Decision Making: Applications in Management and Engineering
Vol. 6, Issue 2, 2023, pp. 691-721.
ISSN: 2560-6018
eISSN: 2620-0104
DOI: https://doi.org/10.31181/dmame622023713

THE GAME PROBLEM OF ASSIGNING STAFF TO PROJECT IMPLEMENTATION

Agnieszka Kowalska-Styczeń[1*], Petro Kravets[2], Vasyl Lytvyn[2], Victoria Vysotska[2,3] and Oksana Markiv[2]

[1] Faculty of Organization and Management, Silesian University of Technology, Zabrze, Poland
[2] Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine
[3] Institute of Computer Science, Osnabrück University, Osnabrück, Germany

Received: 28 February 2023;
Accepted: 2 July 2023;
Available online: 25 July 2023.

Original scientific paper

Abstract: *This article describes how to solve the game problem of assigning staff to work on projects based on the ontological approach. The stochastic game algorithm for colouring an undirected random graph has been used to plan project implementation. The stochastic game mathematical model has been described, and the self-learning Markov method has been used for its solution. It is highlighted that the goal of the players is to minimize the functions of average losses. The Markov recurrent method that provides the adaptive choice of colours for the vertices of the random graph based on dynamic vectors of mixed strategies, the values of which depend on the current losses of players has been used. A computer experiment was carried out, which confirmed the convergence of the stochastic game for the problem of colouring the random graph. In conclusion. the possibility of defining the procedure for appointing staff to implement projects has been justified.*

Key words: *Random graph colouring, stochastic game, project implementation, Markov recurrent method, adaptation, self-learning.*

1. Introduction

Nowadays, in the modern information space with developed computer networks and means of telecommunications, the formation issues of virtual professional teams are essential for the development of the project, especially, in conditions of remote work (Charteris et al., 2021; Chen & Wang, 2011; Lawrence, 2022; Maynard et al., 2019; Medeni et al., 2011; Umoren et al., 2017; Whillans et al., 2021; Yang & Chen,

* Corresponding author.
E-mail addresses: agnieszka.kowalska-styczen@polsl.pl (A. Kowalska-Styczeń), Petro.O.Kravets@lpnu.ua (P. Kravets), Vasyl.V.Lytvyn@lpnu.ua (V. Lytvyn), Victoria.A.Vysotska@lpnu.ua (V. Vysotska), oksana.o.markiv@lpnu.ua (O. Markiv)

2020). Appropriate teamwork of qualified staff is the key point to fast and effective project implementation (Heagney, 2012).

The assignment of staff to carry out projects is similar to the classic task of resources (equipment or people) assignment to jobs implementation.

However, there is a significant limitation in the such formulation of the task. As a general rule, every worker must be assigned exclusively to one job and all workers are interchangeable and completely reliable. In the practice of creating project teams, it is possible to deal with a more complicated situation. The project implementation requires various specialists and the staff can be partially interchangeable and unreliable. Then each project executor can be characterized by the probability of refusal. Refusals may exist due to the lack of experience, uncoordinated work, changes in contractors' health, or other unforeseen factors that affect project quality and timing.

Considering identified differences, the method of colouring graphs can be used to solve the problem of allocating staff to projects implementation. The schedule will be planned so that the vertices will define the projects, and the edges will cooperate only with that project's vertices for the implementation in which individual executors are involved. The vertices on opposite edges should be painted in different colours. Then, projects with vertices of the same colour can be executed in parallel, and those with different colours can be conducted sequentially over time.

Renewable refusals of project executors form a dynamic structure of links between the graph vertices. Therefore, it is necessary to consider a random graph of projects instead of the deterministic one. The random graph colouring task belongs to the NP-complex class (Asdre et al., 2007; Li et al., 2018). To solve it, it is necessary to use or develop self-learning or adaptive methods that can provide colouring, which is close to the optimal one in the allowable polynomial time.

Considering that the graph is a structural model of a distributed system and that the random graph colouring task has aspects of local competing and global consistency of goals, there is a need to use an adaptive stochastic game method to solve it. Adaptive methods can directly or indirectly control random processes and form selective solutions, which optimize the mean values of random variables or their stochastic moments.

The research aim is to solve the staff assignment problem to carry out projects based on the stochastic game model for random graph colouring. In this article, the mathematical model and the algorithm of the stochastic game were therefore developed, a computer simulation was performed, the results were analysed and recommendations for their practical application were formulated.

The scientific novelty of the work lies in the application of the stochastic game model to solve the problem of assigning staff to perform projects in conditions of uncertainty based on the ontological coverage of projects by colouring a stochastic graph.

