Model and Algorithms for Synthesis of Bi-Assignment

Yuriy Fedosenko[0000-0002-9434-4386], Dmitriy Khandurin[0000-0003-2485-525X], Anatoliy Sheyanov[0000-0003-3550-4068]

[1]Volga State University of Water Transport, Nesterova str., 5, Nizhny Novgorod, 603950, Russia

fds1707@mail.ru, [2]kaf_isut@vsuwt.ru, [3]asheyanov@ya.ru

Abstract. In discrete idealization, a mathematical model of the assignment of the pairs of non-interchangeable tasks between the agents is formulated. The model describes, in particular, the state of the water transport logistics system at the decision-making moment during planning the usage of a group of heterogeneous cargo ships. Within the framework of the formulated model, the concept of “bi-assignment” is introduced and optimization problem of bi-assignment with a minimax criterion is posed, which generalizes the classical assignment problem. For the task and its practically significant refinements, the intractability is proved. Solving algorithms implementing the concept of dynamic programming and the branch-and-bound scheme are constructed. Examples of the numerical implementation of the bi-assignment synthesis are given. The developed model and algorithm are implemented in the logistics management support system in the Kazan river port.

Keywords: assignment problem, bi-assignment, dynamic programming, branch-and-bound scheme, computational complexity

1 Introduction

The classical assignment problem (AP) (linear balanced assignment problem) – a problem of optimal, in one sense or another, assignment of the finite n-element collection of tasks between the same number of agents was first formulated in 1952 by Votaw, D. F., Jr. and Orden, A. in [1].

Subsequently, the AP served as the basis for setting various modifications and applied interpretations, including flexible production systems, determining the location of production facilities and so on.

A detailed analysis of the models and varieties of the assignment problem is presented, for example, in the overview chapters of [2, 3].

The development and research of algorithms for solving AP also have a long history since the early work of Kuhn H.W. [4] describing the method for solving the AP named by the author "Hungarian method".

This method and the method of potentials, as well as their modifications developed in recent years, are the main regular methods for synthesizing exact solutions of the AP [5, 6].

By these methods, the AP is solved in polynomial time of the size of the input. Among the numerous approaches developed for the synthesis of approximate solutions of the AP we note the algorithms developed on the basis of the heuristic concepts [7, 8].

Nowadays, due to the intensive development and integration in everyday practice of digital support management applications, in particular, in the operational planning of logistics and production and transportation processes [9], there is a need for advanced models for the distribution of discrete resources.

One of these extensions relates to the pair distribution of non-interchangeable resources (tasks) between the agents considered in the article.

As an example, we will point to the production and transport system [10], in which a selected group of non-identical river cargo ships is used for transporting sand and gravel mixture (SGM) to specified storage (consumption) points, loaded in a single technological cycle by floating hydro-mechanized mining complexes (HMC) on large-scale area of riverbed deposits. In such a system, up to 10 – 15 HMC units can be involved, and the same number of ships arrives daily for loading.

At the end of the development session of the next operational plan for the functioning of the system under consideration, it should be unambiguously determined:

- to which HMC unit from the number of riverbed deposits located at the area, each specific ship should be sent for loading SGM;
- which unloading destination should be assigned to each specific ship after loading the SGM.

For the automated formation of operational plans for the operating of the production and transportation system, effective in the conditions of a current operating environment, the computer control support complex should include both a digital modeling system module and tools for solving the corresponding optimization problem of ship allocation between the HMC for loading SGM and ships distribution at SGM unloading points.

As a second example, we will mention the problem of paired distribution of high-speed passenger ship destinations along routes for mass transportation in megacities, regional and island agglomerations.

The purpose of this article is to develop a mathematical model for the distribution of pairs of non-interchangeable tasks between the agents, formulate an optimization problem with a minimax criterion, construct decision-making algorithms acceptable for practical use, and consider the computational complexity of the problem and its special cases.

It is in this way that the material of the article is presented, which includes six sections, two appendixes and References.

In the next (second) section a mathematical model for the pair use of discrete resources of the type in question is formulated, and the general problem of bi-assignment with a minimax criterion is stated, generalizing AP with a minimax criterion (Linear bottleneck assignment problem).

