АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ
ALGORITHMIC AND PROGRAMMING

УДК 004.4–004.9

A. A. Prihozhy
Belarusian National Technical University

DYNAMIC REDUCTION OF TIME COSTS ON IT-PROJECT BY FORMING TEAMS OF COMPATIBLE PROGRAMMERS

The combinatorial problem of forming programming teams has been studied in several works. The proposed techniques and algorithms for solving the problem account for various aspects and parameters of the software development process and programming teams' operation. The problem is NP-hard in general case. Accounting for compatibility of programmers leads to forming teams with increased efficiency of operation which reduces IT-project time costs. Our previous work researched how the compatibility of programmers influences the overall runtime of teams. This paper proposes a more accurate dynamic model of calculating the programmers' time costs changes during forming teams. At each adding of a programmer to a team, the model recalculates the time costs of the programmers and teams accounting for their compatibility. The advanced dynamic optimization algorithm of stepwise pairwise merging of teams be developed in the paper aims to reduce the time costs of the project the programmers are working on. The created software and conducted computational experiments have shown the reduction in project time costs by tens of percent for large sets of programmers.

Keywords: programmer, project, time costs, compatibility of programmers, forming teams, optimization.

For citation: Prihozhy A. A. Dynamic reduction of time costs on IT-project by forming teams of compatible programmers. *Proceedings of BSTU, issue 3, Physics and Mathematics. Informatics*, 2024, no. 1 (278), pp. 70–76.
DOI: 10.52065/2520-6141-2024-278-11.

А. А. Прихожий
Белорусский национальный технический университет

ДИНАМИЧЕСКОЕ СОКРАЩЕНИЕ ЗАТРАТ ВРЕМЕНИ НА ИТ-ПРОЕКТ ПУТЕМ ФОРМИРОВАНИЯ КОМАНД СОВМЕСТИМЫХ ПРОГРАММИСТОВ

Комбинаторная задача формирования команд программистов изучалась в ряде работ. Предложенные методы и алгоритмы решения задачи учитывают различные аспекты и параметры процесса разработки программного обеспечения и работы команд программистов. В общем случае задача является NP-трудной. Учет совместимости программистов приводит к формированию команд с повышенной эффективностью работы, что значительно сокращает временные затраты на ИТ-проект. Наши предыдущие работы исследовали, как совместимость программистов влияет на общее время работы команд. В данной работе предлагается более точная динамическая модель расчета изменения временных затрат программистов в процессе формировании команд. При каждом добавлении программиста в команду модель пересчитывает временные затраты программистов и команд с учетом их совместимости. Разработанный в статье алгоритм динамической оптимизации путем пошагового попарного слияния команд направлен на снижение временных затрат на проект, над которым работают программисты. Созданное программное обеспечение и проведенные вычислительные эксперименты показали снижение временных затрат на проект на десятки процентов при большом количестве участников проекта.

Ключевые слова: программист, проект, временные затраты, совместимость программистов, формирование команд, оптимизация.

Для цитирования: Прихожий А. А. Динамическое сокращение затрат времени на ИТ-проект путем формирования команд совместимых программистов // Труды БГТУ. Сер. 3. Физико-математические науки и информатика. 2024. № 1 (278). С. 70–76 (In English).
DOI: 10.52065/2520-6141-2024-278-11.

Introduction. The problem of forming programming teams and managing projects has been studied in works [1–10]. The problem is combinatorial and NP-hard in general case. Therefore, exact and heuristic algorithms have been developed for solving it for various objective functions and constraints. Work [11] has considered how the compatibility of programmers influences the overall runtime of teams and how the influence of programmers on each other can be used for reducing the project time costs. A matrix of compatibility of programmers has been proposed and a greedy algorithm of stepwise pairwise merge of teams has been developed at the aim of solving the problem of forming teams. The algorithm analyses and exploits programmers' compatibility to find the number, size and staff of the teams reducing the overall runtime.

In this paper, we propose a more accurate dynamic model of calculating changes in the time costs of programmers during forming teams, propose and implement an advanced optimization dynamic algorithm of stepwise pairwise merge of programming teams.

Main part. Let $P = \{p_1, \ldots, p_n\}$ be a set of n programmers working on an IT project. Vector $t = (t_1, \ldots t_i, \ldots t_n)$ describes the programmers' basic time costs, which do not include interaction costs within team.