2. An overview of problem-solving methods

The task of staff assignment to carry out projects is formulated similarly to the problem of equipment matching, resources, tasks, programs, scheduling, classification and other similar tasks (Liu, 2020; Makris, 2021, 2021a; Panik, 2017; Torabi Yeganeh & Zegordi, 2020; Wang & Man, 2021; Yang & Chen, 2020).

The classic definition of a general assignment problem is the following. There are L types of work or tasks and $K \geq L$ resources that can be used for their implementation (machines, devices, robots, software agents, people). The costs $c_{i,j}$ of using the *j*-th resource (*j*=1..*K*) to perform the *i*-th (*i*=1..*L*) work are given. Each resource can be used for only one type of work (it is considered that resources are interchangeable). So, there is a need to find such a plan $u = \left[u_{i,j} \in \{0,1\} \mid i = 1..L,\ j = 1..K \right]$, where the work $u_{i,j} \in \{0,1\}$ is performed (means to distribute the resources between the works) so that the total costs $f(u)$ are minimal. Variable $u_{i,j} = 1$ if the resource *j* implements the work *i*, and $u_{i,j} = 0$ in another case.

The mathematical model with the formulated problem consists of the following issues:

1) objective function, which determines the total cost of using the resource to perform work: $f(u) = \sum_{i=1}^{L} \sum_{j=1}^{K} c_{i,j} u_{i,j} \to \min_x$;
2) constraints: $\sum_{j=1}^{K} u_{i,j} = 1$, *i*=1..*L* – if only one *j*-th resource can be assigned to perform the *i*-th work; $\sum_{i=1}^{L} u_{i,j} \le 1$, *j*=1..*K* – if $K > L$.

The problem formulation may vary depending on the interpretation of the parameters, usually the cost or time of the projects.

In a special case, if $K = L$, then $\sum_{i=1}^{L} u_{i,j} = 1$, *j*=1..*K*. This task is called a basic or linear assignment problem. It is one of the fundamental combinatorial optimization problems.

The following methods can be used to solve assignment problems: linear programming (Hungarian algorithm, simplex method), integer programming (Branch-and-Bound method, Cutting-plane method) (Bowman et al., 2002; Flesch et al., 2003; Neogy et al., 2020; Torabi Yeganeh & Zegordi, 2020), ant colony (Dowsland & Thompson, 2008; Wang et al., 2009), genetic algorithm (Dey et al., 2019; Sahu & Tapadar, 2006), graphs colouring (Gozhyj et al., 2019; Monasson, 2004; Zais & Laguna, 2016), artificial neural networks (Philipsen & Stok, 1991; Zhu & Yang, 2006), heuristic methods (Wu & Sweeting, 1994).

The practical use of the assignment problem is limited. In practice, to perform a single work, as a rule, it is usually needed to use several different resources of various types at the same time, i.e. $1 < \sum_{j=1}^{K} u_{i,j} \le K$. If the same resources are needed for other works, then $0 \le \sum_{i=1}^{L} u_{i,j} \le L$ for $K \ge L$. In this case, the matrix $u = \left[u_{i,j} \right]$ is considered to be given, and there is a need to proceed to the schedule formation of the same resources to perform different works. The solution to this problem is to determine the work schedule so that the same resources can be used only sequentially. As shown by different researchers (Chartrand & Zhang, 2008; Saoub, 2021), this task can be conveyed to the graph vertices colouring task.

When having an undirected graph with vertices representing works (tasks) and edges representing resources to perform them, then the graph edges connect only those works that require the same resource. For each resource j=1..K, there is a need to construct the graph clique, linking between the vertices, for which $u_{i,j} = 1$, i=1..L.

As a result, there is an undirected graph $G = (V, E)$ with a finite set of vertices V and edges set E. If several identical resources are needed for two or more works, then there is a multigraph.

Colouring is a display $g : V \to X$, where $X = (x_1, x_2, ..., x_N)$ ($N \le L$) is a colour palette. The colouring is correct if $g(k) \ne g(l)$ for each edge presented in the graph, i.e. $\forall e_{k,l} \in E$, $k, l \in V$. From a coloured graph, a work schedule over time can be obtained. Works whose vertices are painted in the same colour can be done simultaneously and painted sequentially with different colours.

A large-order graph colouring $|V|$ with a limited number of colours N is considered as an NP-complex problem that cannot be solved by a complete search of options in a reasonable polynomial time.

To speed up the search, the method of backtracking (search with return) can be used (Monasson, 2004). This method excludes from consideration a significant number of options in one test, building a decision tree and bypassing it in depth. Although this method is classified as a metaheuristic, it is guaranteed to find all solutions to a finite discrete problem in a limited time.

