N$, where δ is a constant. To obtain the optimal schedule S' for an arbitrary instance $C \in \mathcal{C}$, it is sufficient to sequence the jobs in nondecreasing order of their release dates. The schedule S' can be found in $O(n \log n)$ time. To reduce an arbitrary instance A of our problem to $C \in \mathcal{C}$, it is sufficient to equalize all due dates to some constant δ. Let us formulate a linear program similar to (11)–(11a) for finding the instance C with the minimal distance $\rho(A, C)$. Find

$$\min\,(x_d - y_d) \tag{12}$$

subject to

$$y_d \leqslant d_j^A - \delta \leqslant x_d, \qquad j \in N. \tag{12a}$$

The solution of the linear program (12)–(12a) is an arbitrary value of δ, and x_d, y_d such that $x_d - y_d = \max_{j\in N} d_j^A - \min_{j\in N} d_j^A$, i.e., the distance $\rho(A, C)$ for a given instance A is the same for any instance $C \in \mathcal{C}$, and the absolute error of S' in any case cannot be larger than the difference between the maximum and minimum due dates of jobs of A.

The third example of a polynomially solvable class of instances that can be written as a linear system of the type (10) is the class defined by Hoogeveen [7] (let us call it a *Hoogeveen class*). An instance $C = \{(r_j^C, p_j^C, d_j^C) \mid j \in N\}$ belongs to the Hoogeveen class, if there is a constant β such that for each job $j \in N$ we have

$$d_j^C - r_j^C - p_j^C \leqslant \beta \leqslant d_j^C - r_j^C. \tag{13}$$

Note that it is sufficient to consider the class defined by this property with $\beta = 0$. Indeed, if for a given instance A we found an instance C belonging to the Hoogeveen class with $\beta \neq 0$, then instead of it we

could take the instance C' obtainable from C by decreasing all d_j by β. At that, C' will belong to the Hoogeveen class with $\beta = 0$, and $\rho(A, C) = \rho(A, C')$.

The Hoogeveen class with $\beta = 0$ can be defined by the inequalities

$$d_j^C - r_j^C - p_j^C \leqslant 0, \qquad -d_j^C + r_j^C \leqslant 0, \qquad j \in N, \tag{14}$$

which, evidently, can be easily written in the form of (10).

Some other variants of application of the above-described general scheme not based on the solution of a linear program of the type (11)–(11a) are considered in the following subsections.

3.1. A Variant of the Scheme Based on the L-Class of Instances

Let us consider the polynomially solvable class L of instances of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ proposed by Lazarev [10].

We say that an instance $C = \{(r_j^C, p_j^C, d_j^C) \mid j \in N\}$ *belongs to class* L, if there exists a numbering of jobs $\{1, 2, \ldots, n\}$ such that

$$d_1^C \leqslant \cdots \leqslant d_n^C; \qquad \Delta_1^C \geqslant \cdots \geqslant \Delta_n^C, \tag{15}$$

where $\Delta_j^C = d_j^C - r_j^C - p_j^C$ denotes the *float time* of job j.

Given A, let us define the function

$$\rho^L(A) = \max_{i,j \in N} \rho_{ij}^L(A), \tag{16}$$

where

$$\rho_{ij}^L(A) = \min\{d_j^A - d_i^A, \Delta_j^A - \Delta_i^A\}. \tag{17}$$

It can be easily seen that $\rho^L(A) \geqslant 0$ for each A (for example, $\rho_{ij}^L(A) = 0$ for $i = j$); at that, $\rho^L(A) = 0$, if and only if A belongs to L.

The exact solution for any instance of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ from the class L can be obtained in time $O(n^3 \log n)$ [10].

Suppose, we are given an instance A not belonging to class L. Let us apply to it the above-described scheme of approximate solution as follows: We reduce A to some instance C that inherits from A the job processing times and belongs to class L. Note that the class L cannot be written by means of a linear system like (10), since it is not a convex polyhedron in $3n$-dimensional parametric space. For instance, as follows from (15), in the case that $p_j \equiv \text{const}$ it represents a family of $n!$ cones whose union is not convex. That is why while implementing the general scheme with class L, we apply a nonstandard approach at the first step of the scheme.

