Modeling of the process of resource allocation between digital software technologies

Irina Zaitseva[1*], *Oleg* Malafeyev[2], *Shirin* AL Manai[2], *Svetlana* Belyaeva[3], *Marina* Leshcheva[4] and *Andrei* Murashko[2]

[1]Russian State Hydrometeorological University, 79 Voronezhskaya street, Saint-Petersburg, 192007, Russia
[2]Saint-Petersburg State University, 7/9 Universitetskaya nab., Saint-Petersburg, 199034, Russia
[3]Branch "Plekhanov Russian University of Economics" in Pyatigorsk, 8 Kuchura Street, Pyatigorsk, 357500, Russia
[4]Stavropol State Agrarian University, 12 Zootechnical lane, Stavropol, 355017, Russia

Abstract. Resource distribution amongst digital software technology systems is a tough operation that demands high effectiveness and a significant amount of time, especially because it is a dynamic activity with basic factors that vary over time. In this article, we highlight three different situations in the technology sector. Firstly, the role of using the dynamic programming approach to construct a model that helps find the maximum profit value for digital software technology subsystems by choosing the optimal sequence of control distribution of resources between these subsystems. Secondly, constructing the optimal order for digital software technology project execution to find minimal waiting time using the compromise solution algorithm. Thirdly, the role of the optimal assignment method in finding a compromise solution is that it determines the "fairest" strategy for assignment in the sense that it minimizes the loss to the owner, who has the maximum loss in profit. The assignment methodology finds the optimal software teams for digital software technology system projects, where each software team is qualified for a digital software technology system project, taking other participants' choices into account, to achieve the maximum profit. The dynamic programming methods considered here can be useful for planning and allocating resources between the enterprises of the agro-industrial complex.

1 Introduction

The scientist, Richard Bellman, is given credit for laying the intellectual foundation for dynamic programming between 1940 and 1950 in the Rand Corporation working group [1], where three conferences were held. The first Symposium, held in Chicago in 1949, and the second in Washington in 1955, were tasked with discussing distributed research on linear programming and other operations research theories. The third conference, also known as the Symposium, was held in California in 1959 and was more concerned with nonlinear

* Corresponding author: irina.zaitseva.stv@yandex.ru

programming and dynamic programming, which were two modern methods for dealing with more unsolved problems at the time [2]. Bellman's research was not published until 1957 in his book Dynamic Programming, and this book included applications of dynamic programming only in engineering fields. But he supplemented this book with two other books, one in 1961 and the other jointly with Dreyfus in 1962, in which he laid the theoretical and practical basis for dynamic programming in economic fields [3].

One of the most important properties of dynamic programming is that it assumes, for the purposes of analysis, that the relationships between economic variables are nonlinear relationships. The dynamic programming technique, on the other hand, allows for greater flexibility in economic analysis and planning since economic policy decisions may be adjusted to remedy any deviations that emerge during implementation. As a result, dynamic programming is an important mathematical method in the formation of sequential decisions, as the general principle on which it is based is dividing a large problem into several small problems, each of which has an optimal solution, and then evaluating all of the optimal solutions for these small problems to yield the optimal choice. Dynamic programming has been used in all scientific fields, including engineering, physics, chemistry, economics, and other sciences. Thus, in the article, we used the dynamic programming approach to construct the first model that helps find the maximum profit value for digital software technology subsystems by choosing the optimal sequence of control distribution of resources between these subsystems [4].

Based on that, digital software technology systems management is the art of directing and coordinating human and material resources throughout the digital software technology life cycle of a project by applying modern management methods and techniques to achieve project-defined results in terms of scope, cost, time, quality, and the satisfaction of project participants. The main goal of the second algorithm of a project is defined as meeting the project objectives while ensuring compliance with the established constraints on the duration and timing by finding the optimal order for digital software technology project execution for minimal waiting time [5].

The third model introduce the assignment problem, which was first formulated by Monge [6] in the eighteenth century (1784) as a massive assignment problem that minimizes the cost of transporting all the molecules by determining the minimum distance for the total transportation distance moving between two areas using an interesting geometric method. Dénes Kőnig [7] discovered in 1915 that Frobeniuś theorem published in 1912 [6] can be equivalently formulated in terms of graphs, and based on that, he gave the theorem that each regular bipartite graph has a perfect matching [8].