It is proposed to denote the graph vertices with Latin letters and the colours with consecutive integers from 1 to n. First, the vertex $v(a)$ is painted in colour 1. If the vertex $v(b)$ is not adjacent to $v(a)$, it is painted in colour 1; otherwise – in colour 2. Next it is suggested to consider the vertex $v(c)$ for which an attempt is made to paint in the smallest number i of colours n. If this is not possible, then select the next number of permissible colours. When a vertex that cannot be coloured with any of n the colours is reached, the last coloured vertex is returned, change its colour to the next possible colour from the ordered colour list, and try to colour it. If this is not possible, then there is a return to the previous vertex. The process continues in the same way until the correct colouring of the graph is achieved, or it turns out that the graph cannot be coloured in n colours.

The mathematical model for the graph colouring task can be formulated as a task of 0 - 1 integer programming (Kay & Christofides, 1976):

- objective function, which indicates that the graph should be painted with a minimum number of colours: $z = \sum_{j=1}^{Q} \sum_{i=1}^{L} w_j c_{i,j} \to \min_c$.
- limitations system: $\sum_{j=1}^{Q} c_{i,j} = 1 \;\; \forall i = 1..L$ – each vertex can be painted with only one colour, $M \cdot (1 - c_{i,j}) - \sum_{k=1}^{L} a_{i,k} c_{k,j} L \ge 0 \;\; \forall i = 1..L, \forall j = 1..Q$ – each pair of adjacent vertices does not have the same colour.

The matrix of adjacencies of the graph vertices has the following form: $\left[a_{i,j} \,|\, a_{i,j} \in \{0,1\}, i = 1..L, j = 1..L \right]$, where L is the number of the graph vertices, Q is the number of colours; $\left[c_{i,j} \,|\, c_{i,j} \in \{0,1\}, i = 1..L, j = 1..Q \right]$ is the matrix of coloured vertices of the graph (where $c_{i,j} = 1$, if the vertex v_i has a colour j);

$\left(w_j \mid w_{j+1} > L \cdot w_j, w_1 = 1, j = 1..Q\right)$ is the vector of positive weights of colours; M is a large positive number and $M > L$.

For large orders of graphs (numbers of vertices) $|V|$, the determination of the optimal solution of the problem by integer programming methods may not give a satisfactory result due to the large dimension of the matrices. Therefore, it is recommended to use other methods with the polynomial time of solving task, for example, based on optimization of search options, “soft” calculations or heuristic assumptions. The graphs colouring methods overview can be found in the different articles (Denysenko, 2019; Hardy et al., 2018; Lim & Lee, 2020; Shimizu & Mori, 2022; Thevenin et al., 2017; Zhang et al., 2021). The following methods of graphs colouring are used to solve various practical problems:

1) Dynamic programming (De Lima & Carmo, 2018);
2) Greedy algorithm (Gupta & Singh, 2020);
3) Genetic algorithm (Dey et al., 2019);
4) Artificial neural networks (ANNs) (Philipsen & Stok, 1991);
5) Swarm intelligence algorithms (SIAs) or the emergent collective behaviour of groups of animals based on Particle Swarm Optimization (PSO), for example, Artificial Bee Colony, Bat Swarm Optimization, Salp swarm algorithm (SSA) or Salp Swarm Optimization (SSO) algorithm (Meraihi et al., 2019);
6) Ant colony optimization (ACO) algorithms (Dowsland & Thompson, 2008);
7) Multiagent Graph Colouring method (Blum & Rosenschein, 2008);
8) Game theory (Kravets et al., 2019, 2021; Panagopoulou & Spirakis, 2008).

As researchers (Frieze & Karoński, 2015; Raigorodskii, 2016; Raigorodskii, 2017; Raigorodskii & Karas 2022; Zhukovskii & Raigorodskii, 2015) emphasize, the colouring problem is much more complicated for time random graphs $G(t) = (V(t), E(t))$, the edges are marked by the probabilities of their belonging to the graph. Then at each moment t=1,2,..., the graph appears as one of the possible realizations.

The study of random graphs is mainly related to obtaining probabilistic asymptotic estimates of its parameters, including the chromatic number. Information on effective methods for random graph colouring is insufficiently covered in scientific papers.

Deterministic methods for random graph colouring are unproductive. It is necessary to use or develop multi-step refinement methods with elements of self-learning, built based on “soft” calculations or various heuristics. The work of such methods should be aimed at improving the chromatic picture to achieve the correct colouring of the graph in the asymptotic of time. Modifications of methods 1 – 8 can be used for random graph colouring. Special attention should be paid to the development of stochastic variants of the implementation of these methods.