This problem is denoted in the article as GBAP (generalized bi-assignment problem).

The third section of the article is devoted to the construction of a discrete dynamic programming algorithm [11, 12], which allows one to synthesize exact solutions of the GBAP posed in the second section; an estimate of its computational complexity is given here [13] and, as an illustration, the result of the implementation of the algorithm on the numerical data of the example is given.

In the fourth section, an algorithm for the synthesis of optimal and suboptimal bi-assignments in the process of iterative synthesis of the solution of GBAP according to the branch-and-bound scheme is constructed.

The fifth section is devoted to two practically significant special cases of the GBAP, including taking into account the laboriousness of tasks and the productivity of agents; it is established that all these problems are intractable [9], algorithms of polynomial computational complexity cannot be built for them.

The sixth section of the article is Conclusion

Appendix 1 and 2 contains step-by-step implementations of the constructed dynamic programming algorithms and branches and boundaries for the example of GBAP given in the third section.

2 Formal Problem Statement

Let there be a set of agents $\boldsymbol{I} = \{1, 2, \dots , n\}$ and two sets of tasks $\boldsymbol{P} = \{p_1, p_2, \dots , p_n\}$ and $\boldsymbol{Q} = \{q_1, q_2, \dots , q_n\}$. Each agent must be assigned to one of the tasks of the set $\boldsymbol{P}$ and one of the tasks of the set $\boldsymbol{Q}$. Each of the tasks must be completed in full by only one agent.

The $(n{\times}n)$-matrices $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ of numerical estimates are assumed to be given, where a_{ij} is the cost of execution of the task p_j by the agent i, b_{ij} is the cost of the execution of the task q_j by the same agent, $i = \overline{1,n}$, $j = \overline{1,n}$.

Let us introduce the following notation: $\boldsymbol{\pi}_1 = \{\pi_1(i),\ i = \overline{1,n}\ \}$ is the set of assignments of agents to tasks from the set $\boldsymbol{P}$, $\boldsymbol{\pi}_2 = \{\pi_2(i),\ i = \overline{1,n}\ \}$ is the set of assignments of agents to tasks from the set $\boldsymbol{Q}$.

Both the assignment $\boldsymbol{\pi}_1$ and the assignment $\boldsymbol{\pi}_2$ are a bijection of the set $\boldsymbol{I} = \{1, 2, \dots , n\}$ to itself (permutation).

The equality $\pi_1(i) = j$ means the appointment of agent i to task p_j. Similarly, the equality $\pi_2(i) = j$ means the appointment of agent i to task q_j.

We denote pairs of the form $< \pi_1(i), \pi_2(i) >$ as bi-assignment; it is supposed that when implementing bi-assignment $< \pi_1(i), \pi_2(i) >$, each agent i, first starting from time 0 performs task with the number $\pi_1(i)$, and then immediately proceeds to task with the number $\pi_2(i)$, $i = \overline{1,n}$.

In a general form, a GBAP with a minimax criterion is written as follows:

$$\min_{\pi_1, \pi_2} (\max_{\alpha}[a_{\alpha\pi_1(\alpha)} + b_{\alpha\pi_2(\alpha)}]) \tag{1}$$

If matrices A and B set the processing time of the task by the agents, then solving (1), we look for a bi-assignment that ensures the minimality of the total processing time of the entire complex of tasks $\{p_1, p_2, \dots , p_n, q_1, q_2, \dots , q_n\}$.

It is easy to see that (1) is a generalization of AP.

3 Construction of a Solving Algorithm Based on the Dynamic Programming Concept

Let i be a natural constant not exceeding n, W_1, W_2 be arbitrary i-element subsets of $\{1, 2, \dots , n\}$.

By $Z(i, W_1, W_2)$ we denote the subproblem of the problem (1), in which among the agents of the set $\boldsymbol{I}$ one should distribute tasks with lower indices (numbers) from the subsets W_1 and W_2; in this case, each agent should receive only one task from the set $\boldsymbol{P}$ with a subscript included in the subset W_1, and only one task from the set $\boldsymbol{Q}$ with the index included in the subset W_2. The optimal value of the criterion of the problem (1) is denoted by B_{opt}.