The following theorem enables one to find an instance C of class L which provides the minimum to the distance $\rho(A, C)$.

Theorem 2. *For any instance A of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ and any instance $C \in L$ that inherits the job processing times from instance A, the following bound on the distance between A and C is valid:*

$$\rho(A, C) \geqslant \rho^L(A). \tag{18}$$

Bound (18) *is attainable at some instance C that can be found in $O(n \log n)$ time.*

Proof. For each $C \in L$ the parameters of every two jobs $i, j \in N$ satisfy one of the following inequalities:

$$d_i^C - d_j^C \geqslant 0 \tag{19}$$

or

$$\Delta_i^C - \Delta_j^C \geqslant 0. \tag{20}$$

If (19) holds for jobs i, j then from the definition of the distance $\rho(A, C)$ and (17) we derive

$$\rho(A, C) \geqslant \rho_d(A, C) \geqslant (d_j^A - d_j^C) + (d_i^C - d_i^A) \geqslant d_j^A - d_i^A \geqslant \rho_{ij}^L(A). \tag{21}$$

Alternatively, if (20) holds for jobs i, j then (17), (20), and the definition of the metric $\rho(A, C)$ imply

$$\begin{aligned}\rho(A,C) &\geqslant (d_j^A - d_j^C) + (d_i^C - d_i^A) + (r_j^C - r_j^A) + (r_i^A - r_i^C)\\ &= (d_j^A - r_j^A - p_j) - (d_j^C - r_j^C - p_j) + (d_i^C - r_i^C - p_i) - (d_i^A - r_i^A - p_i)\\ &= \Delta_j^A - \Delta_j^C + \Delta_i^C - \Delta_i^A \geqslant \Delta_j^A - \Delta_i^A \geqslant \rho_{ij}^L(A). \quad (22)\end{aligned}$$

The desired bound (18) follows from (21), (22), and (16):

$$\rho(A,C) \geqslant \max_{i,j\in N} \rho_{ij}^L(A) = \rho^L(A).$$

To prove that (18) is attainable at some instance C of class L, we construct an instance C that inherits from A the processing times and release dates of jobs. Once the parameters p_j and r_j do not differ in A and C, we will denote them without the upper indices A or C.

Let us number the jobs of instance A by nondecreasing $r_j + p_j$:

$$r_1 + p_1 \leqslant \cdots \leqslant r_n + p_n. \quad (23)$$

We define the increasing sequence of *dividing indices* $j_0 < j_1 < \cdots < j_K = n$ as follows:

$$\begin{aligned}&j_0 := 0;\ \ k := 0;\\ &\textbf{while}\ \ j_k < n\ \ \textbf{do}\quad \{k := k+1;\ j_k := \max \arg \min_{\{j\,|\,j_{k-1}<j\leqslant n\}} d_j^A\}.\end{aligned} \quad (24)$$

It can be easily seen that

$$d_{j_1}^A < d_{j_2}^A < \cdots < d_{j_K}^A. \quad (25)$$

Put

$$d_j^C = \begin{cases} d_{j_1}^A, & \text{for } j \leqslant j_1,\\ \min\{d_{j_k}^A, d_j^A - \Delta_j^A + \Delta_{j_{k-1}}^C\}, & \text{for } j_{k-1} < j \leqslant j_k,\ k > 1.\end{cases} \quad (26)$$

Let us prove that the so-defined instance C belongs to L. To this end, we first prove the following inequalities by induction on $k = 1, \ldots, K$:

$$d_j^C \leqslant d_j^A \qquad (j \leqslant j_k); \quad (27)$$

$$d_i^C \leqslant d_j^C \qquad (i < j \leqslant j_k); \quad (28)$$

$$\Delta_i^C \geqslant \Delta_j^C \qquad (i < j \leqslant j_k). \quad (29)$$

The induction base.