To solve the random graph colouring problem, that is built based on the model of the projects ontological support, we have proposed to use the stochastic game method, which has the properties of self-learning and adaptation in conditions of uncertainty.

3. Game-Theoretical Formulation

It is necessary to organize the implementation of L projects $\Pi = \{\Pi_1, \Pi_2, \ldots, \Pi_L\}$, and each of them is determined by the competencies necessary for its implementation

$\Pi_i = \{O_1, O_2, ..., O_r\}$ in the form of a set of ontologies. Each ontology formally describes the knowledge in a particular problem area required to complete the project (Keet, 2018).

For ontological support of projects, it is necessary to involve qualified specialists in the necessary fields of knowledge. The information model of project executors will be referred to as agents using multi-agent terminology. It is also assumed that the labour market offers a variety of agents $A = \{A_1, A_2, ..., A_K\}$, $K \geq L$, who can be involved in the implementation of projects. Each agent A_k is defined by a set of ontologies $A_k = \{O_{k,1}, O_{k,2}, ..., O_{k,s}\}$, k=1..K, which describe his abilities in one or more areas of knowledge. Agent ontologies can have a non-empty cross-section $A_i \cap A_j \neq \varnothing$, i.e. agents partially have the same abilities. It is assumed that the aggregate ontological knowledge of the agents is sufficient to perform all projects.

To carry out projects, it is necessary to form a set of teams $\Gamma = \{\Gamma_1, \Gamma_2, ..., \Gamma_L\}$, each of which is an organized group of agents $\Gamma_i = \{A_{i,1}, A_{i,2}, ..., A_{i,g}\}$, i=1..L, where $\bigcup_{i=1..L} \Gamma_i = A$.

A necessary condition for the successful implementation of the project is its full ontological support by a team of agents. The agent team capabilities should cover the competencies required to carry out the project:

$\bigcup_{\forall A_k^i \in \Gamma_i} A_k^i \supseteq \Pi_i$, i=1..L.

The game approach to covering the necessary ontologies of projects with existing ontologies of agents is considered in the work of (Burov et al., 2019; Kravets et al., 2019, 2021).

The selection of agent teams is performed by project managers independently of each other. Then, in a limited number of qualified professionals, some agents may be involved in various projects, i.e. $\Gamma_i \cap \Gamma_j \neq \varnothing$.

It is suggested to assume that the execution of each project cannot be interrupted, and the executors cannot move from one project to another until it is fully completed. It raises the task of determining the timing of projects in time. Similar to the task formulation of the equipment matching to perform certain works, it is assumed that the execution time of each project is the same. Then the problem of planning the sequence of projects can be reduced to an undirected graph colouring problem $G = (\Gamma, E)$, where Γ is a finite set of vertices, E is the edges set. In this graph, the vertices are marked by sets of agent commands involved in the implementation of the project Γ_i, i=1..L, and the edges connect those vertices of the project graph that contain the same agents:

$E_i = \left\{ e_{i,j} \mid \chi\left(\Gamma_i \cap \Gamma_j \neq \varnothing\right) \right\}$,

where $\chi() \in \{0,1\}$ is the event indicator function. The value $\chi() = 1$ indicates the presence of the corresponding edge $e_{i,j}$ in the graph. If the cardinality of the agents' sets intersection is greater than 1, i.e. $|\Gamma_i \cap \Gamma_j| > 1$, then a multigraph is considered.

Then the vertices of the project graph should be coloured so that the vertices connected by the edges $e_{i,j}$, i,j=1..L, should have different colours. Then by such colouring, it is possible to define the sequence of projects performance. Those whose vertices have the same colours can be done simultaneously and those with various colours - sequentially.

However, at the time of team formation, there is some uncertainty about agents' involvement in project implementation. This uncertainty is defined by the probabilities $q_{i,k}$ of agreements between the manager of *i*-th project and the *k*-th agent. These probabilities include several risk factors for non-implementation of the project, such as refusal to cover the project with the necessary ontologies due to insufficient capabilities of agents, inability to implement the project due to the contractors' health, and unpredictable external factors and more. Project managers can define values $q_{i,k}$ as the confidence degree that the agent will participate in the project, or by the agent as his propensity to implement a project, or comprehensively - by both parties. Simplified, it can be assumed that $q_{i,k} = q_k$ – is the probability of participation of the *k*-th agent in the implementation of any project. Then $1 - q_k$ – is the probability of refusal of the *k*-th agent. It is considered that agent refusals are renewable.

Taking into consideration the stochastic nature of agents, a random graph of relationships between projects should be considered instead of a deterministic one. The renewable refusal of one of the agents, due to which there is an edge in the graph between the vertices-projects, leads to a temporary loss of this edge. Therefore, instead of a given deterministic graph at certain intervals, its random realizations in the form of all possible subgraphs will be observed.

