Article

AN INTERPOLATION-BASED POLYNOMIAL METHOD OF ESTIMATING THE OBJECTIVE FUNCTION VALUE IN SCHEDULING PROBLEMS OF MINIMIZING THE MAXIMUM LATENESS

Alexander Lazarev [1], Darya Lemtyuzhnikova [1,2] and Andrey Tyunyatkin [1,*]

[1] Institute of Control Sciences, 65 Profsoyuznaya street, 117997, Moscow, Russia
[2] Moscow Aviation Institute, 4 Volokolamskoe highway, Moscow, 125993, Russia
* Correspondence: a@tuniatk.in

Abstract: An approach to estimating the objective function value of minimization maximum lateness problem is proposed. It is shown how to use transformed instances to define a new continuous objective function. After that, using this new objective function, the approach itself is formulated. We calculate the objective function value for some polynomially solvable transformed instances and use them as interpolation nodes to estimate the objective function of the initial instance. What is more, two new polynomial cases, that are easy to use in the approach, are proposed. In the end of the paper numeric experiments are described and their results are provided.

Keywords: discrete mathematics; scheduling; optimization; interpolation; approximation; objective function.

1. Introduction

The vast majority of scheduling theory problems are NP-hard [1]. To solve such problems, it is common to use algorithms, the performance of which strongly depends on the input data. A new approach to estimating the objective function value of scheduling theory problems is proposed - the interpolation approach.

Algorithms for solving problems in the theory of schedules, considered, for example, in [1,2], can be used. Algorithms and methods from [3] can be used to work with random data, and metric interpolation speeds up their execution when processing difficult cases.

Since the interpolation approach works only with the values of the objective function, it can also be used to create schedules for multi-stage systems, solving problems, for example, using algorithms from [4].

For certainty, this article considers the solution of the problem of minimizing the maximum time offset $1|r_j|L_{\max}$.

New polynomial cases, that can be easily used in the interpolation approach, are defined. Using these cases and Lagrange interpolation [5,11], the objective function value is approximated.

Other interpolation methods[5] also can be used in the approach: for instance, Chebyshev interpolation[20] or Spline interpolation [21]. However these methods will be considered in our future work, while in this paper we will keep using Lagrange interpolation polynome.

2. The problem of minimizing the maximum lateness for single machine

2.1. The problem statement

In the problem $1|r_j|L_{\max}$[1,7,10], which we will consider, a set of n jobs is given $A = \{1, ..., n\}$. For each job j, the following parameters are set: the release time r_j, the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

processing time p_j and the due date d_j [1]. By *schedule* π we mean some permutation of the jobs of the set A. Let's enter the completion time of the job j with the schedule π:

$$C_j(\pi) = \max_{\pi}\left\{ r_j, \max_{(k\to j)_\pi} C_k(\pi) \right\} + p_j. \tag{1}$$

Here $(k \to j)_\pi$ is the set of jobs that are processed before the work of j with the schedule of π.

The lateness of the job j in the schedule π is defined as follows:

$$L_j(\pi) = C_j(\pi) - d_j. \tag{2}$$

Thus, the task of minimizing the maximum lateness is to find such schedule π_0, at which the objective function obtains the minimum value:

$$L_{\max}(\pi_0) = \min_{\pi} \max_{j=1,\ldots n} \{C_j(\pi) - d_j\}. \tag{3}$$

This problem is NP-hard in the strong sense[6].

3. The feature space

In the paper each instance of the scheduling problem [1], consisting of n jobs, is considered as a point in a 3n-dimensional feature space [8,9] with coordinates $(r_1, r_2, ..., r_n, p_1, p_2, ..., p_n, d_1, d_2, ..., d_n)$.

For convenience, we will denote each instance as a $3 \times n$ matrix:

$$\begin{pmatrix} r_1 & r_2 & \ldots & r_n \\ p_1 & p_2 & \ldots & p_n \\ d_1 & d_2 & \ldots & d_n \end{pmatrix}$$

Let pick a point A in this space. Then the instance for which we want to solve the scheduling problem is an instance consisting of n jobs with r_j, p_j, d_j parameters specified by the coordinates of the point A.

More about the 3n-dimensional feature space can be found in [7].

4. The $r' = \alpha r$ transform

Definition 1. *The $r' = \alpha r$ (where α is an arbitrary non-negative real value) is a transform that matches the initial instance* $A = \begin{pmatrix} r_1 & r_2 & \ldots & r_n \\ p_1 & p_2 & \ldots & p_n \\ d_1 & d_2 & \ldots & d_n \end{pmatrix}$ *with the transfromed instance* $A' = \begin{pmatrix} \alpha r_1 & \alpha r_2 & \ldots & \alpha r_n \\ p_1 & p_2 & \ldots & p_n \\ d_1 & d_2 & \ldots & d_n \end{pmatrix}$.

Thus, the $r' = \alpha r$ transform multiplies all the release times of the instance by some factor α while keeping the processing times and due dates constant.

5. Introduction to the interpolation approach

Notation 1. *When writing A_α we refer to a transformed instance A' obtained from the initial instance A using the $r' = \alpha r$ transform with some coefficient α.*

Notation 2. *The optimal value of the $L_{\max}$ objective function obtained for the initial instance A will be denoted as $L^*_{\max}$.*

Now it is time to define the $L_{\max}(\alpha)$ function which will be used for interpolation later.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

Definition 2. *Function $L_{\max}(\alpha)$ recieves a real non-negative transform coefficient α value and returns the optimal value of the objective function obtained on the transformed instance A_α.*

The concept of the approach is that it is possible to draw a straight line through the point A in the 3n-dimensional feature space mentioned above, pick some other points on that line, solve the instances specified by those points and then, using interpolation [1,5], find an approximate value of the objective function at the point A.