Let $G = \{g_1 \ldots g_k\}$ be a set of teams the programmers are allocated to. If programmers are in a same team, their time costs must be corrected depending on the compatibility of programmers. Matrix dP represents corrections (%) of the programmers' costs. In matrix principal diagonal, $dP_{i,i} = t_i$. For programmers i and j, $dP_{i,j}$ ($dP_{j,i}$) shows how programmer i (j) influences on t_j (t_i). Values $dP_{i,j}$ and $dP_{j,i}$ can be negative and positive. Four combinations are possible: 1) $dP_{i,j} < 0$ and $dP_{j,i} < 0$; 2) $dP_{i,j} \geq 0$ and $dP_{j,i} \geq 0$; 3) $dP_{i,j} < 0$ and $dP_{j,i} \geq 0$; 4) $dP_{i,j} \geq 0$ and $dP_{j,i} < 0$. The first combination is the most preferable since the time costs of both programmers are reduced.

In work [11], the changes in the programmers' time costs are calculated with $dT_{i,j} = t_j \cdot dP_{i,j} / 100$ before allocating the programmers to a team and then are summed. A drawback of the approach is that for significant changes of programmers' time costs the overall costs can become negative. In this paper, we develop more accurate model for estimating the time costs changes. Every adding of a programmer to a team causes the recalculation of the costs using matrix TP.

$$TP = \begin{bmatrix} T_1 & d_{1,2} & \cdots & d_{1,n} \\ d_{2,1} & T_2 & \cdots & d_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & T_n \end{bmatrix}.$$

Initially, $T_i = t_i$, $i = 1 \ldots n$. The non-diagonal element $d_{i,j} = 100 \cdot dP_{j,i}$ is a positive or negative share of T_j that is added to T_j if programmers i and j are included in a same team: $T_j = T_j + T_j \cdot d_{i,j}$. According to the model, adding a new programmer to a team immediately changes the time costs of all programmers belonging to the team. The positive value of $d_{i,j}$ makes larger the T_j costs, and the negative value makes them smaller. To reduce the time costs of team g, programmers with negative $d_{i,j}$ should be included in the team first.

Theorem 1. Let $u = u_1 \ldots u_{|g|}$ be an order of including programmers in team g. The overall time costs $T(g)$ of the programmers of team g is determined by (1).

$$T(\mathrm{g}) = \sum_{p=u_1 \ldots u_{|\mathrm{g}|}} \left(T_p \cdot \prod_{\substack{j=u_1 \ldots u_{|\mathrm{g}|} \\ j \neq p}} \left(1 + d_{j,p}\right) \right). \quad (1)$$

Proof. Let's prove by induction that the time costs of programmer p from team $h^k = \{u_1 \ldots u_k\}$, $k \leq |g|$ are determined by equation

$$T(p, h^k) = T_p \cdot \prod_{\substack{j=u_1 \ldots u_k \\ j \neq p}} \left(1 + d_{j,p}\right). \quad (2)$$

Base case. Let a team of two programmers (shown in Fig. 1 by dark cells of matrix TP^{k-1}) be $h^2 = \{u_1, u_2\}$. Then programmer u_2 influences on the time costs of programmer u_1 and, therefore,

$$T(u_1, h^2) = T_{u_1} \cdot \left(1 + d_{u_2,u_1}\right).$$

Similarly,

$$T(u_2, h^2) = T_{u_2} \cdot \left(1 + d_{u_1,u_2}\right).$$

The time costs of programmer $j = u_3 \ldots u_k$ who establishes a separate team are T_{j}.

Induction step. Suppose (2) holds for $h^{k-1} = \{u_1 \ldots u_{k-1}\}$ as shown by dark cells of dimension $(k-1)\times(k-1)$ in Fig. 2 describing matrix TP^2 for two teams. The costs of programmer u_k who is not in team h^{k-1} are T_{uk}. If u_k is added to h^{k-1}, a team h^k is established (filled block of dimension $k\times k$ in Fig. 3). Programmer u_k gets influence on the time costs of each of programmers $u_1 \ldots u_{k-1}$. Therefore, their time costs are multiplied by factors $(1 + d_{uk,u1})$, … , $(1 + d_{uk,uk-1})$ respectively, which is compliant with (2). In their turn, all the programmers get influence on the time costs of programmer u_k with factor $(1 + d_{u1,uk}) \cdot \ldots \cdot (1 + d_{uk-1,uk})$. As a result, the time costs of programmer u_k are determined by (2).