Determining methods cannot be used for random graph colouring because, at each step of the game, the implementations of the graph change and the values of the current losses are random variables. Adaptive stochastic methods that can be adapted to random changes in the structure of the graph should therefore be used. To do this, we have proposed to use a method of the multistage stochastic game.

Stochastic games, presented by works (Flesch et al., 2003; Hartley & Thuijsman, 1994; Huang & Ma, 2016; Neyman & Sorin, 2003; Thuijsman et al., 1991; Thuijsman & Vrieze, 1991, 1993, 1999), model the dynamic interactions in which the environment is changing in response to player's behaviour.

In the stochastic game the agents choose actions simultaneously. The state space, the action space, and the time instants of action selection are considered as discrete.

The formal model of Markov stochastic game (Chen, 2019; Huang & Ma, 2016; Puterman, 2008; Thuijsman et al., 1991; Thuijsman & Vrieze, 1987) is given by the tuple:

$$\Im = \langle I, S, A, P, R \rangle ,$$

where *I* is the finite set of players; the number of players is equal to the cardinal number of this set: $L = |I|$; *S* is the set of possible game states; $A(s) = \times_{i \in I} A_i(s)$ is the set of combined actions of players in state *s*, which is determined by the *L*-ary Cartesian (direct) product of possible actions $A_i(s)$ of players $\forall i \in I$; $P(s') = P(s,a)$ is the function of transition probabilities between game states, which reflects the probability of transition from state $s \in S$ to state $s' \in S$ when performing the action $a \in A(s)$; $R(i,s') = R(i,s,a) \forall i \in I$ is the reward function that specifies the expected reward of each player when transitioning from state $s \in S$ to state $s' \in S$ when performing the action $a \in A(s)$.

When defining the stochastic game, the discount factor $\gamma \in [0,1]$ can be additionally set, which indicates how much players take future rewards into account in their strategies.

If the sets *S* or $A_i(s)$ are uncountable, then they are supplemented by the σ-algebra of measurable sets (Huang & Ma, 2016; Shiryaev, 1996, 1996a).

Next, we have considered the simplified formulation of the stochastic game, when there is only one game state $|S|=1$, and the sets of actions A_i $\forall i \in I$ of all players are finite (Fudenberg & Tirole, 1991, 1991a; Shoham & Leyton-Brown, 2008). Then such a stochastic game can be described by the tuple:

$$\Im = \langle I, A, P, R \rangle,$$

where *I* still denotes the finite set of players; $A = \underset{i \in I}{\times} A_i$ is the set of joint actions of players combined on the *L*-ary Cartesian product of actions (pure strategies) of players; $P = P(A)$ is the distribution of probabilities in the set *A*, the elements of which determine the probability that in the result of the independent choice of actions $a_i \in A_i$ $\forall i \in I$ of players will form a combined action $a = (a_1, a_2, ..., a_L) \in A$. So, if $p_i = \{ p_i(a_j) | \forall a_j \in A_i \}$ is the set of probabilities of choosing pure strategies (or, otherwise, the mixed strategy) of *i*-th player $\forall i \in I$, where $\sum_{a_j \in A_i} p_i(a_j) = 1$, then $P(A) = \underset{i \in I}{\times} p_i$; $R = \{ R(i, A) | \forall i \in I \}$ is the reward function that gives each player's expected reward in conditions when all players have taken the combined action $a \in A$.

To formulate the game problem of the stochastic colouring of the graph as the main problem of assigning the staff to the project implementation, with each vertex (project) of the graph, there is a need to assign the player, whose pure strategies determine the numbered colours palette $X_i = \{x_{i,1}, x_{i,2}, ..., x_{i,N}\}$, where $x_{i,j}$ is the colour number; *N* is the number of elements in the colour palette, which is limited to the value $N = L$ required for a fully connected graph colouring.

Each vertex of the graph represents one of the projects, and each project contains the set of agents whose set of ontologies (capabilities) cover the set of ontologies required for the implementation of this project (competencies). The correctly coloured graph will allow to determine the sequence of project execution.

The choice of pure strategies $x_i(t) \in X_i$ is made by players randomly and independently at times *t*=1,2,… . Due to player refusals, a random graph implementation $G(t) = (\Gamma(t), E(t)) \subseteq G$ is determined at each step of the game, in which only the mark-up of vertices $\Gamma(t) = \{\Gamma_i(t) | i = 1..L\}$ by sets of agents $\Gamma_i(t) = \{A_{i,1}, A_{i,2}, ..., A_{i,g}\} \setminus \{A_{i,k}(t) | k = 1..l, l \le g\}$ and the corresponding connections between vertices $E(t)$ change. The expression $\{A_{i,k}(t)\}$ denotes the set of agents that refused at the moment *t*. The number of vertices *L* and the corresponding projects Π_i remains unchanged. Players do not have information about the current implementation of the graph as a whole. Each player knows only their local subgraph – a set of adjacent vertices connected by edges to the player-controlled vertex of the graph.