Let $k = 1$. For any $j \leqslant j_1$, we have $d_{j_1}^A \leqslant d_j^A$ by (24), while $d_j^C = d_{j_1}^A$ by (26), which implies (27). At that, for any i and j such that $i < j \leqslant j_1$, inequality (28) holds as equality (due to (26)), while (29) follows from the relations:

$$\Delta_i^C = d_i^C - r_i - p_i \overset{\text{from (26)}}{=} d_{j_1}^A - r_i - p_i \overset{\text{from (23)}}{\geqslant} d_{j_1}^A - r_j - p_j \overset{\text{from (26)}}{=} d_j^C - r_j - p_j = \Delta_j^C.$$

The induction step.

Assume that (27)–(29) are valid for $k = k' - 1 < K$. Let us prove them for $k = k'$.

For all j $(j_{k'-1} < j \leqslant j_{k'})$, we can derive $d_j^C \leqslant d_{j_{k'}}^A \leqslant d_j^A$ from (26) and (24), which implies (27) by the induction hypothesis.

To prove (28) and (29), by the induction hypothesis, it is sufficient to consider the following variants of the indices i and j: a) $i = j_{k'-1} < j \leqslant j_{k'}$; b) $j_{k'-1} < i < j \leqslant j_{k'}$.

For any j $(j_{k'-1} < j \leqslant j_{k'})$, from (26) we have either c) $d_j^C = d_{j_{k'}}^A$ or d) $d_j^C = r_j + p_j + d_{j_{k'-1}}^C - r_{j_{k'-1}} - p_{j_{k'-1}}$.

In case c),

$$d^C_{j_{k'-1}} \overset{\text{from (27)}}{\leqslant} d^A_{j_{k'-1}} \overset{\text{from (25)}}{\leqslant} d^A_{j_{k'}} = d^C_j.$$

In case d), (23) implies $d^C_j \geqslant d^C_{j_{k'-1}}$. In both cases we have (28) for the variant of indices a). Let us show that (28) is valid for the variant of indices b), as well.

In case c), from the definition (26) we have $d^C_i \leqslant d^A_{j_{k'}} = d^C_j$. In case d),

$$d^C_i \overset{\text{from (26)}}{\leqslant} r_i + p_i + \Delta^C_{j_{k'-1}} \overset{\text{from (23)}}{\leqslant} r_j + p_j + \Delta^C_{j_{k'-1}} = d^C_j.$$

Thus, in both cases we obtain (28) for the variant of indices b).

The relations

$$\begin{aligned}\Delta^C_{j_{k'-1}} &\geqslant \min\{\Delta^C_{j_{k'-1}}, d^A_{j_{k'}} - d^A_j + \Delta^A_j\}\\ &= \min\{\Delta^C_{j_{k'-1}} + d^A_j - \Delta^A_j, d^A_{j_{k'}}\} - d^A_j + \Delta^A_j = d^C_j - r_j - p_j = \Delta^C_j\end{aligned}$$

imply (29) for the variant of indices a).

In case b), inequality (29) follows from

$$\begin{aligned}\Delta^C_i &= d^C_i - r_i - p_i \overset{\text{from (26)}}{=} \min\{d^A_{j_{k'}} - r_i - p_i, \Delta^C_{j_{k'-1}}\}\\ &\overset{\text{from (23)}}{\geqslant} \min\{d^A_{j_{k'}} - r_j - p_j, \Delta^C_{j_{k'-1}}\} \overset{\text{from (26)}}{=} d^C_j - r_j - p_j = \Delta^C_j.\end{aligned}$$

Thus, (27)–(29) are proved for all k. This, in particular, implies that C belongs to class L.