In 1931, E. Egerváry found a weighted version of Kőnig's theorem, which describes the maximum weight of matching in a bipartite graph, and thus applies to the assignment problem. Dantzig [9] presented the simplex method in 1947, which showed that the assignment problem can be formulated as a linear programming problem that automatically has an integer optimum solution. Halmos and Vaughan [10] presented the marriage problem in 1949.

In 1955, Harold W. Kuhn [11], presented a modern polynomial time algorithm for the assignment problem called the "Hungarian method" was derived from two important articles for two Hungarian mathematicians by Kőnig (1916) and Egerváry (1931), by translating into English the article published (in Hungarian) for Jenő Egerváry [12] in 1931 mentioned in Kőnig's book.

As a result, several researchers attempted to expand and investigate the assignment problem (e.g., Kuhn [13], Schrijver [14], Orden [15], and others). Here, it should be noted that most of the research has been able to prove to us that the pivot point in the development of combinatorial optimization is linear programming (Kőnig [7], Egerváry [12]).

In this article, we pointed to the role of the assignment problem methodology in the practical work of the technological sector or even in several aspects of our lives. By finding the optimal software teams for software technology system projects, where each software team is qualified for a software technology system project but unqualified for others, the aim is to determine whether it is possible to assign all software teams to software technology system projects, meaning that exactly one software technology system project is assigned to each software team, and to achieve the maximum profit for owners in the company. Then, working on the compromise solution algorithm, which determines the "fairest" assignment strategy in the sense that it minimizes the loss to the owner, who has the maximum loss in profit for all owners, that depends on the assignments of software teams to the software technology system projects. Thus, the goal of each software team group for owners, in addition to maximizing their profit, is to reach a compromise with the other participants. As a result, it is possible that not all participants in the company will achieve their maximum possible profit. The dynamic programming methods considered here can be useful for planning and allocating resources between the enterprises of the agro-industrial complex. Besides these methods can be applied for the development of effective methods of crop rotation planning.

2 Dynamic programming approach to the problem of resource allocation between digital software technology subsystems

2.1 Resource allocation model of digital software technology subsystems

We are planning the development of digital software technology subsystems $A_1,..,A_k$ over the period of time T that consists of m years. Let us at the beginning of the year i the resources $(C_i^1, C_i^2, \dots, C_i^k) = U_i$ are invested to the digital software technology subsystems $A_1,..,A_k$. The control $U_i = (C_i^1, C_i^2, \dots, C_i^k)$ is the set of steps controls $U = (U_1,..,U_m)$. Control efficiency of U is estimated by $W = W(U) = W(U_1, U_2, \dots U_m)$.

We aim to maximize W by choosing $U_1, U_2, \dots, U_m$. As a result, we denoted the optimal control as $U^* = (U_1^*, U_2^*, \dots., U_m^*)$. Here

$$W = \sum_{i=1}^{m} W_i \tag{1}$$

where W_i - is profit from the digital software technology subsystems over the year i. We are aiming to find U^* under which we have got

$$W_{\max} = \max_{U} W(U) . \tag{2}$$

If the state of the digital software technology subsystems $A_1,..,A_k$ is S, then let $W_i(S)$- is a profit beginning from i until the end. We denote $U_i^*(S)$, $i = 1, \dots, m$ - an optimal control in the process of resources distributing . So, we have the problem of finding $W_i(S)$and $U_i^*(S)$ for all steps. Let us have a step i . Let be in the result of the previous (i-1) steps the digital software technology subsystem has come to the state S and we have chosen U_i for all the steps i. If we apply a control U_i then we get the profit $\overline{W_i} = \overline{W_i}(S, U_i)$. The digital software technology subsystem is transformed from the state S to the state S'. It depends on S and U_i. Let us denote the rule of transitions T_i , it means that $S' = T_i(S, U_i)$. So, we have the equation

$$W_i(S) = \max_{U_i} \left\{ \overline{W}(S, U_i) + W_{i+1}(T_i(S, U_i)) \right\} \tag{3}$$