The elements of principal diagonal of matrix TP^1 shown in Fig. 3 prove that the time costs of all programmers included in a team are calculated with (2), and at $k = |g|$ and $h^k = g$, the overall sum of the programmers' time costs is equal to $T(\mathrm{g})$ defined by (1). The theorem is proved.

Corollary 1. The value of $T(g)$ defined by (1) does not depend on the order of including programmers in team g.

$$TP^{k-1} = \begin{bmatrix} T_{u_1} \cdot (1 + d_{u_2,u_1}) & d_{u_1,u_2} & \cdots & d_{u_1,u_k} \\ d_{u_2,u_1} & T_{u_2} \cdot (1 + d_{u_1,u_2}) & \cdots & d_{u_2,u_k} \\ \vdots & \vdots & \ddots & \vdots \\ d_{u_k,u_1} & d_{u_k,u_2} & \cdots & T_{u_k} \end{bmatrix}$$

Fig. 1. Matrix TP^{k-2} of time costs of k programmers included in $k – 1$ teams (two programmers are in the first team)

$$TP^{2} = \begin{bmatrix} T_{u_1} \cdot \prod_{j=u_2 \ldots u_{k-1}} (1 + d_{j,u_1}) & \cdots & d_{u_1,u_{k-1}} & d_{u_1,u_k} \\ \vdots & \ddots & \vdots & \vdots \\ d_{u_{k-1},u_1} & \cdots & T_{u_{k-1}} \cdot \prod_{j=u_1 \ldots u_{k-2}} (1 + d_{j,u_{k-1}}) & d_{u_{k-1},u_k} \\ d_{u_k,u_1} & \cdots & d_{u_k,u_{k-1}} & T_{u_k} \end{bmatrix}$$

Fig. 2. Matrix TP^2 of time costs of k programmers included in 2 teams ($k – 1$ programmer are in the first team)

$$TP^{1} = \begin{bmatrix} T_{u_1} \cdot \prod_{j=u_2 \ldots u_k} (1 + d_{j,u_1}) & \cdots & d_{u_1,u_{k-1}} & d_{u_1,u_k} \\ \vdots & \ddots & \vdots & \vdots \\ d_{u_{k-1},u_1} & \cdots & T_{u_{k-1}} \cdot \prod_{j=u_1 \ldots u_k, j \neq u_{k-1}} (1 + d_{j,u_{k-1}}) & d_{u_{k-1},u_k} \\ d_{u_k,u_1} & \cdots & d_{u_k,u_{k-1}} & T_{u_k} \cdot \prod_{j=u_2 \ldots u_{k-1}} (1 + d_{j,u_k}) \end{bmatrix}$$

Fig. 3. Matrix TP^1 of time costs of k programmers included in single team

Proof. Let u be a permutation that determines the order of including the programmers in team g. Let's reorder the programmers listed in u to obtain a permutation v such that $v_j = j$, $j = 1\ldots|g|$. To do this, we find element $u_k = j$ and exchange it with element u_j for $j = 1\ldots|g|$. According to (2), the exchange does not change the value of $T(p, g)$ for the elements of matrix TP^1's principal diagonal (Fig. 3) since the multiplication operation used in expression $(1 + d_{u1,p}) \cdot \ldots \cdot (1 + d_{uk,p})$ is commutative and associative. Any permutation of the programmers can be replaced with v, therefore all of them yield the same value of $T(p, g)$. The corollary is proved.

Corollary 1 allows to rewrite (1) as

$$T(\mathrm{g}) = \sum_{p \in g} \left(T_p \cdot \prod_{\substack{j \in g \\ j \neq p}} \left(1 + d_{j,p}\right) \right). \quad (3)$$

If $d_{j,p}$ is negative, then $(1 + d_{j,p}) < 1$, therefore, $T(p, g)$ is decreased. If $d_{j,p}$ is positive, then $(1 + d_{j,p}) > 1$, therefore, $T(p, g)$ is increased. The value of $T(p, g)$ is smaller if the larger number of negative elements $d_{j,p}$ are in the matrix.