Lagrange interpolation polynomial is defined as follows [5]:

$$L_m(x) = \sum_{k=0}^{m} \frac{\prod\limits_{i \neq k} (x - x_i)}{\prod\limits_{j \neq k} (x - x_j)} f(x_k). \tag{4}$$

Let presume we have calculated the objective function values for the n transformed instances $A_{\alpha_1} \dots A_{\alpha_n}$. Now we are willing to find the $L_{\max}$ value of the initial instance A. Using Lagrange interpolation polynomial (4) we will obtain the following formula:

$$L^*_{\max} = L_n(1) = \sum_{k=1}^{n} \frac{\prod\limits_{i \neq k} (1 - L_{\max}(\alpha_i))}{\prod\limits_{j \neq k} (1 - L_{\max}(\alpha_j))} L_{\max}(\alpha_k). \tag{5}$$

This procedure is formalized in the following algorithm.

Algorithm 1. *The algorithm receives the initial instance N and returns the estimated objective function value $L^*_{\max}$.*

1. *Create a set $A = \{\alpha_1, \alpha_2, \dots \alpha_n\}$, $\alpha_i \geq 0$ containing the α values for all the n points we want to use for interpolation.*
2. *For each α_i in A create a transformed instance A_{α_i} using the $r' = \alpha r$ transform. Obtain the $L_{\max}(\alpha_i)$ value for this instance.*
3. *Using Lagrange interpolation and the calculated objective function values - return the $L^*_{\max}$ value using the formula (5).*

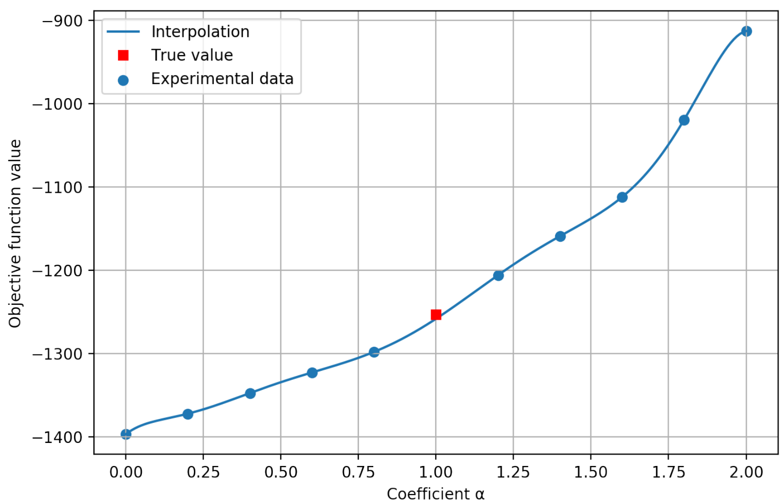

Figure 1. An illustration for the Algorithm 1. The round points are the interpolation nodes, the objective function value is known for each of them. Then the interpolating curve is plotted and the initial instance objective function value is estimated using this curve. The square point is the true value of the objective function so we can compare the true value with its approximation found by the interpolation curve.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

It would be highly effective, however, to use polynomially solvable instances as the interpolation nodes to be able to estimate the $L^*_{\max}$ value in polynomial time. For this purpose we have developed two polynomial classes of instances which can be easily used in the interpolation approach avoiding massive calculations.

These classes are called the "highly different r" polynomial subcase and "slightly different r" polynomial subcase.

6. The "highly different r" polynomial subcase

Definition 3. *An instance $A = \{j_1 \dots j_n\}$ is a case of "highly different r" if the following inequality is true for this instance:*

$$r_j - r_i \geq p_i, \text{ where } i, j = 1 \dots n,\ i \neq j, r_j > r_i. \tag{6}$$

To get an intuitive understanding of the situation described in the definition, let consider the following Gantt chart[12].

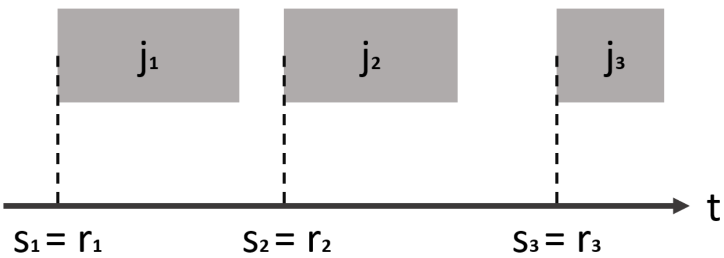

Figure 2. Gantt chart example for the "highly different r" case.

Each r_i, r_j are so far away from each other on the timeline, that the processor has enough time to complete the previous job before receiving the next one. So it is obvious that the optimal schedule π^* for this case is obtained by sorting the jobs by increasing receivement time order.

However, a strict proof of this fact is given below.

Lemma 1. *For an instance N of n jobs we will consider such schedule $\pi = j_1 \dots j_n$, for which the inequality $r_{j_1} < r_{j_2} < \dots < r_{j_n}$ is obtained. Then in the "highly different r" the following equality is true:*

$$r_j = s_j \ \forall j \in A. \tag{7}$$

Proof.

1. For the job j_1 the equality (7) $s_1 = r_1$ is true, because it is the first job in the schedule and so it will start being processed right after the receivement time.
2. May the equality (7) be true for the job j_i: $s_i = r_i$. Then for the job j_{i+1}: $s_{i+1} = \max(C_i, r_{i+1}) = \max(s_i + p_i, r_{i+1}) = \max(r_i + p_i, r_{i+1})$. According to the definition 3: $r_{i+1} - r_i \geq p_i$, which means that

$$r_{i+1} \geq r_i + p_i. \tag{8}$$

From (8) we can conclude that $\max(r_i + p_i, r_{i+1}) = r_{i+1}$. Then, $s_{i+1} = r_{i+1}$. The equality (7) is obtained and hereby the lemma is proven.