Here, we take $\overline{W}(S,U_i)$ and $T_i(S,U_i)$ as known, unknown functions are $W_i(S)$ and $W_{i+1}(S)$. It is clear that

$$W_m(S) = \max_{U_m} \left\{\overline{W}_m(S,U_m)\right\} \tag{4}$$

2.2 Examples on the process of the resource allocation model of digital software technology subsystems

The example is a set of digital software technology subsystems $A_1,..,A_{40}$. Now, based on our equation, we will find the optimal sequence of controls to gain the maximum profit. In figure 1 we show of digital software technology subsystem distribution, and in Table 1 we give the value of outcomes for this distributions. From the figure 1 for digital software technology subsystems distribution in the first transition T_1 we have a set of three subsystems, as follows

$$A_1 = \{A_2, A_3, A_4\}$$

The second transition T_2we have another sets of subsystems distribution, as follows,

$$A_2 = \{A_5, A_6, A_7\},$$
$$A_3 = \{A_8, A_9, A_{10}\},$$
$$A_4 = \{A_{11}, A_{12}, A_{13}\}.$$

The third transition T_3 sets of subsystems distribution , as follows,

$$A_5 = \{A_{14}, A_{15}, A_{16}\},$$
$$A_6 = \{A_{17}, A_{18}, A_{19}\},$$
$$A_7 = \{A_{20}, A_{21}, A_{22}\},$$
$$A_8 = \{A_{23}, A_{24}, A_{25}\},$$
$$A_9 = \{A_{26}, A_{27}, A_{28}\},$$
$$A_{10} = \{A_{29}, A_{30}, A_{31}\}$$
$$A_{11} = \{A_{32}, A_{33}, A_{34}\},$$
$$A_{12} = \{A_{35}, A_{36}, A_{37}\},$$
$$A_{13} = \{A_{38}, A_{39}, A_{40}\}.$$

So, the task is to find the maximum profit value for digital software technology subsystems by choosing the optimal sequence of resources distribution between these subsystems for each transition from T_1 , T_2 and T_3, when the values of profits for this distribution are shown in Table (1).

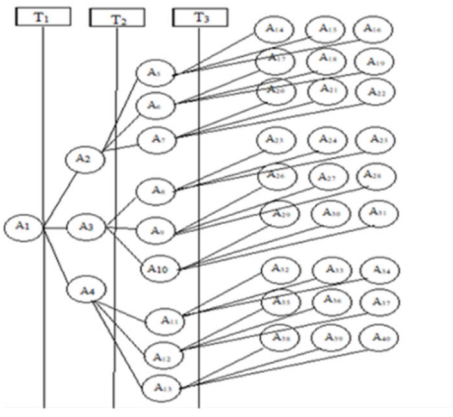

Fig. 1. Digital software technology subsystems distribution.

Table 1. Outcomes of digital technology subsystems.

T	Control	Innovative Technology	Outcomes
T1	u1	A1- A2	200.000
	u2	A1- A3	100.000
	u3	A1- A4	50.000
T2	u1	A2- A5	200.000
	u2	A2- A6	150.000
	u3	A2- A7	100.000
	u4	A3- A8	200.000
	u5	A3-A9	310.000
	u6	A3- A10	300.000
	u7	A4- A11	350.000
	u8	A4- A12	400.000
	u9	A4- A13	410.000
T3	u1	A5- A14	40.000
	u2	A5- A15	120.000
	u3	A5- A16	30.000
	u4	A6- A17	100.000
	u5	A6- A18	150.000
	u6	A6- A19	220.000
	u7	A7- A20	470.000
	u8	A7- A21	500.000
	u9	A7- A22	210.000
	u10	A8- A23	350.000
	u11	A8- A24	200.000
	u12	A8- A25	100.000
	u13	A9- A26	480.000
	u14	A9- A27	550.000
	u15	A9- A28	600.000
	u16	A10- A29	110.000
	u17	A10- A30	450.000
	u18	A10- A31	230.000
	u19	A11- A32	300.000
	u20	A11- A33	260.000
	u21	A11- A34	350.000
	u22	A12- A35	450.000
	u23	A12- A36	400.000
	u24	A12- A37	540.000
	u25	A13- A38	200.000
	u26	A13- A39	230.000
	u27	A13- A4	150.000

2.2.1 Solution of the problem

Step 1. Find maximum outcome value of transition T_3 of state S3 for each set of digital software technology subsystems distributions .Which are derived from A5, A6, A7, A8, A9, A10 , A11, A12 , A13 , respectively.