To prove that (18) is attained at C, we first note that if to represent $\rho^L_{ij}(A)$ (defined by (17)) in the form $\rho^L_{ij}(A) = \min\{d^A_j - d^A_i, d^A_j - d^A_i + (r_i + p_i) - (r_j + p_j)\}$, then due to (23) we will have

$$\rho^L_{ij}(A) = \begin{cases} d^A_j - d^A_i, & \text{for } i \geqslant j,\\ \Delta^A_j - \Delta^A_i, & \text{for } i < j.\end{cases} \tag{30}$$

From (26) with $k \geqslant 2$ and $j = j_k$ we derive $d^C_{j_k} = \min\{d^A_{j_k}, d^A_{j_k} - \Delta^A_{j_k} + \Delta^C_{j_{k-1}}\}$. Subtracting from both parts of this equality the amount $r_{j_k} + p_{j_k}$, we obtain

$$\Delta^C_{j_k} = \min\{\Delta^A_{j_k}, \Delta^C_{j_{k-1}}\}. \tag{31}$$

By (31) and the equality $\Delta^C_{j_1} = \Delta^A_{j_1}$, we infer for every $k \geqslant 1$:

$$\Delta^C_{j_k} = \min_{\nu \leqslant k} \Delta^A_{j_\nu}. \tag{32}$$

Next, let us prove for every j the inequality

$$d^A_j - d^C_j \leqslant \rho^L(A). \tag{33}$$

For $j \leqslant j_1$, due to (30), we have $d^A_j - d^C_j = d^A_j - d^A_{j_1} = \rho^L_{j_1 j}(A) \leqslant \rho^L(A)$.

For any j, $j_{k-1} < j \leqslant j_k$, and any $k \geqslant 2$, we have

$$d^A_j - d^C_j = d^A_j - \min\{d^A_{j_k}, d^A_j - \Delta^A_j + \Delta^C_{j_{k-1}}\} = \max\{d^A_j - d^A_{j_k}, \Delta^A_j - \Delta^C_{j_{k-1}}\} =$$

(by (32) for some $\nu \leqslant k-1$)

$$= \max\{d^A_j - d^A_{j_k}, \Delta^A_j - \Delta^A_{j_\nu}\} \overset{\text{from (30)}}{=} \max\{\rho^L_{j_k j}(A), \rho^L_{j_\nu j}(A)\} \leqslant \rho^L(A),$$

which implies (33). Finally, since $\rho_r(A,C)=0$, we have

$$\rho(A,C)=\rho_d(A,C)=\max_j\{d_j^A-d_j^C\}+\max_j\{d_j^C-d_j^A\}\overset{\text{from (33) and (27)}}{\leqslant}\rho^L(A),$$

which together with (18) implies $\rho(A,C)=\rho^L(A)$.

The overall running time of the algorithm of finding the desired instance C is formed of three components, being complexities of the steps:

1) job numbering according to (23);
2) finding the sequence of dividing indices according to (24);
3) finding new due dates according to (26).

The first step can be performed by means of a standard procedure that sequences n numbers in time $O(n\log n)$. The third step, clearly, can be done in linear time. As far as the second step is concerned, the straightforward realization of formula (24) requires time $O(n^2)$. However, the amount of calculation at this step can be significantly reduced if to look through the jobs in the opposite order and compute the dividing indices as follows.

Let us assume (just for ease of notation) that the number K of dividing indices is known. Put $j_K=n$. Assume that $1<j_k<\cdots<j_K$ are already defined. We proceed with consecutively considering jobs $j=j_k-1,j_k-2,\ldots$, and for j_{k-1} we take the first index $j<j_k$ such that $d_j^A<d_{j_k}^A$ (if any). It can be easily checked that the so-defined sequence of indices $j_1,\ldots,j_K$ meets (24); at that, all inequalities (25) hold as equalities, and the running time of this procedure is linear in n. Theorem 2 is proved. □

In fact, in the algorithm of computing the desired instance C it is expedient from the very beginning to number the jobs of instance A in the reverse order: in the nonincreasing order of r_j+p_j. The formal record of such algorithm is presented below. (It is assumed that the jobs of instance A are already numbered in the nonincreasing order of $r_j^A+p_j^A$.)