According to the concept of dynamic programming [12], the state of the process of the pair distribution of the tasks of the sets $\boldsymbol{P}$, $\boldsymbol{Q}$ between the agents of the set $\boldsymbol{I}$ at step i is uniquely determined by the triple (i, W_1, W_2), where i is a natural constant not exceeding n, restricting the assignment to agents with numbers 1, 2, ..., i, and W_1, W_2 □ arbitrary i -element subsets of the sets of indices of the tasks $\boldsymbol{P}$, $\boldsymbol{Q}$, respectively, available for distribution.

The optimal criterion value in the problem $Z(i, W_1, W_2)$ is denoted by $B(i, W_1, W_2)$. $B(i, W_1, W_2)$ is the Bellman function for problem (1), and

$$B(1, \{j\}, \{k\}) = a_{1j} + b_{1k};\quad j, k\in \boldsymbol{N}. \tag{2}$$

According to the optimality principle, for solving problem (1), we write the following recurrent relations of dynamic programming

$$B(i, W_1, W_2) = \min_{\alpha,\beta}(\max[(a_{i\alpha} + b_{i\beta}), B(i -1, \{W_1\backslash \alpha \}, W_2\backslash \beta \})]), \tag{3}$$

$$B_{opt}(n, \boldsymbol{I}, \boldsymbol{I}) = \min_{\alpha,\beta}(\max[(a_{n\alpha} + b_{n\beta}), B(n\text{-}1, (\boldsymbol{I} \backslash \alpha \}, \boldsymbol{I} \backslash \beta))]), \tag{4}$$

where (α, β) are arbitrary pairs of indices from the set $W_1\times W_2$.

The implementation of the computational algorithm along these relations, denoted below by DP, begins with the determination of the quantities $B(1, \{j\}, \{k\})$ for all singleton sets W_1 и W_2.

Next, sequentially in increasing order of parameter i for all possible sets W_1 and W_2, the values of the Bellman function $B(i, W_1, W_2)$, $i = \overline{1, n-1}$ are determined by formula (3).

The value $B(n, \boldsymbol{I}, \boldsymbol{I})$ determined by relation (4) is the optimal criterion value B_{opt} in problem (1).

In the process of executing the DP algorithm for each triple (i, W_1, W_2) of the values of the arguments of the Bellman function, we should fix the pair (α, β) on which the minimum of the right-hand side of relations (3), (4) is realized. This will allow, after finding the optimal value of the B_{opt} criterion, to uniquely determine the corresponding bi-assignment.

Relations (2) - (4) suppose implementation of the direct calculation scheme, without taking into account the state, unattainable from the initial one.

We specify below a phased implementation of the described synthesis algorithm for solving problem (1).

Stage 1. The initial triple (1, $\{j\}$, $\{k\}$) and empty lists G and G^* are initialized. By the formula (2), the values of the Bellman function $B(1, \{j\}, \{k\})$ are calculated at j, $k = 1, 2, \dots, n$. Goes to stage 2.

Stage 2. For values of i satisfying the inequality $i < n$, we calculate the values of the Bellman function $B(i, W_1, W_2)$ by the formula (3) for all possible subsets of the indices W_1 and W_2 of tasks from respectively the sets $\boldsymbol{P}$ and $\boldsymbol{Q}$ of dimension i; list G is updated with entries of the form (i, W_1, W_2, α^*, β^*), where α^*, β^* are the values of the indices α, β, at which the optimal value of the function $B(i, W_1, W_2)$, $i = 2, 3, \dots, n\text{-}1$ is reached. Otherwise, i.e. if $i = n$, go to stage 3.

Stage 3. By the formula (4), the value of the Bellman function $B_{opt}(n, \boldsymbol{I}, \boldsymbol{I})$ is calculated. The entry (n, $\boldsymbol{I}$, $\boldsymbol{I}$, α^*, β^*) is added to the list G; the loop counter z is set to n, the working arrays V_1, V_2 are initialized according to the relations $V_1 = \boldsymbol{I}$, $V_2 = \boldsymbol{I}$. Go to stage 4.