□

Theorem 1. *The optimal schedule $\pi^* = j_1 \dots j_n$ for the "highly different r" case is such schedule, in which the jobs are ordered by increasing release times: $r_{j_1} < r_{j_2} < \dots < r_{j_n}$.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

Proof. Let consider the job j_i on which the maximum lateness value is obtained: $L_{j_i}(\pi^*) = L_{\max}(\pi^*)$. Let suppose that a schedule π exists, for which $L_{\max}(\pi) < L_{\max}(\pi^*)$. This means that also $L_{\max}(\pi) < L_{j_i}(\pi^*)$.

By definition $L_{j_i}(\pi^*) = C_{j_i}(\pi^*) - d_{j_i} = s_{j_i}(\pi^*) + p_{j_i} - d_{j_i}$. Using Lemma 1 we obtain the following equality:

$$L_{j_i}(\pi^*) = s_{j_i}(\pi^*) + p_{j_i} - d_{j_i} = r_{j_i} + p_{j_i} - d_{j_i}.$$

As shown above for the schedule π: $L_{\max}(\pi) < L_{j_i}(\pi^*)$. It means that $L_{\max}(\pi) < r_{j_i} + p_{j_i} - d_{j_i}$. Then:

$$L_{j_i}(\pi) < r_{j_i} + p_{j_i} - d_{j_i}. \tag{9}$$

According to the definition 2, $L_{j_i}(\pi) = s_{j_i}(\pi) + p_{j_i} - d_{j_i}$. Then we obtain the following inequality for the equation (9): $s_{j_i}(\pi) < r_{j_i}$. Which is impossible According to the definition of the release time.

Therefore we came to a contradiction. Hence, there cannot exist a schedule π for which $L_{\max}(\pi) < L_{\max}(\pi^*)$. π^* is the optimal schedule.

□

7. The "slightly different r" polynomial subcase

Definition 4. *An instance $A = \{j_1 \dots j_n\}$ is a case of "slightly different r" if the following inequality is true for this instance:*

$$r_j - r_i \leq p_i, \text{ where } i, j = 1 \dots n,\ i \neq j, r_j > r_i. \tag{10}$$

Remark 1. *Let note that the inequality (10) is equivalent to the following one:*

$$r_j \leq p_i + r_i, \text{ where } i, j = 1 \dots n,\ i \neq j, r_j > r_i. \tag{11}$$

To get an intuitive understanding of the situation described in the definition 4, let consider the following Gantt chart.

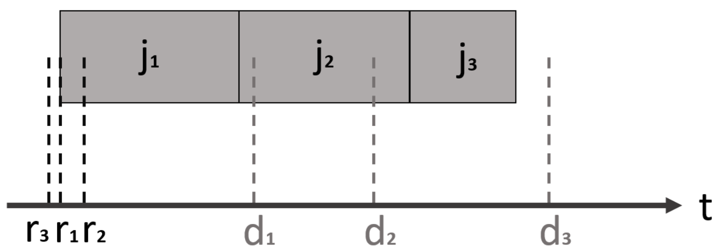

Figure 3. Gantt chart example for the "slightly different r" case.

In this case all release times are so near to each other on the time line, that all the jobs in the instance will have been recieved after completing the first job in the schedule.

Algorithm 2 (Solution of the "slightly different r" case)**.**

1. *Create n different schedules $\pi_1 \dots \pi_n$ using the following rule: $\pi_i = \{i, argsort(\vec{d}) \setminus i\}$, $i = 1 \dots n$ - the job number i is put first in the schedule π_i, all other jobs are sorted by non-decreasing due date.*
2. *Choose the index k of the schedule π_k on which the minimum objective function value is obtained: $k = argmin_{i=1 \dots n}(L_{\max}(\pi_i))$.*
3. *$\pi^* = \pi_k$ - return the optimal schedule.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

A strict proof that the schedule π^* obtained by the algorithm is optimal follows.

Lemma 2. *In the "slightly different r" case the following inequality is true for any schedule:*

$$s_{j_i} > r_{j_i},\ i = 2 \dots n. \tag{12}$$

Proof.

1. Let
   According to (11): $s_{j_2} = r_{j_1} + p_{j_1}$.
2. Assume the inequality (12) is true for the job j_i. Then for the job j_{i+1}: $s_{j_{i+1}} = max(C_{j_i}, r_{j_{i+1}}) = max(s_{j_i} + p_{j_i}, r_{j_{i+1}})$.
   According to (11) for the jobs j_i, j_{i+1}: $r_{j_i} + p_{j_i} > r_{j_{i+1}}$. And from the inequality (12) for the job j_i: $s_{j_i} + p_{j_i} > r_{j_i} + p_{j_i} > r_{j_{i+1}}$.
   Finally we obtain $s_{j_{i+1}} = max(s_{j_i} + p_{j_i}, r_{j_{i+1}}) = s_{j_i} + p_{j_i} > r_{j_{i+1}}$. And for the job j_{i+1} the following is true: $s_{j_{i+1}} > r_{j_{i+1}}$.

□

Lemma 3. *In the "slightly different r" case the following inequality is true for any schedule:*

$$C_{j_i}(\pi) = r_{j_1} + \sum_{k=1}^{i} p_{j_k}. \tag{13}$$

Proof.

$s_{j_{i+1}}(\pi) = max(C_{j_i}(\pi), r_{j_{i+1}})$.

According to (12): $s_{j_{i+1}}(\pi) > r_{j_{i+1}}$. Thus, $s_{j_{i+1}}(\pi) = C_{j_i}(\pi), i = 2 \dots n$. This equality will be used in the proof further.