S3

- A5=max[A14,A15,A16]= A15=120.000.
- A6=max[A17,A18,A19]= A19=220.000.
- A7=max[A20,A21,A22]= A21=500.000.

- A8=max[A23,A24,A25]= A23=350.000.
- A9=max[A26,A27,A28]= A28 = 600.000.
- A10= max [A29,A30,A31]= A30= 450.000.
- A11= max [A32,A33,A34]= A34=350.000.
- A12= max [A35,A36,A37]= A37=540.000.
- A13= max [A38,A39,A40]= A39 = 230.000.

Step 2. Now, we have the maximum value of the outcome for A5, A6, A7, A8, A9, A10, A11, A12, A13 subsystems from the state S3, and the outcome of subsystems in S2. So, we calculate the summation between these profits for each digital software technology subsystem related to each set of transition T_2 of state S2 , where $\{A_5, A_6, A_7\}$, $\{A_8, A_9, A_{10}\}$, and $\{A_{11}, A_{12}, A_{13}\}$ are derived from A_2, A_3, A_4.

S2

- A2=max[(A5+200,000),(A6+150,000),(A7+100.000)]= A7=600.00.
- A3=max[(A8+200,000),(A9+310,000),(A10+200,000)]= A9 = 910.000.
- A4=max [(A11+350,000),(A12+400,000),(A13+410,000)] = A13=940.000.

Step 3. Find the maximum value of the outcome for A1 between digital software technology subsystems A2, A3, A4 after calculating the summation for each digital software technology subsystem between the outcome from transition T_2 of state S2 and the outcome of subsystems in S1.

S1

- A1=max[(A2+200,000),(A3+100,000),(A4+50,000)]= A3 = 1.010.000.

So, the sequence of optimal controls is A1 A3 A9 A28.

3 The Model of investment in digital software technology projects in a competitive environment

3.1 Formalization of the problem of investment digital software technology projects in a competitive environment

Suppose teams of software programmer's $k = \overline{1,n}$ are going to jointly implement several digital software technology projects $i = \overline{1,n}$. Each software project i has an execution time T_i which is showing that the problem has many possible solutions $\sigma_m = (i_1, \ldots, i_n)$ based on permutation the order of the projects in time of execution T_i . So, the teams of software programmer's are trying to find minimal execution time using the compromise solution algorithm.

Let the cost function of project is calculated by the formula,

$$F^k(\sigma_m) = \sum_i \alpha_i^k t_i(\sigma_m) \quad (5)$$

where α_i^k- cost of the k -th team of software programmer's for the implementation of the i -th project per unit of time, $t_i(\sigma_m)$- the execution time of the software digital technology project in solution σ_m;

$$t_i(\sigma_m) = t_{i-1}(\sigma_m) + T_i \quad (6)$$

3.2 Example on the problem of investment digital software technology projects in a competitive environment

Now, let us suppose that we have three team's of software programmer's $k = 3$ jointly plan to carry out three digital software technology projects $i = 3$.The task is to determine the optimal order for the software project's for the software project's minimum execution time if

it is known that the costs of the k -th programmer for the implementation of the i -th digital software technology project per unit of time, $\alpha_1^1 = 2; \alpha_2^1 = 6; \alpha_3^1 = 4; \alpha_1^2 = 4; \alpha_2^2 = 4; \alpha_3^2 = 3; \alpha_1^3 = 3; \alpha_2^3 = 5; \alpha_3^3 = 4$. $T_1 = 2; T_2 = 1{,}5; T_3 = 2{,}5$, where T_i is the execution time of the $\boldsymbol{i}$ –th digital software technology project. Let the payoff function $H_i(\sigma_m)$ of each team of software programmer's $\boldsymbol{i}$ in the implementation for each of the possible solutions is also given as follows.