Stage 4. An element corresponding to the triple (z, V_1, V_2) is selected from the list G. The entry (z, α^*, β^*) is added to the list G^*; for $z = n, n\text{-}1, \dots, 2$, the operations $z = z\text{-}1$, $V_1 = V_1 \setminus \alpha^*$, $V_2 = V_2 \setminus \beta^*$ are performed.

Stage 5. Ending the DP algorithm: the list G^* contains the optimal solution to problem (1).

The complexity of the DP algorithm is determined by the number of calculated values of the Bellman function and is determined by the value $O(4^n)$.

As an illustration, we present the result of executing the DP algorithm on the numerical data of problem (1) with square matrices of the form

$$A = \begin{pmatrix} 70 & 35 & 10 & 68 \\ 72 & 68 & 69 & 12 \\ 42 & 62 & 8 & 96 \\ 50 & 60 & 98 & 84 \end{pmatrix}; \; B = \begin{pmatrix} 69 & 73 & 32 & 15 \\ 85 & 3 & 39 & 96 \\ 13 & 6 & 31 & 28 \\ 3 & 33 & 54 & 51 \end{pmatrix} \qquad (5)$$

The optimal value of the criterion $B_{opt}(n, \boldsymbol{I}, \boldsymbol{I}) = 53$ in the above example problem (5) is achieved with bi-assignment of the form

$$\pi_p(1) = 2,\ \pi_p(2) = 4,\ \pi_p(3) = 3,\ \pi_p(4) = 1, \qquad (6)$$

$$\pi_q(1) = 4,\ \pi_q(2) = 2,\ \pi_q(3) = 3,\ \pi_q(4) = 1.$$

Step-by-step execution of the DP algorithm is reproduced in tables 2 – 5 of Appendix 1.

4 Construction of a Solving Algorithm Based on the Branch-and-Bound Paradigm

The implementation of the branch-and-bound paradigm [14] consists in constructing a fragment of the variant tree sufficient to determine the optimal solution. The vertices of this tree correspond to states (i, π'_1, π'_2), where i is the number of assigned agents, π'_1, π'_2 are the subsets of the sets $\boldsymbol{P}$ and $\boldsymbol{Q}$ that determine the assignments of agents $\{1, 2, \dots, i\}$; in particular, the root of the variant tree corresponds to the triple $(0, \{ø\}, \{ø\})$.

The algorithm, hereinafter referred to as WG, for the implementation of the computational procedure according to the branch-and-bound scheme is completely determined by defining the methods for:

a) obtaining upper bounds U_B for the values of minimax criterion (1) at the constructed vertices of the variant tree (in minimization problems, the upper bound is provided by a feasible solution);

b) obtaining lower bounds for L_B at the constructed vertices of the variant tree;

c) branching regulating the order of traversal of vertices.

To obtain an upper bound of the minimax criterion (1) in the root of the variant tree, the following four possible options for paired assignments should be considered:

1. $\pi_1^*(i) = i,\ \pi_2^*(i) = i;$
2. $\pi_1^*(i) = i,\ \pi_2^*(i) = n - i + 1;$
3. $\pi_1^*(i) = n - i + 1,\ \pi_2^*(i) = i;$
4. $\pi_1^*(i) = n - i + 1,\ \pi_2^*(i) = n - i + 1$

and select a paired assignment with a minimum value of $\max_{\alpha}([a_{\alpha\pi_1(\alpha)} + b_{\alpha\pi_2(\alpha)}])$.

The bi-assignment $< \pi_1^*(i),\ \pi_2^*\ (i) > \pi^{**}$ thus formed provides an upper bound for U_B in the root of the variant tree.

It is easy to see that, as a lower bound of L_B at the root, we can take the quantity determined by the formula

$$\max_{\alpha}[\min_{\beta} a_{\alpha\beta} + \min_{\beta} b_{\alpha\beta}].$$

The above methods for obtaining bounds of U_B and L_B at the root induce procedures for finding the upper and lower bounds in the vertices obtained at all subsequent intermediate vertices of the variant tree.

The smallest of the U_B bounds obtained during the calculation is called the record *Rec*; initially, the record values are assumed to be $+\infty$, while the in WG algorithm execution process, the record value decreases.