The overall time costs of teams of set G are

$$T^G = \sum\nolimits_{g \in G} T(\mathrm{g}). \quad (4)$$

If Ω is a set of all possible partitioning of set P of programmers into a set G of teams, the combinatorial optimization problem we solve is

$$\min_{G \in \Omega} T^G. \quad (5)$$

In the paper, we propose a dynamic greedy algorithm to solve (5) heuristically for large sets of programmers. Unlikely to [11], the algorithm recalculates time costs of programmers at every step of pairwise merge of teams.

The dynamic greedy algorithm of stepwise pairwise merge of teams (DGAMT) is described by Algorithm 1. The set P of programmers, vector t of programmers' basic time costs and matrix TP of pairwise changes of programmers' time costs are its inputs. The set G of teams and the overall time costs T^G are its outputs. *DGAMT* is derived from Theorem 1 and Corollary 1.

Performing initialization, the algorithm allocates each programmer p_i to team $\{p_i\}$. The teams' overall time costs T^G are the sum of t_i, $i = 1\ldots n$. Each element of two-dimensional array ΔT is initialized by calculating a difference between $T(g' \cup g'')$ and $T(g') + T(g'')$ where g' and g'' are teams-candidates for merging.

Algorithm 1: Dynamic greedy algorithm of stepwise pairwise merge of teams (*DGAMT*)

```
Input: A set P = {p1, …, pn} of programmers
Input: A vector t = (t1…tn) of programmers' basic time
       costs
Input: A matrix TP[n×n] of programmers' pairwise
       time costs changes
Output: A set G of programming teams
Output: A runtime Time(G) of programming teams
G ← ∅   T^G ← 0   go ← true
for i ← 1 to n do
    gi ← {pi}   T(gi) ← t(pi) ← ti
    G ← G ∪ {gi}   T^G ← T^G + ti
for g' ∈ G do
    BestC(g').team ← undefined
    BestC(g').ΔT ← ∞
    for g" ∈ G do
        T(g' ∪ g") ← TeamRuntime(P, t(p), TP, g', g")
        ΔT(g', g") ← T(g' ∪ g") – T(g') – T(g")
        ΔT(g", g') ← ΔT(g', g")
        if BestC(g').ΔT > ΔT(g', g") then
            BestC(g').ΔT ← ΔT(g', g")
            BestC(g').team ← g"
while (go) do
    go ← false
    g' ←SelectBestPairOfTeams(G, BestC)
    if BestC(g').ΔT < 0 then
        go ← true
        g" ← BestC(g').team
        g ← g' ∪ g"
        t ← UpdateProgrammerCosts(P, t, TP, g', g")
        T(g) ← T(g') + T(g") + BestC(g').ΔT
        G ← (G \ {g', g"}) ∪ {g}
        T^G ← T^G + BestC(g').ΔT
        BestC(g).team ← undefined
        BestC(g).ΔT ← ∞
        for g# ∈ G \ {g} do
            T(g ∪ g#) ← TeamRuntime(P, t(p), TP, g, g#)
            ΔT(g, g#) ← T(g ∪ g#) – T(g) – T(g#)
            ΔT(g#, g) ← ΔT(g, g#)
            if BestC(g).ΔT > ΔT(g, g#) then
                BestC(g).ΔT ← ΔT(g, g#)
                BestC(g).team ← g#
            if BestC(g#).ΔT > ΔT(g#, g) then
                BestC(g#).ΔT ← ΔT(g#, g)
                BestC(g#).team ← g
return G, Time(G)
```

Algorithm 2: Calculating time costs of team formed by merging a pair of selected teams (*TeamRuntime*)

```
Input: A set P = {p1, …, pn} of programmers
Input: A vector t(p) = (t(p1)… t(pn)) of programmers'
       time costs in IT project
Input: A matrix TP[n×n] of pairwise changes of pro-
       grammers' time costs
Input: Teams g' and g" selected for merging
Output: Time costs T(g' ∪ g") of union of two teams
  T(g' ∪ g") ← 0
  for v ∈ g' do
      t*(v) ← t(v)
      for u ∈ g" do
          t*(v) ← t*(v) · (1 + TP(u, v))
      T(g' ∪ g") = T(g' ∪ g") + t*(v)
  for v ∈ g" do
      t*(v) ← t(v)
      for u ∈ g' do
          t*(v) ← t*(v) · (1 + TP(u, v))
      T(g' ∪ g") = T(g' ∪ g") + t*(v)
return T(g' ∪ g")
```