For the first team of software programmer's $k = 1$,

$$H_1(\sigma_1) = 55;$$
$$H_1(\sigma_2) = 60;$$
$$H_1(\sigma_3) = 49;$$
$$H_1(\sigma_4) = 40;$$
$$H_1(\sigma_5) = 65;$$
$$H_1(\sigma_6) = 50;$$

For the first team of software programmer's $k = 2$,

$$H_2(\sigma_1) = 56;$$
$$H_2(\sigma_2) = 50;$$
$$H_2(\sigma_3) = 45;$$
$$H_2(\sigma_4) = 60;$$
$$H_2(\sigma_5) = 55;$$
$$H_2(\sigma_6) = 57;$$

For the third team of software programmer's $k = 3$,

$$H_3(\sigma_1) = 53;$$
$$H_3(\sigma_2) = 60;$$
$$H_3(\sigma_3) = 59;$$
$$H_3(\sigma_4) = 57;$$
$$H_3(\sigma_5) = 66;$$
$$H_3(\sigma_6) = 63.$$

3.2.1 Solution of the Problem

Step 1. Putting all possible solution σ_{m} for the problem can be presented as follows,

$$\sigma_1 = (i_1, i_2, i_3);$$
$$\sigma_2 = (i_1, i_3, i_2);$$
$$\sigma_3 = (i_2, i_1, i_3);$$
$$\sigma_4 = (i_2, i_3, i_1);$$
$$\sigma_5 = (i_3, i_1, i_2);$$
$$\sigma_6 = (i_3, i_2, i_1);$$

Step 2. Calculate the costs $F^k(\sigma_m)$ for each team of software programmer's in the implementation of each of the possible solutions, using the formula 5,

For the first team of software programmer's $k = 1$,

$$F^1(\sigma_1) = 2 * 2 + 6 * (2 + 1{,}5) + 4 * (2 + 1{,}5 + 2{,}5) = 49;$$
$$F^1(\sigma_2) = 2 * 2 + 4 * (2 + 2{,}5) + 6 * (2 + 2{,}5 + 1{,}5) = 58;$$
$$F^1(\sigma_3) = 6 * 1{,}5 + 2 * (1{,}5 + 2) + 4 * (1{,}5 + 2 + 2{,}5) = 40;$$
$$F^1(\sigma_4) = 6 * 1{,}5 + 4 * (1{,}5 + 2{,}5) + 2 * (1{,}5 + 2{,}5 + 2) = 37;$$
$$F^1(\sigma_5) = 4 * 2{,}5 + 2 * (2{,}5 + 2) + 6 * (2{,}5 + 2 + 1{,}5) = 55;$$
$$F^1(\sigma_6) = 4 * 2{,}5 + 6 * (2{,}5 + 1{,}5) + 2 * (2{,}5 + 1{,}5 + 2) = 46;$$

For the second team of software programmer's $k = 2$,

$$\mathrm{F}^2(\sigma_1) = 4 * 2 + 4 * (2 + 1{,}5) + 3 * (2 + 1{,}5 + 2{,}5) = 40;$$
$$\mathrm{F}^2(\sigma_2) = 4 * 2 + 3 * (2 + 2{,}5) + 4 * (2 + 2{,}5 + 1{,}5) = 45.5;$$
$$\mathrm{F}^2(\sigma_3) = 4 * 1{,}5 + 4 * (1{,}5 + 2) + 3 * (1{,}5 + 2 + 2{,}5) = 38;$$

$$F^2(\sigma_4) = 4 * 1{,}5 + 3 * (1{,}5 + 2{,}5) + 4 * (1{,}5 + 2{,}5 + 2) = 42;$$
$$F^2(\sigma_5) = 3 * 2{,}5 + 4 * (2{,}5 + 2) + 4 * (2{,}5 + 2 + 1{,}5) = 49.5;$$
$$F^2(\sigma_6) = 3 * 2{,}5 + 4 * (2{,}5 + 1{,}5) + 4 * (2{,}5 + 1{,}5 + 2) = 47.5;$$

For the third team of software programmer's k = 3,

$$F^3(\sigma_1) = 3 * 2 + 5 * (2 + 1{,}5) + 4 * (2 + 1{,}5 + 2{,}5) = 47.5$$
$$F^3(\sigma_2) = 3 * 2 + 4 * (2 + 2{,}5) + 5 * (2 + 2{,}5 + 1{,}5) = 54$$
$$F^3(\sigma_3) = 5 * 1{,}5 + 3 * (1{,}5 + 2) + 4 * (1{,}5 + 2 + 2{,}5) = 42$$
$$F^3(\sigma_4) = 5 * 1{,}5 + 4 * (1.5 + 2{,}5) + 3 * (1{,}5 + 2{,}5 + 2) = 41.5$$
$$F^3(\sigma_5) = 4 * 2{,}5 + 3 * (2.5 + 2) + 5 * (2.5 + 2 + 1.5) = 53.5$$
$$F^3(\sigma_6) = 4 * 2{,}5 + 5 * (2{,}5 + 1{,}5) + 3 * (2{,}5 + 1{,}5 + 2) = 48.$$