The branching procedure consists in assigning the next agent to tasks from the sets $\boldsymbol{P}$ and $\boldsymbol{Q}$, respectively, and pruning out the subsets of feasible solutions that knowingly do not contain optimal solutions.

When branching for agent i, all tasks p_j and q_k are considered, for which previous agents of the current branch were not assigned. Thus, we obtain $(n - i + 1)\times(n - i + 1)$ new vertices.

Branching at the vertices in which the L_B bound exceeds the existing record value is not performed as impractical.

As an example, on the numerical data of problem (1) with square matrices (5), the execution of the WG algorithm up to obtaining the optimal bi-assignment (6) is reproduced step-by-step in tables 6–10 in Appendix 2. For a clear presentation of the solution process in these tables the operator $K(i, \pi'_1, \pi'_2)$ transforming the current (i, π'_1, π'_2) state is introduced.

The result of applying this operator is a pair of numerical values corresponding to the lower bound L_B and the upper bound U_B of the minimax criterion values (1) in the constructed vertex of the variant tree.

The root of the decision tree corresponds to the state $(0, \{\varnothing\}, \{\varnothing\})$; the result of applying the operator to this state according to the procedures described above is the pair {53, 107}, the current record *Rec* becomes 107.

To evaluate the performance of the software implementation of the WG algorithm in comparison with the performance of the software implementation of the DP algorithm, computational experiments were performed on a test data set.

For each dimension $n = 10 – 13$, a hundred problems were solved, determined by the square matrices of numerical estimates A and B of the corresponding dimension.

The integer values of the elements of the matrices a_{ij}, b_{ij} were generated in a pseudo-random manner according to the uniform distribution law on the interval [0, 99].

The duration of the test run of each algorithm was measured to the nearest second. The experimental results are shown in table 1.

Table 1. Results of the experiments

The value of dimension *n* of the problem (1)	The average execution time of the algorithm	
	DP	WG
10	37	8
11	174	30
12	912	476
13	4377	4314

In production and transport applications, in the presence of strict regulatory restrictions on the duration of solving problem (1), it is advisable to synthesize approximate solutions.

In this context, the principal advantage of the WG algorithm is the ability to terminate it as soon as the set time limit for the task has been exhausted. In this case, an estimate of the deviation of the obtained approximate value of the minimax criterion from its optimal value will be known.

5 Special Cases of the General Problem of Bi-Assignment, Estimates of Computational Complexity

As a concretization, we introduce GBAP$_{hw}$ in which each of the available $2n$ tasks is characterized by the labor consumption of h, so task p_j has labor consumption of $h(p_j)$, task q_j – $h(q_j)$; each agent i is characterized by performance w_i. With that, the elements of the matrices of numerical estimates A and B are calculated by the formulas

$$a_{ij} = h(p_j) / w_i , b_{ij} = h(q_j) / w_i, i = \overline{1,n} , j = \overline{1,n} .$$

Under these conditions, for GBAP and GBAP$_{hw}$ problems, the following recognition problems corresponding to them can be distinguished.

Problem 1: with the initial GBAP data and the additional constant T, it is asked if there is a bi-assignment, implementation of which executes the entire set of available tasks no later than the time instant T.

Problem 2: with the initial data GBAP$_{hw}$ and additionally indicated constant T, the restriction is established by analogy with problem 1.

The computational complexity of GBAP is no less than the computational complexity of GBAP$_{hw}$, Problem 1 and Problem 2; the computational complexity of GBAP$_{hw}$ and Problem 1 is no less than the computational complexity of Problem 2.

It is easily shown that Problem 2 in polynomial time can be reduced to the problem "matching with weight restrictions," which, in turn, is NP-complete [13].

Thus, the intractability of both the general problem of bi-assignment and the particular modifications considered in this section is established; according to the accepted hypothesis about the distinction of the classes of problems P and NP, it is not possible to construct decisive algorithms of polynomial computational complexity for these problems.

6 Conclusion

The article proposes a mathematical model of the assignment between the agents of pairs of non-interchangeable tasks. The model describes, in particular, the state of the production and transport system at the decision-making moment when planning the usage of a group of heterogeneous cargo ships.