Let $K_i^{loc}(t) = |E_i(t)|$ be the number of edges of the *i*-th vertex of a random graph at a time *t*. Then the *i*-th player takes part in the game if it corresponds to an uninsulated vertex, i.e. $K_i^{loc}(t) \ge 1$.

When all players choose pure strategies, each of them calculates the value of the current loss as the average number of identical colours of adjacent vertices of the graph:

$$\xi_i(t) = \left(K_i^{loc}(t)\right)^{-1} \sum_{j \in D_i(t)} \chi\left(x_i(t) = x_j(t)\right), \tag{1}$$

where $\chi() \in \{0,1\}$ is the indicator function of the event; $D_i(t) = \{\underset{j}{index}(e_{i,j}(t))\}$ is the set of numbers of adjacent vertices for the *i*-th vertex of a random graph. Adjacent to the vertex *i* is the vertices of a random graph directly related to it at a time *t*, $|D_i(t)| = K_i^{loc}(t)$.

Players evaluate their actions during the game using the current values of the functions of average losses (or losses):

$$\Xi_i(t) = t^{-1} \sum_{\tau=1}^{n} \xi_i(\tau), \; i=1..L. \tag{2}$$

The course of the stochastic game as a whole can be controlled using the system function of average losses:

$$\Xi(t) = L^{-1} \sum_{i=1}^{L} \Xi_i(t). \tag{3}$$

The behaviour strategy of each player should be aimed at minimizing their functions of average losses (to minimize the colour matches of adjacent vertices of the graph):

$$\overline{\lim_{t \to \infty}} \, \Xi_i(t) \to \min_{\{x_i(t)\}}, \; i=1..L. \tag{4}$$

The multicriteria optimization problem solutions (4) should be sought in points sets of collective optimality such as Slater, Nash, Pareto or others (Nash, 1950; Romanuke, 2022, 2021; Ungureanu, 2018,2018a, 2018b, 2018c, 2018d, 2018f). Most often, tasks without the exchange of current information about the strategies, states and losses of players (or with the minimum required exchange) use the Nash equilibrium criterion (Romanuke, 2022, 2021). At the Nash equilibrium point, it is not beneficial for each player to change their strategy if all other players adhere to the equilibrium point:

$$\overline{\lim_{t \to \infty}} \left[\Xi_i(t, \{x^{D_i}(t)\}) - \Xi_i(t, \{y^{D_i}(t)\}) \right] \le 0, \; i=1..L. \tag{5}$$

The following notation in (5) is used: D_i is the local set of neighboring players whose strategies affect the amount of current losses (1) of the *i*-th player (the composition of this set is determined by the numbers of the vertices of the graph, which are adjacent to the vertex *i*, which is controlled by the *i*-th player); $X^{D_i} = \underset{j \in D_i}{\times} X_j$ is the space of combined pure strategies of players from the local set D_i; $X^{D_i} \subseteq X$; $X = \underset{i=1..L}{\times} X_i$ is the space of combined pure strategies of the whole set of players; the symbol $\times$ is the Cartesian product operation; $x^{D_i} \in X^{D_i}$; $y^{D_i} = x^{D_i} \setminus x_i + y_i \in X^{D_i}$ is the combined pure strategy of neighbouring players from the subset D_i after replacing the pure strategy of the *i*-th player; y_i is the pure strategy of the *i*-th player deviated from the equilibrium point; $x_i, y_i \in X_i$.

Therefore, observing random current losses (1), players must learn to choose colours $x_i(t) \in X_i$ from the colour palette X^i. The formed sequence of options $\{x_i(t)\}$ ensures the fulfilment of goal (4) in the asymptotic of time $t \to \infty$. In practice, the number of steps of a stochastic game is limited by some maximum value or the graph colouring correct achievement.

4. Methods of solving the stochastic game

Exact methods for solving stochastic game problems cannot be applied due to the randomness of players' actions and a priori uncertainty of payment functions. Since in stochastic games the players make decisions under conditions of randomness and uncertainty, there is a need to find an optimal strategy based on statistical data processing.

To solve the stochastic game, there is a need to apply approximate (recurrent) methods, which are used when exact methods of solving are unacceptable due to great complexity or insufficient data. Recurrent methods make it possible to solve stochastic games approximately, optimizing various parameters and strategies of players.