Each element $BestC(g')$ of vector $BestC$ is initialized with $BestC(g').team$ which is paired with g' and has maximal reduction $BestC(g').\Delta T$ of time costs. Function *TeamRuntime* (Algorithm 2) calculates the time costs $T(g)$ of team that is a result of merging $g = g' \cup g''$. At every iteration of the *while* loop, *DGAMT* calls function *SelectBestPairOf-Teams* to choose a pair of teams g' and g'' whose element in ΔT is minimal. If all elements of ΔT are not negative, the process of merging is over. Otherwise, the set G of teams is reconstructed: teams g' and g'' are removed from G and team $g = g' \cup g''$ is added to G. The time costs for each p, g and G are calculated using (2), (3) and (4).

Algorithm 3: Calculating time costs of programmers of two teams to be merged (*UpdateProgrammerCosts*)

```
Input: A set P = {p1, …, pn} of programmers
Input: A vector t(p) = (t(p1)… t(pn)) of time costs
Input: A matrix TP[n×n] of pairwise changes of pro-
       grammers' time costs
Input: Teams g' and g" of programmers to be merged
Output: An updated vector t(p) of time costs
  for v ∈ g' do
      for u ∈ g" do
          t(v) ← t(v) · (1 + TP(u, v))
  for v ∈ g" do
      for u ∈ g' do
          t(v) ← t(v) · (1 + TP(u, v))
  return t(p)
```

For new team g and for each another team $g\# \in G$, the value of $\Delta T(g, g\#)$ is calculated. This may cause updating elements of vector *BestC*.Function *Update-ProgrammerCosts* (Algorithm 3) updates using (2) the time costs of programmers from teams g' and g'' that are intended to be merged.

Example. Let $P = \{p_1 \ldots p_8\}$ be a set of eight programmers. Vector $t = (93, 15, 47, 45, 79, 92,$

67, 64) describes basic time costs of the programmers in a project. Fig. 4 gives matrix dP of programmers' pairwise time costs changes (%). The overall time costs of eight teams (one programmer per team) are 502.0. The overall time costs of the single team (it contains eight programmers) are 469.1 (6.6% less). Fig. 5 describes step 1 of *DGAMT*'s operation.

The rows and columns of matrix ΔT correspond to eight teams of set $G = \{\{p_1\}, \{p_2\}, \{p_3\}, \{p_4\}, \{p_5\}, \{p_6\}, \{p_7\}, \{p_8\}\}$. Element $\Delta T_{ij} = T(\{p_i, p_j\}) - T(\{p_i\}) - T(\{p_j\})$, $i, j = 1 \ldots 8$, $i \neq j$, is calculated using (3). Elements of vector t are in principal diagonal of ΔT. The Δ and *team* elements of components of vector *BestC*'s are in the right column (Fig. 5). Since *BestC*(1).Δ and *BestC*(8).Δ have a minimum value of – 13.22, teams $\{p_1\}$ and $\{p_8\}$ are selected for merging at step 2. Fig. 6 describes step 2 of *DGAMT*. Teams $\{p_1\}$ and $\{p_8\}$ are merged to $\{p_1, p_8\}$. The overall number of teams is reduced to seven. *DGAMT* removes two rows and two columns from matrix ΔT corresponding to teams $\{p_1\}$ and $\{p_8\}$ and adds one row and one column for the new team $\{p_1, p_8\}$. Observing column *BestC*, we see that teams $\{p_7\}$ and $\{p_1, p_8\}$ give a maximum reduction – 14.99 of the time costs. They are selected for merging at step 3.

dP =

93	9.83	0	-1.87	4.66	6.88	-3.57	-8.36
1.32	15	1.19	3.61	-2.68	-9.36	1.49	-3.67
-7.33	-7.38	47	-9.15	-3.4	4.68	-5.19	9.18
-7.64	6.07	5.76	45	-7.45	1.41	7.13	-3.38
-4.14	-1.23	9.41	-8.06	79	3.51	-2.65	8.24
6.2	9.86	4.03	-6.81	7.51	92	9.72	-6.59
-6.52	8.18	0	0	-3.6	0	67	-4.64
-8.46	-1.96	-7.43	-1.65	-2.35	-2.26	-6.69	64