In the framework of the formulated model, the concept of "bi-assignment" is introduced and bi-assignment problem with a minimax criterion, which generalizes the classical assignment problem, is posed. The intractability of the task and its practically meaningful special cases is proved.

Solving algorithms, implementing concepts of dynamic programming and branches and bounds, are constructed.

As an example, the result of a phased implementation of bi-assignment synthesis for a problem with square matrices numerical estimates of fourth-order is given.

For practically significant for industrial transport applications values of dimension $n = 10 – 13$, the results of massive computational experiments on a comparative evaluation of the performance of developed algorithms are presented.

The results obtained demonstrate the importance of the developed algorithms for use, for example, in digital systems of support planning of the distribution of high-speed passenger ships along routes for mass transportation in megacities, regional and island agglomerations.

Prospects for further research are to modify the proposed model for a wider range of applied problems, including those requiring multi-criteria formulation [15].

In terms of mathematical models and the general problem of bi-assignment, characterized by increased dimensions and, accordingly, requiring practically unacceptable calculation time for their implementation, it is advisable to focus on the development of algorithms based on metaheuristic concepts [16], as well as oriented to supercomputer technologies for the implementation of computational processes [17, 18].

Acknowledgments. This paper has been prepared as a result of studies financially supported by the Russian Foundation for Basic Research, project no 15-07-03141.

Appendix 1

Table 2. Values of $B(1, \{j\}, \{k\})$

$j \setminus k$	1	2	3	4
1	139	143	102	85
2	104	108	67	50
3	79	83	42	23
4	137	141	100	83

Table 3. Values of $B(2, W_1, W_2)$ for subsets W_1, W_2 of dimension 2

$W_1 \setminus W_2$	{1, 2}	{1, 3}	{1, 4}	{2, 3}	{2, 4}	{3, 4}
{1, 2}	104	111	153	75	75	107
{1, 3}	79	111	154	75	75	108
{1, 4}	137	102	97	100	83	85
{2, 3}	79	107	153	71	71	107
{2, 4}	104	97	97	67	50	51
{3, 4}	79	79	97	42	25	52

Table 4. Values of $B(3, W_1, W_2)$ for subsets W_1, W_2 of dimension 3

$W_1 \setminus W_2$	{1, 2, 3}	{1, 2, 4}	{1, 3, 4}	{2, 3, 4}
{1, 2, 3}	71	71	197	71
{1, 2, 4}	67	50	51	70
{1, 3, 4}	43	43	51	70
{2, 3, 4}	63	50	51	50

Table 5. Values of $B(4, W_1, W_2)$ for subsets W_1, W_2 of dimension 4

$W_1 \setminus W_2$	{1, 2, 3, 4}
{1, 2, 3, 4}	**53**

Cyclically performing selections of items from the list G and adding to the list G^* according to Stage 4 of the DP algorithm, we'll obtain the solution to problem (1) in the form of bi-assignment (6) corresponding to $B_{opt}(n, \boldsymbol{I}, \boldsymbol{I}) = 53$.

Appendix 2

Table 6. The results of operator $K(i, \pi'_1, \pi'_2)$ on the current state 1.

(i, π'_1, π'_2)	$K(i, \pi'_1, \pi'_2)$	
	L_B	U_B
1, {1}, {1}	139	139
1, {1}, {2}	143	143
1, {1}, {3}	102	111
2, {1, 2}, {3, 1}	153	153
2, {1, 2}, {3, 2}	**102**	**102**

Rec = **102**

Table 7. The results of operator $K(i, \pi'_1, \pi'_2)$ on the current state 2.

(i, π'_1, π'_2)	$K(i, \pi'_1, \pi'_2)$	
	L_B	U_B
2, {1, 2}, {3, 4}	164	164
2, {1, 3}, {3, 1}	154	154
2, {1, 3}, {3, 2}	102	102
2, {1, 3}, {3, 4}	165	165
2, {1, 4}, {3, 1}	102	111
2, {1, 4}, {3, 2}	102	102
2, {1, 4}, {3, 4}	108	108
1, {1}, {4}	85	107
2, {1, 2}, {4, 1}	153	153
2, {1, 2}, {4, 2}	**87**	**87**

Rec = **87**

Table 8. The results of operator $K(i,\ \pi'_1,\ \pi'_2)$ on the current state 3.