Step3. Calculate the remaining payoff function of each team of software programmer's in the implementation of each of the possible solutions minus the costs. Denote by $H'_k(\sigma_m)$,

$$H'_k(\sigma_m) = H_k(\sigma_m) - F^k(\sigma_m),$$

For the first team of software programmer's k = 1,

$$H'_1(\sigma_1) = 6;$$
$$H'_1(\sigma_2) = 2;$$
$$H'_1(\sigma_3) = 9;$$
$$H'_1(\sigma_4) = 3;$$
$$H'_1(\sigma_5) = 10;$$
$$H'_1(\sigma_6) = 4;$$

For the second team of software programmer's k = 2,

$$H'_2(\sigma_1) = 16;$$
$$H'_2(\sigma_2) = 4.5;$$
$$H'_2(\sigma_3) = 7;$$
$$H'_2(\sigma_4) = 18;$$
$$H'_2(\sigma_5) = 5.5;$$
$$H'_2(\sigma_6) = 9.5;$$

For the third team of software programmer's k = 3,

$$H'_3(\sigma_1) = 5.5;$$
$$H'_3(\sigma_2) = 6;$$
$$H'_3(\sigma_3) = 17;$$
$$H'_3(\sigma_4) = 15.5;$$
$$H'_3(\sigma_5) = 12.3;$$
$$H'_3(\sigma_6) = 15.$$

Step 4. Find for each team of software programmer $M_k = \max_{\sigma_m} H'_k(\sigma_m)$, where K={1,2,3}.

$$M_1 = 10;$$
$$M_2 = 18;$$
$$M_3 = 17.$$

Step5. Calculate the deviation from the maximum of the remaining payoff functions $H'_k(\sigma_m)$ for each team of software programmer's, $\Delta^k_{\sigma_m} = M_k - H'_k(\sigma_m), K = \{1,2,3\}$.

For the first team of software programmer's $\boldsymbol{k = 1}$,

$$M_1 - H_1'(\sigma_1) = 4;$$
$$M_1 - H_1'(\sigma_2) = 8;$$
$$M_1 - H_1'(\sigma_3) = 1;$$
$$M_1 - H_1'(\sigma_4) = 7;$$
$$M_1 - H_1'(\sigma_5) = 6;$$

For the second team of software programmer's $k = 2$,

$$M_2 - H_2'(\sigma_1) = 2;$$

$$M_2 - H_2^{'}(\sigma_3) = 13.5;$$
$$M_2 - H_2^{'}(\sigma_4) = 11;$$
$$M_2 - H_2^{'}(\sigma_5) = 12.5;$$
$$M_2 - H_2^{'}(\sigma_6) = 8.5;$$

For the third team of software programmer's $k = 3$,

$$M_3 - H_3^{'}(\sigma_1) = 11.5;$$
$$M_3 - H_3^{'}(\sigma_2) = 11;$$
$$M_3 - H_3^{'}(\sigma_4) = 1.5;$$
$$M_3 - H_3^{'}(\sigma_5) = 4.5;$$
$$M_3 - H_3^{'}(\sigma_6) = 2;$$

Step 6. Calculate the maximum deviation for each team of software programmer's k of

$$\Delta_{\sigma_m}^{k^*_{\sigma_m}} = \max_k \Delta_{\sigma_m}^k,$$
$$\max_{k=1}(M_1 - H_1'(\sigma_m)) = M_1 - H_1'(\sigma_2) = 8;$$
$$\max_{k=2}(M_2 - H_2'(\sigma_m)) = M_2 - H_2'(\sigma_2) = 13.5;$$
$$\max_{k=3}(M_3 - H_3'(\sigma_m)) = M_3 - H_3'(\sigma_1) = 11.5.$$

Step 7. Choose minimal from all maximal deviations $\Delta_{\sigma_m}^{k^*_{\sigma_m}^*} = \min_{\sigma_m} \Delta_{\sigma_m}^{k^*_{\sigma_m}} = \min_{\sigma_m} \max_k \Delta_{\sigma_m}^k$. The order σ^* on which the minimum is reached is a compromise solution for all teams of software programmer's $\min_{\sigma_m} \max_k (M_k - H_k'(\sigma_m)) = M_1 - H_1'(\sigma_2) = 8$.