1. $i = 2$: $C_{j_2}(\pi) = s_{j_2}(\pi) + p_{j_2} = C_{j_1}(\pi) + p_{j_2} = r_{j_1} + p_{j_1} + p_{j_2}$. The equality (13) is true.
2. Assume the inequality (13) is true for the job j_i. Then for the job j_{i+1}:
   $$C_{j_{i+1}}(\pi) = s_{j_{i+1}}(\pi) + p_{j_{i+1}} = C_{j_i}(\pi) + p_{j_{i+1}} = r_{j_1} + \sum_{k=1}^{j_i} p_k + p_{j_{i+1}}.$$
   For the job j_{i+1}: $C_{j_{i+1}}(\pi) = r_{j_1} + \sum_{k=1}^{j_{i+1}} p_k$.

□

Corollary 1. *In the "slightly different r" case the following inequality is true for any schedule:*

$$C_{j_i}(\pi) = C_{j_1} + \sum_{k=2}^{i} p_{j_k}. \tag{14}$$

Proof. According to the definition, $C_{j_1} = r_{j_1} + p_{j_1}$. Then $C_{j_i}(\pi) = r_{j_1} + \sum_{k=1}^{i} p_{j_k} = C_{j_1} + \sum_{k=2}^{i} p_{j_k}$. □

Notation 3. *The set $A_J(\pi) = A \setminus j_1$ is as a set of the jobs not placed on the first position in the current schedule π.*

Notation 4. *The value $L^J_{\max}(\pi) = \max L_{j_i}(\pi), i = 2 \dots n$ is the maximum lateness value of all the n elements of N except the job that comes first in the current schedule π.*

Theorem 2. *Algorithm 2 finds the optimal schedule for the "slightly different r" case.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

Proof.

1. According to the definition, $L_{\max}(\pi^*) = \max(L_{j_1}(\pi^*), L^J_{\max}(\pi^*))$. Let assume that a schedule π exists, for which $L_{\max}(\pi) < L_{\max}(\pi^*)$.
2. If $\max(L_{j_1}(\pi^*), L^J_{\max}(\pi^*)) = L^J_{\max}(\pi^*)$ then $L^J_{\max}(\pi) < L^J_{\max}(\pi^*)$.
   According to the equation 14, the function $L^J_{\max}(\pi^*) = \max_{i=2\to n}(C_{j_i}(\pi^*) - d_{j_i})$ is the objective function of Jackson polynomial instance [13] with $r = C_{j_i}(\pi^*)$. Because j_1 here is fixed, π^* is the schedule on which the minimum maximum lateness is achieved here as proven in [13].
3. If $\max(L_{j_1}(\pi^*), L^J_{\max}(\pi^*)) = L_{j_1}(\pi^*)$ then the inequality $L_{\max}(\pi) < L_{\max}(\pi^*)$ cannot be true because the algorithm puts each job on the first position in the schedule to obtain the minimum objective function value.

□

8. Estimating the α^* and α_* values

In this section we will find the α coefficient values that are to be used in the $r' = \alpha r$ transform to achieve each of the polynomial cases listed above.

Theorem 3. *For an arbitrary instance $A = \{\vec{r}, \vec{p}, \vec{d}\}$, $\vec{r} = asc(\vec{r})$ there exists a set of transformed instances $A_\alpha = \{\alpha\vec{r}, \vec{p}, \vec{d}\}$ which are the cases of "highly different r", if α satisfies the following inequality:*

$$\alpha \geq \max \frac{p_i}{r_j - r_i},\ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{15}$$

Proof. According to the definition, in the "highly different r" case the following inequality is true:

$$r_j - r_i \geq p_i,\ \ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{16}$$

Let consider the $r' = \alpha r$ transform.

$$\alpha(r_j - r_i) \geq p_i,\ \ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i,$$

$$\alpha \geq \frac{p_i}{r_j - r_i},\ \ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i.$$

For brevity we will denote ξ_i^j as $\xi_i^j = \frac{p_i}{r_j - r_i}$, then:

$$\alpha \geq \xi_i^j,\ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i\ \Rightarrow\ \alpha > \max_{i,j} \xi_i^j.$$

And we finally obtain:

$$\alpha \geq \max \frac{p_i}{r_j - r_i},\ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{17}$$

□

So the coefficient α, to achieve the "highly different r" case should lie in the following interval: $\alpha \in [\max \frac{p_i}{r_j - r_i}; +\infty),\ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i$.

Definition 5. *The minimum value of the coefficient α to achieve the "highly different r" case is denoted as α^* and calculated, according to the Theorem 3, as follows:*

$$\alpha^* = \max \frac{p_i}{r_j - r_i},\ i, j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{18}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

It can be concluded from the definition that $\alpha^* \geq 0$, because the numerator of the fraction there is non-negative and denominator is a positive value.

From the equation (18) the condition of existence of the "highly different r" case can also be easily concluded.

Corollary 2 (The condition of existence of the "highly different r" case)**.** *The "highly different r" case exists for the initial instance A (which means that the value α^* is defined) if the following condition is met:*

$$r_i \neq r_j \; \forall i, j = 1 \dots n, \; i \neq j. \tag{19}$$

What is more, a sufficient condition of the "highly different r" case can be stated as follows.

Theorem 4 (A sufficient condition of the "highly different r" case)**.** *If the α^* value satisfies the inequality: $\alpha^* \leq 1$ then the instance is already a case of "highly different r".*

Proof. According to the definition, $\alpha^*(r_j - r_i) = p_i, \; i, j = 1 \dots n, \; i \neq j, \; r_j > r_i$.

Then, if $\alpha^* \leq 1$:

$$(r_j - r_i) \geq p_i, \; i, j = 1 \dots n, \; i \neq j, \; r_j > r_i. \tag{20}$$

This means that the initial instance A is is already a case of "highly different r". □

Now we will proceed to proving the equivalent theorems for the "slightly different r" case.