$(i,\ \pi'_1,\ \pi'_2)$	$K(i,\ \pi'_1,\ \pi'_2)$	
	L_B	U_B
2, {1, 2}, {4, 3}	107	107
2, {1, 3}, {4, 1}	154	154
2, {1, 3}, {4, 2}	85	93
3, {1, 3, 2}, {4, 2, 1}	138	138
3, {1, 3, 2}, {4, 2, 3}	93	93
3, {1, 3, 4}, {4, 2, 1}	114	114
3, {1, 3, 4}, {4, 2, 3}	127	127
2, {1, 3}, {4, 3}	108	108
2, {1, 4}, {4, 1}	97	114
2, {1, 4}, {4, 2}	85	101
3, {1, 4, 2}, {4, 2, 1}	152	152
3, {1, 4, 2}, {4, 2, 3}	97	114
2, {1, 4}, {4, 2}	85	101
3, {1, 4, 2}, {4, 2, 1}	152	152
3, {1, 4, 2}, {4, 2, 3}	101	101
3, {1, 4, 3}, {4, 2, 1}	114	114
3, {1, 4, 3}, {4, 2, 3}	**85**	**85**

Rec = **85**

Table 9. The results of operator $K(i,\ \pi'_1,\ \pi'_2)$ on the current state 4.

$(i,\ \pi'_1,\ \pi'_2)$	$K(i,\ \pi'_1,\ \pi'_2)$	
	L_B	U_B
2, {1, 4}, {4, 3}	85	93
1, {2}, {1}	104	104
1, {2}, {2}	108	108
1, {2}, {3}	67	101
2, {2, 1}, {3, 1}	157	157
2, {2, 1}, {3, 2}	87	87
2, {2, 1}, {3, 4}	168	168
2, {2, 3}, {3, 1}	154	154
2, {2, 3}, {3, 2}	72	87
3, {2, 3, 1}, {3, 2, 1}	135	135
3, {2, 3, 1}, {3, 2, 4}	87	87

3, {2, 3, 4}, {3, 2, 1}	101	101
3, {2, 3, 4}, {3, 2, 4}	124	124
2, {2, 3}, {3, 4}	165	165
2, {2, 4}, {3, 1}	97	101
2, {2, 4}, {3, 2}	67	101
3, {2, 4, 1}, {3, 2, 1}	149	149
3, {2, 4, 1}, {3, 2, 4}	101	101
3, {2, 4, 3}, {3, 2, 1}	101	101
3, {2, 4, 3}, {3, 2, 4}	**67**	**67**

Rec = **67**

Table 10. The results of operator $K(i,\ \pi'_1,\ \pi'_2)$ on the current state 5.

$(i,\ \pi'_1,\ \pi'_2)$	$\mathrm{K}(i,\ \pi'_1,\ \pi'_2)$	
	L_B	U_B
2, {2, 4}, {3, 4}	108	108
1, {2}, {4}	53	104
2, {2, 1}, {4, 1}	157	157
2, {2, 1}, {4, 2}	87	87
2, {2, 1}, {4, 3}	111	111
2, {2, 3}, {4, 1}	154	154
2, {2, 3}, {4, 2}	72	87
2, {2, 3}, {4, 3}	108	108
2, {2, 4}, {4, 1}	97	104
2, {2, 4}, {4, 2}	53	101
3, {2, 4, 1}, {4, 2, 1}	152	152
2, {2, 3}, {4, 3}	108	108
2, {2, 4}, {4, 1}	97	104
2, {2, 4}, {4, 2}	53	101
3, {2, 4, 1}, {4, 2, 1}	152	152
3, {2, 4, 1}, {4, 2, 3}	101	101
3, {2, 4, 3}, {4, 2, 1}	104	104
3, {2, 4, 3}, {4, 2, 3}	**53**	**53**

Rec = **53**

Thus, the optimal value of criterion (1) equal to 53 is achieved on bi-assignment (6).