Algorithm 1 (of finding the instance from L nearest to a given instance A)

FOR $j:=1$ TO n $\quad r_j^C:=r_j^A;\ p_j^C:=p_j^A$ END FOR

$k:=1;\ j_1=1;$

FOR $i:=2$ TO n IF $d_i^A<d_{j_k}^A$ THEN $k:=k+1;\ j_k:=i$ END IF END FOR

FOR $j:=n$ DOWNTO j_k $\quad d_j^C:=d_{j_k}^A$ END FOR

$\Delta_{j_k}^C:=d_{j_k}^C-r_{j_k}^C-p_{j_k}^C;$

FOR $\nu:=k$ DOWNTO 2 $\quad$ FOR $j:=j_\nu-1$ DOWNTO $j_{\nu-1}$

$d_j^C:=\min\{d_{j_{\nu-1}}^A,d_j^A-\Delta_j^A+\Delta_{j_\nu}^C\};\ \Delta_j^C:=d_j^C-r_j^C-p_j^C$ $\quad$ END FOR END FOR

Example. Let us follow Algorithm 1. Suppose, we are given an instance A of Problem $\langle 1\mid r_j\mid L_{\max}\rangle$ with the job parameters specified in Table 1, the jobs numbered in the nonincreasing order of $r_j^A+p_j^A$.

Table 1. Job parameters of A

j	1	2	3	4	5	6	7	8
r_j^A	7	5	3	5	1	2	3	0
p_j^A	2	4	5	3	5	3	1	4
d_j^A	16	18	13	14	15	11	12	14

Let us start our algorithm with the job set $N=\{1,\ldots,8\}$. After completing the first cycle, for all $j\in N$ we obtain $r_j^C=r_j^A$ and $p_j^C=p_j^A$. As a result of the second cycle, the dividing indices are found: $j_1=1$, $j_2=3$, $j_3=6$. At the third cycle we compute: $d_6^C=d_7^C=d_8^C=d_6^A=11$.

The cycle on ν is started:

$\nu = 3$

$$d_5^C = \min\{d_3^A, d_6^C - r_6^C - p_6^C + r_5^A + p_5^A\} = 12,$$
$$d_4^C = \min\{d_3^A, d_6^C - r_6^C - p_6^C + r_4^A + p_4^A\} = 13,$$
$$d_3^C = \min\{d_3^A, d_6^C - r_6^C - p_6^C + r_3^A + p_3^A\} = 13.$$

$\nu = 2$

$$d_2^C = \min\{d_1^A, d_3^C - r_3^C - p_3^C + r_2^A + p_2^A\} = 14,$$
$$d_1^C = \min\{d_1^A, d_3^C - r_3^C - p_3^C + r_1^A + p_1^A\} = 14.$$

The algorithm completes the work.

We thereby obtained an instance C from L. The job parameters of C are specified in Table 2. The bound on the absolute error of our solution is equal to

$$\rho(A, C) = \max_{j \in N}\{d_j^A - d_j^C\} + \max_{j \in N}\{d_j^C - d_j^A\} = d_2^A - d_2^C + 0 = 4.$$

Table 2. Job parameters of C

j	1	2	3	4	5	6	7	8
r_j^A	7	5	3	5	1	2	3	0
p_j^A	2	4	5	3	5	3	1	4
d_j^A	14	14	13	13	12	11	11	11

3.2. The Version of the Scheme on the Base of the Hoogeveen Class

The following version of the scheme of approximate solution is based on the transformation of any given instance to an instance from the polynomial-time solvable class H defined by Hoogeveen [7] (for solving instances from that class there is an algorithm of running time $O(n^2 \log n)$).

As follows from (13), an instance $A = \{(r_j^A, p_j^A, d_j^A) \mid j \in N\}$ belongs to class H if and only if

$$\Delta_{\max}^A \doteq \max_i \Delta_i^A \leqslant d_j^A - r_j^A, \qquad j \in N. \tag{34}$$

Let us define the function ρ^H on the set of instances:

$$\rho^H(A) = \max_{j \in N}\{\Delta_{\max}^A - d_j^A + r_j^A\}. \tag{35}$$

It is clear that a given instance A belongs to H if and only if $\rho^H(A) \leqslant 0$.