Theorem 5. *For an arbitrary instance $A = \{\vec{r}, \vec{p}, \vec{d}\}$, $\vec{r} = asc(\vec{r})$ there exists a set of transformed instances $A_\alpha = \{\alpha\vec{r}, \vec{p}, \vec{d}\}$ which are the cases of "slightly different r" if α satisfies the following inequality:*

$$0 \leq \alpha \leq \min_{i,j} \frac{p_i}{r_j - r_i}, i, j = 1 \dots n, i \neq j, r_j > r_i. \tag{21}$$

Proof. According to the definition, the coefficient α should satisfy the following inequality:

$$\alpha(r_j - r_i) \leq p_i, i, j = 1 \dots n, i \neq j, r_j > r_i. \tag{22}$$

Which means that

$$\alpha \leq \frac{p_i}{r_j - r_i}, i, j = 1 \dots n, i \neq j, r_j > r_i.$$

For brevity we will denote ξ_i^j as $\xi_i^j = \frac{p_i}{r_j - r_i}$. Then we obtain:

$$\alpha \leq \xi_i^j, i, j = 1 \dots n, i \neq j, r_j > r_i. \tag{23}$$

For this inequality to true for any $i, j = 1 \dots n, i \neq j, r_j > r_i$, there is also the following requirement:

$$\alpha \leq \min_{i,j} \xi_i^j, i, j = 1 \dots n, i \neq j, r_j > r_i. \tag{24}$$

This means that

$$\alpha \leq \min_{i,j} \frac{p_i}{r_j - r_i}, i, j = 1 \dots n, i \neq j, r_j > r_i. \tag{25}$$

What is more, $p_i > 0$. Then,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

$$0 \leq \alpha \leq \min_{i,j} \frac{p_i}{r_j - r_i}, i,j = 1 \ldots n, i \neq j, r_j > r_i. \tag{26}$$

□

So the coefficient α to achieve the "highly different r" case should lie in the following interval: $\alpha \in [0, \min \frac{p_i}{r_j - r_i})$, $i,j = 1 \ldots n$, $i \neq j$, $r_j > r_i$.

Definition 6. *The maximum value of the coefficient α to achieve the "slightly different r" case is denoted as α_* and calculated, according to the theorem, as follows:*

$$\alpha_* = \min \frac{p_i}{r_j - r_i},\ i,j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{27}$$

From the equation (27) the condition of existence of the "slightly different r" case can be easily concluded.

Corollary 3 (The condition of existence of the "slightly different r" case)**.** *The "highly different r" case exists for the initial instance A (which means that the value α^* is defined) if the following condition is met:*

$$r_i \neq r_j\ \forall i,j = 1 \ldots n,\ i \neq j. \tag{28}$$

Theorem 6 (A sufficient condition of the "slightly different r" case)**.** *If the α_* value satisfies the inequality: $\alpha_* \geq 1$ than the instance is already a case of "highly different r".*

Proof. From the definition, $\alpha_*(r_j - r_i) = p_i$, $i,j = 1 \ldots n$, $i \neq j$, $r_j > r_i$.

Then, if $\alpha^* \geq 1$:

$$(r_j - r_i) \leq p_i,\ i,j = 1 \ldots n,\ i \neq j,\ r_j > r_i. \tag{29}$$

This means that the initial instance A is is already a case of "slightly different r". □

Remark 2. *It can also be shown that, for example, for Lazarev polynomial class of instances, the following inequality is obtained:*

$$\alpha \geq \frac{d_j - d_i - p_j + p_i}{r_j - r_i},\ i,j = 1 \ldots n,\ i \neq j,\ d_j > d_i. \tag{30}$$

However, because the conditions in this and the other polynomial cases are more complex and may require different transforms, in this paper only the "highly different r" and "slightly different r" cases are defined and considered.

9. The interpolation-based polynomial method of estimating the objective function value

Now, since we have defined the general interpolation method algorithm and also have found the coefficient intervals related to the polynomial cases, let provide the interpolation-based polynomial algorithm.

Algorithm 3.

1. *Calculate the values α^* (18) and α_* (27).*
2. *Choose k values (k is an arbitrary positive integer) $\alpha_1 \ldots \alpha_k$ on the interval $[0, \alpha_*]$ so that $\alpha_1 = 0$, $\alpha_k = \alpha_*$ and the points are equally spaced. Denote the interval between two nearest points as Δ.*
3. *Choose k values $\alpha_{k+1} \ldots \alpha_{2k}$ on the interval $[\alpha^*, \alpha^* + k\Delta]$ so that $\alpha_{k+1} = 0$, $\alpha_{2k} = \alpha^*$ and the points are equally spaced.*

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

4. *Calculate the values* $L_{\max}(\alpha_1) \ldots L_{\max}(\alpha_2 k)$.
5. *Estimate the optimal value of the objective function of the initial instances using the* $L_{\max}(\alpha_1) \ldots L_{\max}(\alpha_2 k)$ *values and the formula* (5):

$$L^*_{\max} = \sum_{n=1}^{k} \frac{\prod_{i \neq k} (1 - L_{\max}(\alpha_i))}{\prod_{j \neq k} (1 - L_{\max}(\alpha_j))} L_{\max}(\alpha_n). \tag{31}$$

Remark 3. *The values* $L_{\max}(\alpha_1) \ldots L_{\max}(\alpha_2 k)$ *are independent and so can be calculated parallelly.*

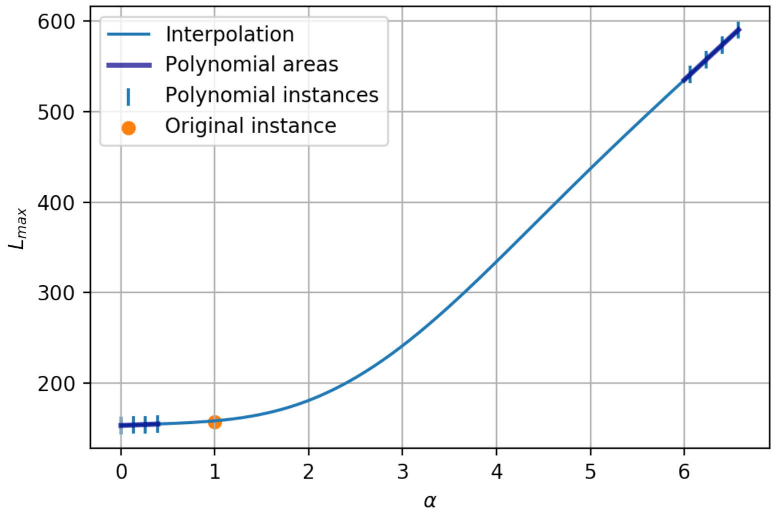

Figure 4. An illustration for the Algorithm 3. The thick dark segments are the polynomial areas. The vertical dashes are the polynomial interpolation nodes. The round point is the true value of the initial instance optimal objective function.