Fig. 4. Example matrix *dP* of programmers' pairwise time costs changes (%) after including in a same team

ΔT =

	Teams								Programmers	BestC	
	1	2	3	4	5	6	7	8		Δ	team
1	93	2.70	-6.82	-7.95	-0.17	12.09	-8.45	-13.22	{1}	-13.22	8
2	2.70	15	-0.55	2.54	-2.30	-7.13	2.23	-2.64	{2}	-7.13	6
3	-6.82	-0.55	47	-1.41	1.73	6.20	-3.48	2.39	{3}	-6.82	1
4	-7.95	2.54	-1.41	45	-9.51	-1.77	4.78	-2.91	{4}	-9.51	5
5	-0.17	-2.30	1.73	-9.51	79	9.17	-4.62	3.42	{5}	-9.51	4
6	12.09	-7.13	6.20	-1.77	9.17	92	6.52	-6.30	{6}	-7.13	2
7	-8.45	2.23	-3.48	4.78	-4.62	6.52	67	-7	{7}	-8.45	1
8	-13.22	-2.64	2.39	-2.91	3.42	-6.30	-7.46	64	{8}	-13.22	1

Fig. 5. Step 1 of merging a pair of teams from $G = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}\}$ by *DGAMT*

ΔT =

	Teams							Programmers	BestC	
	1	2	3	4	5	6	7		Δ	team
1	15.0	-0.55	2.54	-2.30	-7.13	2.23	0.12	{2}	-7.13	5
2	-0.55	47.0	-1.41	1.73	6.20	-3.48	-4.35	{3}	-4.35	7
3	2.54	-1.41	45.0	-9.51	-1.77	4.78	-10.06	{4}	-10.06	7
4	-2.30	1.73	-9.51	79.0	9.17	-4.62	3.05	{5}	-9.51	3
5	-7.13	6.20	-1.77	9.17	92.0	6.52	5.52	{6}	-7.13	1
6	2.23	-3.48	4.78	-4.62	6.52	67.0	-14.99	{7}	-14.99	7
7	0.12	-4.35	-10.06	3.05	5.52	-14.99	143.8	{1, 8}	-14.99	6

Fig. 6. Step 2 of merging a pair of teams from $G = \{\{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{1, 8\}\}$ by *DGAMT*

Stepwise merge of teams by *DGAMT* at seven steps

Step	Team count	Teams	Overall time costs	Pair of teams merged	Time costs reduction
1	8	{1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}	502.0	{1} and {8}	–13.22
2	7	{2}, {3}, {4}, {5}, {6}, {7}, {1, 8}	488.8	{7} and {1, 8}	–14.99
3	6	{2}, {3}, {4}, {5}, {6}, {1, 7, 8}	473.8	{4} and {5}	–9.51
4	5	{2}, {3}, {4, 5}, {6}, {1, 7, 8}	464.3	{3} and {1, 7, 8}	–7.32
5	4	{2}, {4, 5}, {6}, {1, 3, 7, 8}	457.0	{2} and {6}	–7.13
6	3	{4, 5}, {2, 6}, {1, 3, 7, 8}	449.8	{4, 5} and {1, 3, 7, 8}	–5.00
7	2	{2, 6}, {1, 3, 4, 5, 7, 8}	444.8	-	24.45