To generate sequences of strategies $\{x_i(t) \mid i = 1..L\}$, t=1,2,... that ensure the fulfilment of criteria (4), there is a need to construct a probabilistic mechanism based on mixed strategies of players $\{p_i(t) \mid i = 1..L\}$. The mixed strategy $p_i(t) = \left(p_{i,1}(t), p_{i,2}(t), \ldots, p_{i,N}(t)\right)$ consists of conditional probabilities of choosing pure strategies:

$$p_{i,j}(t) = P\left\{x_i(t) = x_{i,j} \,\middle|\, u_i(\tau), \xi_i(\tau), \tau = 1, 2, \ldots, t-1\right\},\ j=1..N,$$

where $\{x_i(\tau)\}$ is the history of pure strategies selected by the player with the number i; $\{\xi_i(\tau)\}$ is the background of the losses received.

To form sequences with the desired properties, mixed strategies at each step of the game are changed by a recurrent method (Nazin & Poznyak, 1986, 1987):

$$p_i(t+1) = \pi^N_{\varepsilon(t+1)}\left\{p_i(t) - \gamma(t) R(p_i(t), x_i(t), \xi_i(t))\right\}, \tag{6}$$

where $\pi^N_{\varepsilon_{t+1}}$ is the projector on a single ε -simplex $S^N_\varepsilon \subseteq S^N \subset R^N$ (here the superscript is not an indicator of degree, but indicates the number of measurements of the space of real numbers); $\gamma(t)$ is a monotonically decreasing sequence of non-negative values, which regulates the step size of the method; R is step method; $\varepsilon(t)$ is monotonically decreasing sequence of non-negative quantities, which regulates the rate of expansion of the ε -simplex.

The coordinates of the points of a unit simplex are normalized so that their sum is equal to 1:

$$S^N = \left\{ p \,\middle|\, \sum_{j=1}^{N} p_j = 1;\ p_j \ge 0 \quad (j = 1..N) \right\}.$$

A single ε -simplex is a compact subset of a unit simplex:

$S_\varepsilon^N = \left\{ p_i \mid p_i \in S^N;\ p_{i,j} \ge \varepsilon \quad (j = 1..N) \right\}$, $\varepsilon \in (0, 1/N)$, $p_i(t) \in S_\varepsilon^N$.

The recurrent method (6) should be constructed so that when choosing a strategy $x_{i,j}(t)$, the corresponding probability $p_{i,j}(t)$ decreases in proportion to the magnitude of the current loss $\xi_i(t)$. The other elements of the mixed strategy do not change or increase proportionally $\xi_i(t)$. The method should increase the probability of choosing more successful strategies that are useful for meeting the criteria for minimizing the average losses of players (4). A technique with such properties is called adaptive or self-learning.

After calculating the new values of the vectors of mixed strategies is their design on ε -simplex S_ε^N. The design operator $\pi_{\varepsilon_{t+1}}^N$ satisfies the conditions:

$$\left\| p_i - \pi_\varepsilon^N\{q_i\} \right\| \le \left\| p_i - q_i \right\|;\ \pi_\varepsilon^N\{q_i\} \in S_\varepsilon^N,\ i = 1..L,\ \forall p_i \in S_\varepsilon^N, \forall q_i \in R^N.$$

Projecting on the expandable ε-simplex ensures the condition $p_{i,j}(t) \ge \varepsilon(t)$, $j = 1..N$ which is necessary for the statistical information completeness on the choice of pure strategies is met.

The construction of recurrent methods of the type (6) will be performed using the method of stochastic approximation (Benveniste et al., 1990; Chung, 1954; Kushner & Yin, 1987, 1987a, 1999, 2003; Nazin & Poznyak, 1986, 1987). For this, the *L*-person ($L \ge 2$) non-cooperative deterministic game with $N \ge 2$ strategies (Fudenberg & Tirole, 1991, 1991a; Osborne, 2000, 2000a, 2000b, 2004, 2010; Osborne & Rubinstein, 1994; Neumann & Morgenstern, 2007), closely related to the corresponding stochastic game has been considered. The asymptotic equivalence of the game the L-person deterministic and stochastic game is proved in the works (Nazin & Poznyak, 1986, 1987). The formulation of the deterministic game is the auxiliary step for the construction of recurrent methods for solving the stochastic game.

Assume that the mathematical expectations of random variables $M\{\xi_i(x,t)\} = v_i(x)$ are known for all combined strategies $x \in X = \underset{i=1..L}{\times} X_i$. Then the deterministic *L*-person game is given by the tuple $\Im = \left\langle X_i, p_i, \left[v_i(x^{D_i}) \right] \middle| i = 1..L \right\rangle$, where X_i is pure strategies (colour numbers of graph vertices), $p_i = \left(p_i[1], p_i[2], \dots p_i[N] \right)$ is mixed strategies of players (conditional probabilities of choosing pure strategies), $\left[v_i(x^{D_i}) \right]$ is the array of average losses, the elements of which are addressed by pure strategies of players from the set D_i.