10. Numeric experiments

Before proceeding to the numerical experiments' results, here is some information on how these experiments have been carried on.

OS	Windows 10
CPU	Intel core i3
RAM	6Gb
Programming language	Python 3.7 [14]
Environment	Jupyter Notebook [14]
Main calculation library	numpy [15]
Graphic library	matplotlib/pyplot [16]
Random generation of r, p, d	Uniform integers $\in [0, 100]$

100 instances of size 8 have been generated. This same set of instances was used in all of the following numerical experiments to make it possible to compare different experiments' results.

The first experiment was conducted to calculate the optimal interpolation nodes number k. The results are presented on the following plot.

The nodes were selected according to the Algorithm 3, the parameter k was being changed.

The relative error value for each instance N was calculated using the following formula:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

$$\text{Err}_i = |\frac{L_i^T - L_i^*}{L_i^T}| \times 100\%, \tag{32}$$

where subscript i is the number of the instance in the set of 100 generated instances, L_i^T is the true optimal value of the initial instance objective function (obtained by the dual algorithm [10]) and L_i^* is the optimal value of the objective function estimated using the Algorithm 3.

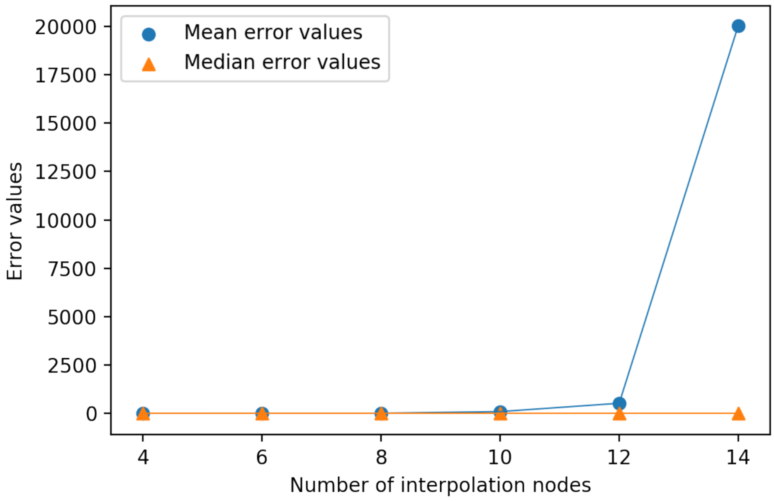

Figure 5. The plot shows the dependence of mean and median relative error values on the total number of the interpolation nodes.

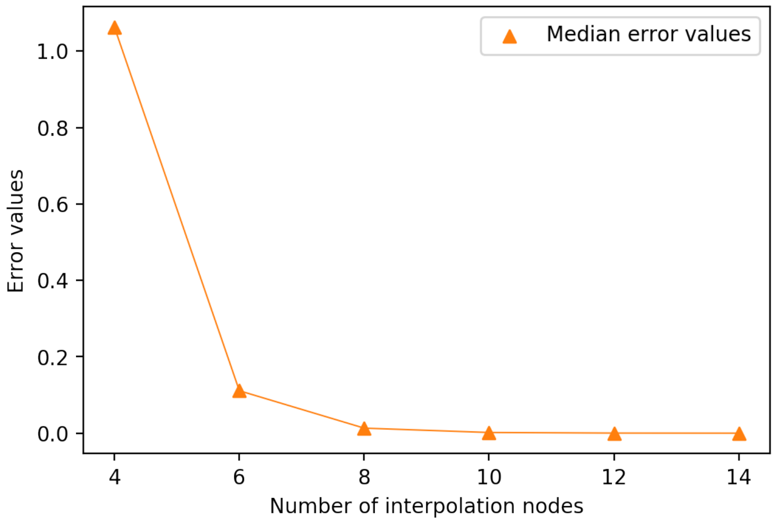

Figure 6. The plot shows the dependence of median relative error values on the total number of the interpolation nodes.

We can see that while median relative error decreases with the growth of the parameter k, the mean relative error increases. This means that although most of the instances are approximated more correctly, some instances become outliers with really high error values.

So to finally figure out the optimal number k, the following plot, showing the the dependence of the product of median and mean relative error values on the total number of the interpolation nodes, was created.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

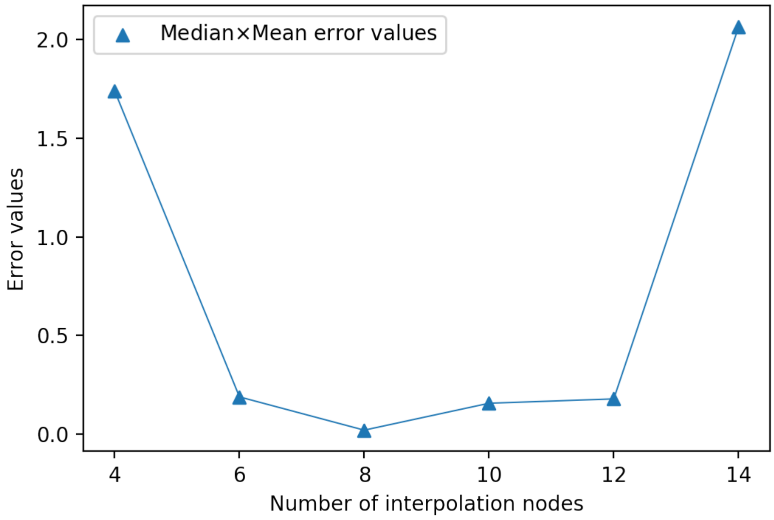

Figure 7. The plot shows the dependence of the product of median and mean relative error values on the total number of the interpolation nodes.