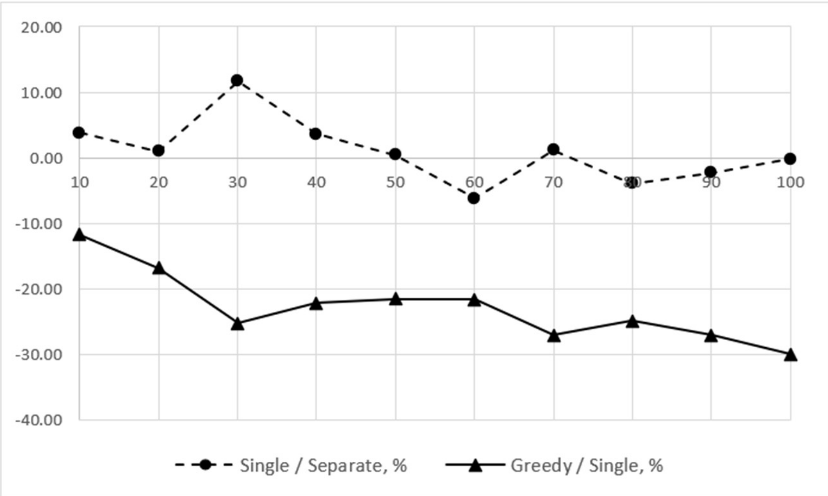

Fig. 7. Comparison of time costs (%) of single team with costs of one-programmer teams (triangles) and comparison of time costs (%) of dynamic greedy teams with costs of single team (diamonds) vs. programmer count

Table briefly describes 7 steps of *DGAMT*'s operation. The time costs have been monotonically reduced. At step 7, the minimum time costs reduction became positive, therefore, the merge is over. Finally, *DGAMT* has obtained two teams having the overall time costs of 444.8 (that is 11.4% less than the costs of eight initial teams).

Results. We have implemented *DGAMT* in the C++ language using Visual Studio 2022 under OS Windows 10. Experiments have been conducted on Intel Core i7-10700 CPU processor using various *P*, *t* and *dP*. Fig. 7 compares the overall time costs of *one-programmer* teams, *single* teams and *dynamic greedy* teams obtained by *DGAMT* for sets of 10 to 100 programmers. Vector *t* and matrix *dP* (average value of element is 5%) were unique for each set of programmers. The time costs of single team differed from those of one-programmer teams by – 6.17% to 12.65% depending on the compatibility of programmers. *DGAMT* have yielded greedy teams having time costs –11.75% to –29.88% lower against *single* teams.

Conclusion. In the paper, we have proposed an accurate model of calculating the IT project time costs which accounts for compatibility of programmers and updates the programmers' time costs at each adding of a programmer to a team. We have used the model for reducing the overall time costs by means of finding an appropriate number of teams, size, and staff of each team. The dynamic greedy algorithm of stepwise pairwise merge of teams realizes the model and shows high accuracy and efficiency while forming programming teams for working on an IT project.

References

1. Rachlin, J. N., Goodwin, R. T., Murthy, S., Akkiraju, R., Wu, F., Kumaran, S., Das, R. A-Teams: An Agent Architecture for Optimization and Decision-Support. In: Müller, J. P., Rao, A. S., Singh, M. P. (eds). *Intelligent Agents V: Agents Theories, Architectures, and Languages. ATAL, 1998. Lecture Notes in Computer Science.* Berlin; Heidelberg, Springer, 1999, vol. 1555, pp. 1–15.

2. Britto R., Neto P. S., Rabelo R., Ayala W. and Soares T. A hybrid approach to solve the agile team allocation problem. *2012 IEEE Congress on Evolutionary Computation*, 2012, pp. 1–8.

3. Prihozhy A. A., Zhdanouski A. M. Genetic algorithm of allocating programmers to groups. *Nauka – obrazovaniyu*, *proizvodstvu*, *ekonomike*: *materialy 13-y Mezhdunarodnoy nauchno-prakticheskoy konferentsii*

[Science to education, industry and economics: Proceedings of 13th international conference. Vol. 1]. Minsk, 2015, pp. 286–287 (In Russian).

4. Gutierrez J. H., Astudillo C. A., Ballesteros-Perez P., Mora-Melia D. and Candia-Vejar A. The multiple team formation problem using sociometry. *Computers and Operations Research*, 2016, vol. 75, pp. 150–162.

5. Masood Z., Hoda R., Blincoe K. Exploring Workflow Mechanisms and Task Allocation Strategies in Agile Software Teams. In: Baumeister H., Lichter H., Riebisch M. (eds). *Agile Processes in Software Engineering and Extreme Programming. XP 2017. Lecture Notes in Business Information Processing*. Springer, 2017, vol. 283, pp. 267–273.

6. Prihozhy A. A., Zhdanouski A. M. Method of qualification estimation and optimization of professional teams of programmers. *System analysis and applied information science*, 2018, no. 2, pp. 4–11 (In Russian).