Theorem 3. *For any instance A of Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$ and any instance $C \in H$ that inherits the job processing times from instance A, the following bound on the distance between A and C is valid*

$$\rho(A, C) \geqslant \rho^H(A). \tag{36}$$

Bound (36) *is attained at some instance C that inherits from A the job processing times and release dates (the instance can be found in time $O(n)$).*

Proof. As is noted above, it is sufficient to prove (36) for any instance C that meets (14). By the definition of $\rho(A, C)$, we have

$$\rho(A,C) = \max_i(d_i^C - d_i^A) + \max_j(d_j^A - d_j^C) + \max_j(r_j^C - r_j^A) + \max_i(r_i^A - r_i^C)$$
$$\geqslant d_i^C - d_i^A + d_j^A - d_j^C + r_j^C - r_j^A + r_i^A - r_i^C = (d_j^A - r_j^A - p_j)$$
$$+ (r_j^C - d_j^C + p_j) + (r_i^A - d_i^A) + (d_i^C - r_i^C) \overset{\text{from (13)}}{\geqslant} (d_j^A - r_j^A - p_j) + (r_i^A - d_i^A).$$

Since these relations are valid for all $j, i \in N$, applying (35) we obtain (36):

$$\rho(A,C) \geqslant \max_j(d_j^A - r_j^A - p_j) + \max_i(r_i^A - d_i^A) = \rho^H(A).$$

Let us prove that bound (36) is attained at some instance $C \in H$. (Since the parameters p_j and r_j not to be distinguished for A and C, we will denote them without the upper index A or C respectively.)

Once the instance A does not meet (34), there are jobs j such that

$$\Delta_{\max}^A > d_j^A - r_j. \tag{37}$$

Let N' be the set of such j. For each job $j \in N$ we define the new (enlarged) value of d_j^C by the formula

$$d_j^C := d_j^A + (\Delta_{\max}^A - d_j^A + r_j)^+. \tag{38}$$

Then

$$\Delta_{\max}^C = \Delta_{\max}^A. \tag{39}$$

Indeed, since $d_j^C \geqslant d_j^A$ for any $j \in N$ (at that, r_j and p_j remained the same), we have $\Delta_{\max}^C \geqslant \Delta_{\max}^A$.

Let us prove the converse statement. If $j \notin N'$ then $\Delta_j^C = \Delta_j^A \leqslant \Delta_{\max}^A$. For $j \in N'$, from (37) and (38) we obtain $d_j^C = \Delta_{\max}^A + r_j$. Hence, $\Delta_j^C = \Delta_{\max}^A - p_j \leqslant \Delta_{\max}^A$. Thus, $\Delta_{\max}^C \leqslant \Delta_{\max}^A$, which implies (39).

Let us prove that the instance C meets (34).

For any $j \in N \setminus N'$ inequalities (34) follow from $\Delta_{\max}^C = \Delta_{\max}^A \leqslant d_j^A - r_j = d_j^C - r_j$.

If $j \in N'$ then $d_j^C - r_j = \Delta_{\max}^A = \Delta_{\max}^C$. Thus, C belongs to class H. Its distance from A is determined by the relations

$$\rho(A,C) = \max_{j \in N}\{d_j^C - d_j^A\} = \max_{j \in N'}\{\Delta_{\max}^A - d_j^A + r_j\} = \rho^H(A).$$

Theorem 3 is proved. □

For the instance from Table 1 the bound on the absolute error is equal to $\rho(A, C) = 1$.

4. CONCLUSION

In this paper some general scheme is presented for finding approximate solutions to Problem $\langle 1 \mid r_j \mid L_{\max}\rangle$. The scheme is based on a transformation of a given instance to another instance belonging to a known polynomially solvable class of instances. For two versions of the scheme (using the polynomial-time solvable classes L and H), we derived analytical formulas that, given an instance of the problem, enable one to compute upper bounds on the absolute errors of approximate solutions found by means of those two methods.