Hence, $\sigma_2 = (i_1, i_3, i_2)$, is a compromise solution.

4 The compromise solution in the optimal assignment software teams for digital software technology system projects

4.1 Formalization of the assignment problem of digital software technology in a competitive environment

We may formalize the problem in the following way. Suppose a certain technology company has n different software technology system projects need to work on them , denoted by $J_1, \cdots, J_n$, and suppose n different software teams $I_1, \cdots, I_n$ are possible nominated for each of the software technology system projects . Assume that each software team is assigned a project based on its qualifications and experience, and that each of the n software technology system projects determines the appropriate information needs. Let α_{ij} be a nonnegative number giving the productivity of I_i for the software technology system project J_j , which can be evaluated by owners based on the volume of money decided by the experts of the company for the development process. Thus, listing all possible possibilities or strategies of assigning or nominating software teams I_i to software technology system projects J_j is difficult or impossible, especially if we are dealing with a large number of n parameters. Therefore, here for our problem, we use the optimal assignment method to shed light on each software team suitable for a particular software technology system project to make them satisfied with the maximum productivity [19]

$$\max_{\beta_{ij}} \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij}\alpha_{ij},$$

subject to $\sum_{i=1}^{n} \beta_{ij} \leq 1$ for all j, $\sum_{j=1}^{n} \beta_{ij} \leq 1$ for all i, where the variable β_{ij} is an integers. So, $\beta_{ij} = 1$ if and only if optimal assigning software team I_i to software technology system project J_j ,otherwise $\beta_{ij} = 0$.

Having the optimal value of profit for all owners lead us to go directly to the step of a compromise solution algorithm, which determines the "fairest" strategy in the sense that it minimizes the loss (that is, the shortfall to the maximum possible profit) of the owner who has the maximum loss. Now, let X_k the possible assignment strategies for software technology system projects to software teams for owner k , $X = \prod_1^m X_k$ is the set of all possible assignments strategy collections. Where $K = \{1,2,\cdots,m\}$ is the set of the owners, $H_k: X \to R_1, k \in K$ is the profit function for the owner k .Thus, we can find $M_k = \max_{x \in X} H_k(x)$, to construct the ideal vector $M = (M_1, \cdots, M_m)$. We define the compromise solution as $C_H^X = \min_{x \in X} \max_{k \in K} \big(M_k - H_k(x)\big)$.

4.2 Example of the compromise solution of the assignment software teams for digital software technology systems in a competitive environment

Now, let us explain this algorithm by finding the optimal assignment for $n = 5$ software technology system projects J_1, J_2, J_3, J_4, J_5 and the software teams I_1, I_2, I_3, I_4, I_5 for hardware owner, software owner, internet-time owner in the software technology company. The assignment process for each software team I_i with the different software technology project J_j gives us the productivity matrix α_{ij}^k , where $K = \{1,2,3\}$ the different owners in the company . The hardware owner matrix α_{ij}^1 , the software owner matrix α_{ij}^2 , and the internet-time owner matrix α_{ij}^3, respectively

$$\boldsymbol{\alpha_{ij}^1} = \begin{array}{c} \\ \beta_i \\ I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \end{array} \begin{array}{c} \rho_j \\ J_1 \quad J_2 \quad J_3 \quad J_4 \quad J_5 \\ \begin{bmatrix} 3 & 4 & 11 & 12 & 1 \\ 5 & 3 & 2 & 7 & 6 \\ 3 & 2 & 1 & 4 & 7 \\ 4 & 1 & 2 & 12 & 2 \\ 3 & 2 & 4 & 11 & 7 \end{bmatrix} \end{array}$$

$$\boldsymbol{\alpha_{ij}^2} = \begin{array}{c} \\ \beta_i \\ I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \end{array} \begin{array}{c} \rho_j \\ J_1 \quad J_2 \quad J_3 \quad J_4 \quad J_5 \\ \begin{bmatrix} 7 & 2 & 5 & 7 & 6 \\ 1 & 3 & 1 & 5 & 13 \\ 3 & 1 & 12 & 6 & 7 \\ 11 & 12 & 4 & 1 & 6 \\ 6 & 1 & 2 & 4 & 1 \end{bmatrix} \end{array}$$