7. Prihozhy A. A. Exact and greedy algorithms of allocating experts to maximum set of programmer teams. *System analysis and applied information science,* 2022, no. 1, pp. 40–46.

8. Prihozhy A., Zhdanouski A. Genetic algorithm of optimizing the size, staff and number of professional teams of programmers. *Open Semantic Technologies for Intelligent Systems*, Minsk, BSUIR Publ., 2019, pp. 305–310.

9. Prihozhy A. A., Zhdanouski A. M. Genetic algorithm of optimizing the qualification of programmer teams. *System analysis and applied information science*, 2020, no. 4, pp. 31–38.

10. Prihozhy A. A. Optimization of data allocation in hierarchical memory for blocked shortest paths algorithms. *System analysis and applied information science*, 2021, no. 3, pp. 40–50.

11. Prihozhy A. A. Optimization of programming teams on compatibility of programmers. *Trudy BGTU* [Proceedings of BSTU], issue 3, Physics and Mathematics. Informatics, 2023, no 2 (272), pp. 104–110.

Список литературы

1. A-Teams: An Agent Architecture for Optimization and Decision-Support / J. N. Rachlin [et al.] // Intelligent Agents V: Agents Theories, Architectures, and Languages, ATAL 1998: Lecture Notes in Computer Science / J. P. Müller, A. S. Rao, M. P. Singh (eds). Berlin; Heidelberg: Springer. 1999. Vol. 1555. P. 1–15.

2. A hybrid approach to solve the agile team allocation problem / R. Britto [et al.] // 2012 IEEE Congress on Evolutionary Computation. 2012. P. 1–8.

3. Прихожий А. А., Ждановский А. М. Генетический алгоритм разбиения коллектива программистов на группы // Наука – образованию, производству, экономике: материалы 13-й Междунар. науч.-практ. конф. Минск, 2015. Т. 1. С. 286–287.

4. The multiple team formation problem using sociometry / J. H. Gutierrez [et al.] // Computers and Operations Research. 2016. Vol. 75. P. 150–162.

5. Masood Z., Hoda R., Blincoe K. Exploring Workflow Mechanisms and Task Allocation Strategies in Agile Software Teams // Agile Processes in Software Engineering and Extreme Programming. XP 2017: Lecture Notes in Business Information Processing / H. Baumeister, H. Lichter, M. Riebisch (eds). Springer, 2017. Vol. 283. P. 267–273.

6. Прихожий А. А., Ждановский А. М. Метод оценки квалификации и оптимизация состава профессиональных групп программистов // Системный анализ и прикладная информатика. 2018. № 2. С. 4–11.

7. Prihozhy A. A. Exact and greedy algorithms of allocating experts to maximum set of programmer teams // System analysis and applied information science. 2022. No. 1. P. 40–46.

8. Prihozhy A., Zhdanouski A. Genetic algorithm of optimizing the size, staff and number of professional teams of programmers // Open Semantic Technologies for Intelligent Systems. Minsk, BSUIR, 2019. P. 305–310.

9. Prihozhy A. A., Zhdanouski A. M. Genetic algorithm of optimizing the qualification of programmer teams // System analysis and applied information science. 2020. No. 4. P. 31–38.

10. Prihozhy A. A. Optimization of data allocation in hierarchical memory for blocked shortest paths algorithms // System analysis and applied information science. 2021. No. 3. P. 40–50.

11. Prihozhy A. A. Optimization of programming teams on compatibility of programmers // Труды БГТУ. Сер. 3, Физико-математические науки и информатика. 2023. № 2 (272). С. 104–110.

Information about the author

Prihozhy Anatoly Alexievich − DSc (Engineering), Professor, Professor, the Department of Computer and System Software. Belarusian National Technical University (65, Nezalezhnasti Ave., 220013, Minsk, Republic of Belarus). E-mail: prihozhy@yahoo.com

Информация об авторе

Прихожий Анатолий Алексеевич − доктор технических наук, профессор, профессор кафедры программного обеспечения информационных систем и технологий. Белорусский национальный технический университет (220013, г. Минск, пр. Независимости 65, Республика Беларусь). E-mail: prihozhy@yahoo.com

Received 07.12.2023