Now we can see from the graph that experimentally calculated optimal k value is $k = 8$.

The next experiment was conducted the following way. The parameter k value remained constant, but the distance Δ^* between each two neighboring points on the "highly different r" interval was increased in relation to the distance Δ_* between each two neighboring points on the "slightly different r" interval.

This can be done because, as shown above, "highly different r" interval has no higher bound on coefficient α.

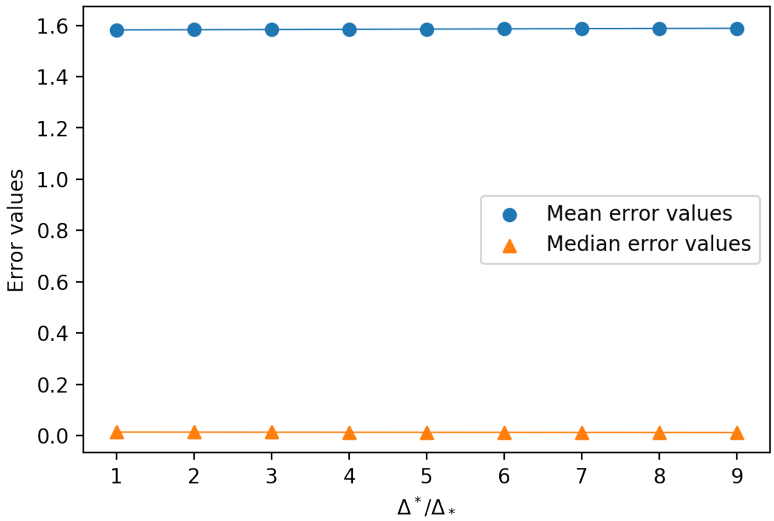

Figure 8. The plot shows the dependence of the median and mean relative error values on the step ratio $\frac{\Delta^*}{\Delta_*}$.

We can see that errors don't depend on the step ratio $\frac{\Delta^*}{\Delta_*}$, so we can just choose the steps to be equal: $\Delta^* = \Delta_* = \Delta$.

In the next experiment we have fixed the intervals $\Delta^* = \Delta_* = \Delta$ but were changing the number k^* of "highly different r" points. The results follow on the Figure 9.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

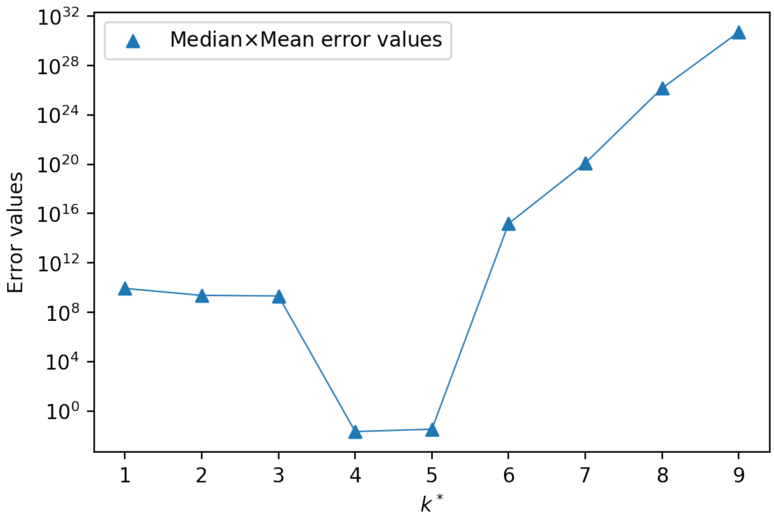

Figure 9. The plot shows the dependence of the product of median and mean relative error values on the number k^* of "highly different r" points.

The complexity[17] of the Algorithm 3 was evaluated as $O(n^p \log(n))$, where n is the number of jobs in the instance.

The resulting p value appeared to be $p \approx 2$, so the complexity can be estimated as $O(n^2 \log(n))$ (see Figure 10).

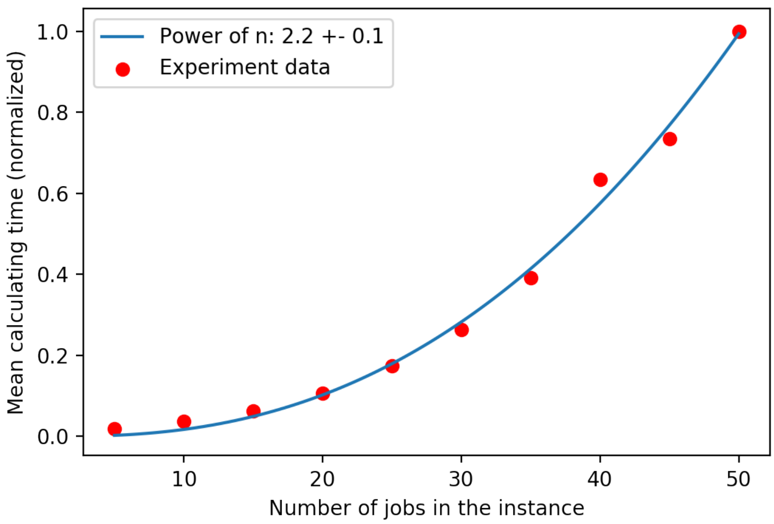

Figure 10. Complexity of the Algorithm 3.