$$\begin{array}{c} \rho_j \\ \beta_i \quad J_1 \quad J_2 \quad J_3 \quad J_4 \quad J_5 \end{array}$$

$$\boldsymbol{\alpha_{ij}^{3}} = \begin{matrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \end{matrix} \begin{bmatrix} 1 & 3 & 7 & 1 & 12 \\ 2 & 4 & 8 & 2 & 3 \\ 4 & 2 & 1 & 12 & 11 \\ 3 & 2 & 5 & 6 & 12 \\ 4 & 2 & 7 & 8 & 16 \end{bmatrix}$$

The task is to determine the "fairest" strategy for all owners in a technology company using the compromise solution algorithm, which depends on the optimal assignments of software teams to the digital software technology system projects.

5 Conclusion

All the managers may be facing some problems or difficulties with making decisions on critical issues in the digital software technology development process, especially when related to resource distribution, because they are always looking for the maximum profit. The mathematical model for resource allocation for digital software technology subsystems solves this problem. But in some cases, in a competitive environment, the optimal solution for the manager may be to reach a compromise with the other managers that have different aims using a compromise solution algorithm. Also, we highlighted that finding the compromise solution of profit for all owners depends on the assignments made to software teams with the software technology system projects and that it determines the "fairest" strategy for assignment in the sense that it minimizes the loss to the owner of the technology company, who has the maximum loss in profit. As a result, it is possible that not all participants in the company will achieve their maximum possible productivity. This is one of the examples related to the technological sector [16-18], and as we mentioned earlier, there are many cases and issues around us related to this algorithm, whether in the economic (particularly in the agriculture), social, political, or many other sectors.

References

1. A. Ravindran, D.T. Phillips, J.J. Solberg, *Operations research: principles and practice* (John Wiley & Sons, 1976), p. 585
2. R.L. Graves, P. Wolfe, Recent advances in mathematical programming (1963)
3. S.E. Dreyfus, R. Bellman, *Applied dynamic programming* (Princeton University Press, 1962), p. 363
4. G. Monge, *Mémoire sur la théorie des déblais et des remblais. In: Histoire de l'Académie Royale des Sciences de Paris, avec les Mémoires de Mathématique et de Physique pour la même année* (Paris, 1781), pp. 666-704
5. D. Kőnig, Math Ann **77**, 453-465 (1916)
6. F.G. Frobenius, *Über Matrizen aus nicht negativen Elementen*. Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin, Phys Math Kl (1968), pp. 456-477
7. G.B. Dantzig, *Application of the simplex method to a transportation problem. in: Activity Analysis of Production and Allocation* — Proceedings of a Conference (Proceedings Conference on Linear Programming, Chicago, Illinois, 1949; Tj.C. Koopmans, ed.) (Wiley, New York, 1951), pp. 359-373
8. P.R. Halmos, H.E. Vaughan, J. Math **72**, 214-215 (1950)
9. H.W. Kuhn, On combinatorial properties of matrices. Logistic Papers 11, 4 (George Washington University, 1955)

10. E. Egerváry, Mat Fiz Lapok **38**, 16-28 (1931)
11. H.W. Kuhn, *On the origin of the Hungarian method* (History of mathematical programming, 1991), pp. 77-81
12. A. Schrijver, *Combinatorial optimization: polyhedra and efficiency* (Springer-Verlag, Berlin Heidelberg, 2003), p. 24
13. A. Orden, Manag. Sci. **2**, 276-285 (1956)
14. P. Hall, *On representatives of subsets* (Classic Papers in Combinatorics, 1987), pp. 58-62
15. G.A. Miller, Quarterly, J. Pure Appl. Math. **41**, 382-384 (1910)
16. I. Zaitseva, O. Malafeyev, A. Dolgopolova, V. Zhukova, Y. Vorokhobina, AIP Conference Proceedings **2116**, 450060 (2019)
17. I. Zaitseva, AIP Conference Proceedings **2116**, 450057 (2019)
18. O. Malafeyev, I. Zaitseva, V. Onishenko, V. Petrova, A. Kirjanen, AIP Conference Proceedings **2116**, 450058 (2019)