References

1. Votaw, D. F., Jr., Orden, A.: The personnel assignment problem. In Symposium on Linear Inequalities and Programming. Scientific Computation of Optimum Programs. Project SCOOP, Washington, D.C., no. 10, pp. 155-163 (1952).
2. Medvedeva, O. A.: Modeli i algoritmy resheniya mnogokriterial'nyh zadach o naznacheniyah s dopolnitel'nymi ogranicheniyami. https://www.dissercat.com/content/modeli-i-algoritmy-resheniya-mnogokriterialnykh-zadach-o-naznacheniyakh-s-dopolnitelnymi-ogr/read (date of request 01.08.2020).
3. Medvedev, S. N.: Obobshchyonnye modeli zadachi o naznacheniyah i adaptivnye algoritmy ih resheniya. https://www.dissercat.com/content/obobshchennye-modeli-zadachi-o-naznacheniyakh-i-adaptivnye-algoritmy-ikh-resheniya (date of request 01.08.2020).
4. Kuhn, H. W.: The Hungarian Method for the assignment problem. Naval Research Logistics Quarterly, vol. 2, pp. 83-97 (1955).
5. Morales, D. R., Romeijn, H. E.: The Generalized Assignment Problem and Extensions. Handbook of Combinatorial Optimization. Supplement vol. B. Springer, pp. 259-312 (2005).
6. Pentico, D.: Assignment problems: A golden anniversary survey. European Journal of Operational Research, no. 176, pp. 774-793 (2007). doi:10.1016/j.ejor.2005.09.014
7. Gonzalez T.F. (ed.): Handbook of Approximation Algorithms and Metaheuristics: Methologies and Traditional Applications, vol. 1. N.Y. Chapman and Hall/CRC (2018). doi:https://doi.org/10.1201/9781351236423
8. Medvedev, S. N., Medvedeva, O. A.: An adaptive algorithm for solving the axial three-index assignment problem. Automation and Remote Control, vol. 80, pp. 718-732 (2019). doi: 10.1134/S000511791904009X
9. Kogan, D. I., Fedosenko, Yu. S., Handurin, D. A.: Problem definition and solving algorithms for assignment problem applied to task of vehicle trains assembling. Bulletin of the Volga State Academy of Water Transport, no. 52. VSUWT publishing house, N. Novgorod, pp. 23-30 (2017).
10. Kogan, D. I., Trukhina, M. A., Fedosenko, Yu. S., Sheyanov, A. V.: Models and optimization problems for single-processor servicing of packets of objects. Automation and Remote Control, vol. 77, no 11, pp. 994-2005 (2016). doi:10.1134/S0005117916110096
11. Aris, R.: Discrete dynamic programming: an introduction to the optimization of staged processes. Blaisdell Pub. Co., Waltham, MA (1964).
12. Dreyfus, S. E., Bellman, R. E.: Applied Dynamic Programming. Princeton Univ. Press, Princeton, N.J. (1971).
13. Garey, M. R., Johnson, D. S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Co., New York (2003).
14. Land, H., Doig, A. G.: An automatic method of solving discrete programming problems. Econometrica, vol. 28, no 3, pp. 497-520 (1960) doi: 10.2307/1910129
15. Kogan, D. I., Fedosenko, Yu. S.: Handurin, D. A.: Concepts and algorithms for solving multi-criteria modifications of the assignment problem. Bulletin of the

Volga State Academy of Water Transport, no. 53. VSUWT publishing house, N. Novgorod, pp. 25-31 (2017).
16. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys, no 35(3), pp. 268-308 (2003). https://doi.org/10.1145/937503.937505
17. Cuencaa, J., Gimnezb, D., Martnez, J.: Heuristics for work distribution of a homogeneous parallel dynamic programming scheme on heterogeneous systems. Parallel Comput. 31(7), pp. 711-735 (2005).
doi:https://doi.org/10.1016/j.parco.2005.04.005
18. Fedosenko Yu., Reznikov M.: A Model of FPGA Massively Parallel Calculations for Hard Problem of Scheduling in Transportation Systems. In: Battiti, R. et al. (Eds): Learning and Intelligent Optimization. LION 2017, LNCS, vol. 10556, pp. 370-375 (2017). doi:https://doi.org/10.1007/978-3-319-69404-7_33