11. Conclusion

In this paper a new approach to approximating the objective function value of the $1|r_j|L_{\max}$ problem is proposed.

The approach is based on the $L_{\max}(\alpha)$ function (using the $r' = \alpha r$ transform) and Lagrange interpolation.

The numeric experiments that have been carried out show how to optimize the hyperparameters of the method. The average complexity of the proposed algorithm is $O(n^2 \log(n))$, where n is the number of jobs in the instance.

 doi:10.20944/preprints202111.0169.v1

12. Further research

Further research into the features of the $L_{\max}(\alpha)$ will be conducted to develop a method of error estimation for the approach. The results will be compared with the results of error estimation of the metric approach[7].

There are also other transforms and polynomial cases that have to be studied. What is more, we are planning to study combinations of different transforms and their geometry in the 3n-dimensional feature space.

The Hypotheses stated in this paper will also be proven, so that we can boost the efficiency and the accuracy of the approach.

Different interpolation methods, including Chebyshev interpolation[20] and spline interpolation[21], can be used.

Also a combination of metric and interpolation approaches - the metric interpolation method - is being studied and developed.

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used "Conceptualization, Lazarev A.A., Lemtyuzhnikova D.V. and Tyunyatkin A.A.; methodology, Lazarev A.A., Lemtyuzhnikova D.V. and Tyunyatkin A.A.; software, Tyunyatkin A.A.; validation, Lazarev A.A., Lemtyuzhnikova D.V. and Tyunyatkin A.A.; formal analysis, Lazarev A.A., Lemtyuzhnikova D.V. and Tyunyatkin A.A.; Tyunyatkin A.A.; resources, Lazarev A.A., Lemtyuzhnikova D.V.; data curation, Lazarev A.A.; writing—original draft preparation, Tyunyatkin A.A.; writing—review and editing, Lazarev A.A., Lemtyuzhnikova D.V.; visualization, Tyunyatkin A.A.; supervision, Lazarev A.A.; project administration, Lemtyuzhnikova D.V.; funding acquisition, Lazarev A.A., Lemtyuzhnikova D.V.

Funding: This research was partially supported by RFBR grant number 20-58-S52006.

Data Availability Statement: The code to reproduce the experiments can be found in [18]. Many other experiments results, including numeric experiments with Chebyshev interpolation, are available in [19].

References

1. Lazarev, A.A. *Scheduling Theory: Methods and Algorithms*; ICS RAS: Moscow, Russia, 2019.
2. Brucker P. *Scheduling algorithms*; Berlin: Springer, 2006.
3. Sotskov Y.N., Werner F. (Editors) *Sequencing and Scheduling with Inaccurate Data.* New York, USA: Nova Science Publishers, Inc., 2014.
4. Tanaev V.S., Sotskov Y.N., Strusevich V.A. *Scheduling theory: multi-stage systems.* Dordrecht, The Netherlands: Kluwer Academic Publishers, 1994.
5. Davis, P.J. *Interpolation and approximation.* Courier Corporation, 1975.
6. Lenstra J.K.; Rinnooy Kan A.H.G.; Brucker P. *Complexity of machine scheduling problems*; Annals of discrete mathematics. Elsevier, 1977. V. 1. pp. 343–362.
7. Lazarev A.A.; Lemtyuzhnikova D.V.; Werner F. *A metric approach for scheduling problems with minimizing the maximum penalty*, Appl. Math. Modell. 2021. No. 89. pp. 1163–1176.
8. Kansizoglou I.; Bampis L.; Gasteratos A. *Deep Feature Space: A Geometrical Perspective*; IEEE Transactions on Pattern Analysis and Machine Intelligence. IEEE, 2021. pp. 1–1.
9. Efimov N.V.; Rozendorn E.R. *Linear algebra and multidimensional geometry*, Mir. 1975.
10. Lazarev A.A.; Pravdivets N.; Werner F. *On the Dual and Inverse Problems of Scheduling Jobs to Minimize the Maximum Penalty*, Mathematics, 8(7), p.1131. 2020.
11. Murty K.G. *Computational And Algorithmic Linear Algebra And N-Dimensional Geometry*. 2014.
12. Gantt H.L. *ASME Transactions*, 1903. 24. pp. 1322–1336.
13. Jackson J.R. *Scheduling a production line to minimize maximum tardiness*, Los Angeles, CA: University of California, 1955. Manag. Sci. Res. Project. Research Report N 43.
14. Kluyver T., Ragan-Kelley B., Pérez F., et.all. *Jupyter Notebooks – a publishing format for reproducible computational workflows.* Netherlands: InELPUB, 2016. P. 87–90.
15. Harris C.R., Millman K.J., van der Walt S.J., et al. *Array programming with NumPy.* Nature. 2020. V. 585. No. 7825. P. 357–362.
16. Hunter J.D. *Matplotlib: A 2D Graphics Environment.* Computing in Science & Engineering. 2007. V. 9. No. 3. P. 90–95.
17. Bovet D.P.; Crescenzi P.; Bovet D. *Introduction to the Theory of Complexity*, London: Prentice Hall. 1994.
18. Tyunyatkin A.A. *Schedule Research Repository*. Available online: https://github.com/andtun/schedules (accessed on 01.09.2021).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 November 2021 doi:10.20944/preprints202111.0169.v1

19. Tyunyatkin A.A. *Interpolation approach to minimizing the maximum lateness in scheduling problems*. Available online: https://tuniatk.in/schedules.html (accessed on 01.09.2021).
20. Mason, J.C., Handscomb, D.C. *Chebyshev polynomials.* CRC press, 2002.
21. Knott, G.D. *Interpolating cubic splines (Vol. 18)*. Springer Science & Business Media, 2000.