Further research could address the theoretical and practical comparison of the performance guarantees of our approximation methods and other approaches.

ACKNOWLEDGMENTS

The third author was supported by the Russian Foundation for Basic Research (projects no. 05–01–00960 and 05–06–90606HHC)

REFERENCES

1. J. Adams, E. Balas, and D. Zawack, "The Shifting Bottleneck Procedure for Job Shop Scheduling," J. Manag. Sci. **34** (3), 391–401 (1988).
2. K. R. Baker, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnoy Kan, "Preemptive Scheduling of a Single Machine to Minimize Maximum Cost Subject to Release Dates and Precedence Constraints," J. Oper. Res. **31** (2), 381–386 (1983).
3. K. R. Baker and Z. S. Su, "Sequencing with Due Dates and Early Start Times to Minimize Tardiness," J. Naval Res. Logist. Quart. **21** (1), 171–176 (1974).
4. J. Carlier, "The One-Machine Sequencing Problem," European J. Oper. Res. **11** (1), 42–47 (1982).
5. R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnoy Kan, "Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey," Ann. Discrete Math. **5**, 287–326 (1979).
6. L. A. Hall and D. B. Shmoys, "Jackson's Rule for One-Machine Scheduling: Making a Good Heuristic Better," J. Math. Oper. Res. **17** (1), 22–35 (1992).
7. J. A. Hoogeveen, "Minimizing Maximum Promptness and Maximum Lateness on a Single Machine," J. Math. Oper. Res. **21** (1), 100–114 (1996).
8. J. R. Jackson, "Scheduling a Production Line to Minimize Maximum Tardiness," Manag. Sci. Res. Project, No. 43. UCLA, 1955.
9. B. J. Lageweg, J. K. Lenstra, and A. H. G. Rinnoy Kan, "Minimizing Maximum Lateness on One Machine: Computational Experience and Some Applications," J. Statistica Neerlandica **30** (1), 25–41 (1976).
10. A. A. Lazarev, *Efficient Algorithms for One-Machine Scheduling Problems with Due Dates on Job Completion Times*, Candidate's Dissertation (Kazansk. Gos. Univ., Kazan, 1989).
11. A. A. Lazarev and O. N. Shulgina, "A Pseudo-Polynomial Algorithm for an NP-Hard Scheduling Problem of Minimizing the Maximum Job Lateness," in *Proceedings of the XIth International Baikal Seminar on Optimization Methods and Applications, Irkutsk, Russia, 1998* (Energy Systems Inst., Irkutsk, 1998), pp. 163–167.
12. E. L. Lawler, "Optimal Sequencing of a Single Machine Subject to Precedence Constraints," J. Manag. Sci. **19** (5), 544–546 (1973).
13. J. K. Lenstra, A. H. G. Rinnoy Kan, and P. Brucker, "Complexity of Machine Scheduling Problems," Annals of Oper. Res. **1**, 343–362 (1975).
14. M. Mastrolilli, "Efficient Approximation Schemes for Scheduling Problems with Release Dates and Delivery Times," J. of Scheduling **6** (6), 521–531 (2003).
15. G. McMahon and M. Florian, "On Scheduling with Ready Times and Due Dates to Minimize Maximum Lateness," J. Oper. Res. **23** (3), 475–482 (1975).
16. L. Péridy, E. Pinson, and D. Rivraux, "Using Short-Term Memory to Minimize the Weighted Number of Late Jobs on a Single Machine," European J. Oper. Res. **148** (3), 591–603 (2003).
17. C. N. Potts, "Analysis of a Heuristic for One Machine Sequencing with Release Dates and Delivery Times," J. Oper. Res. **28** (6), 1436–1441 (1980).
18. B. B. Simons, "A Fast Algorithm for Single Processor Scheduling," in *Proceedings of the 19th Annual Symposium Foundation of Computer Science* (Ann. Arbor, New York, 1978), pp. 246–252.