In the deterministic game, the average losses of the v_i players are assumed to be given. In the stochastic game, the average losses of the players are not known a priori. Only their random realizations ξ_i are available for observation.

For the deterministic *L*-person game there is a need to define polylinear functions of average losses, which are mathematical expectations of each players losses when implementing mixed strategies $p_i \in S^N$, *i*=1..*L*:

$$V_i(p^{D_i}) = \sum_{x^{D_i} \in X^{D_i}} v_i(x^{D_i}) \prod_{j \in D_i; x_j \in x^{D_i}} p_j(x_j), \tag{7}$$

where D_i is the local set of neighbouring players; $X^{D_i} = \underset{j\in D_i}{\times} X_j$ is space of combined pure strategies of players from the local set D_i; $x^{D_i} \in X^{D_i}$ is one of the combined local strategies of players; $p^{D_i} \in S^{D_i} = \underset{j\in D_i}{\times} S_j^N$; $p_i \in S^N$.

The goal of the players is to minimize their functions of average loss (7):

$$V_i(p^{D_i}) \to \min_{p_i}, \ i=1..L. \tag{8}$$

According to Nash theorem, for every L-person game there is at least one mixed strategy for each player such as none of the players can reduce the loss (in the case of minimizing the average loss functions) only by changing own strategy in the case when the other players' strategies are fixed (Nash, 1950; Nazin & Poznyak, 1986, 1987; Osborne, 2000, 2000a, 2000b, 2004, 2010; Osborne & Rubinstein, 1994).

At the Nash equilibrium points in mixed strategies, the following condition is fulfilled:

$$V_i(p_*^{D_i}) - V_i(p_*^{D_i\setminus\{i\}}, p_i) \le 0, \ i=1..L,$$

where $V_i(p_*^{D_i})$ is the function of average losses, defined at the Nash point $p_*^{D_i} \in S^{D_i}$ on the local simplex $S^{D_i} = \underset{j\in D_i}{\times} S_j^N$ of combined mixed strategies of players from a set D_i of neighbouring players; $V_i(p_*^{D_i\setminus\{i\}}, p_i)$ is the function of average losses, defined in simplex S^{D_i} for any deviation of the mixed strategy of the i-th player from the Nash point.

If the mixed strategies of all players determine the Nash point $p_*^{D_i}$, then no player can change his or her optimal strategy p_i^*, to any other p_i in such a way as to get a smaller average loss V_i, if the other players stay with their optimal strategies.

According to the theory of stochastic approximation (Nazin & Poznyak, 1986, 1987), to minimize the system of functions (8), the motion vector R of the recurrent method (6) is defined so that its mathematical expectation is the gradient of mean losses function (7):

$$M\{R(p_i(t), x_i(t), \xi_i(t))\} = \nabla_{p_i} V_i(p^{D_i}).$$

Taking into account that

$$\nabla_{p_i} V_i = M\left\{ \frac{\xi_i(t)}{e^{\mathrm{T}}(x_i(t))\, p_i(t)} e(x_i(t)) \,\middle|\, p_i(t) = p_i \right\},$$

where $e(x_i(t))$ is unit vector-indicator of the choice of pure strategy $x_i(t) \in X_i$, $e^{\mathrm{T}}(x_i(t))$ is transposed vector, then a recurrent gradient method for solving the game problem is obtained:

$$p_i(t+1) = \pi_{\varepsilon(t+1)}^N \left\{ p_i(t) - \gamma(t) \frac{\xi_i(t) e(x_i(t))}{e^{\mathrm{T}}(x_i(t))\, p_i(t)} \right\}. \tag{9}$$

Other recurrent methods can be obtained from the complementary slackness condition (Kravets et al., 2020, 2021; Neogy et al., 2018, 2020), which is performed for Nash equilibrium points in completely mixed strategies:

$$\nabla_{p_i} V_i = V_i e_N, \ i=1..L, \tag{10}$$

where e_N is a vector consisting of N units. The complementary slackness condition describes the independence of the average loss functions of players from their mixed strategies at the Nash point. No matter how the mixed strategy $p_i(t)$ of player with the number i (i=1..L) on a unit simplex can be changed, when all other players follow their strategies at the Nash point, the value of the win function V_i remains constant.

Since

$$\nabla_{p_i} V_i - V_i e_N = M\left\{ \xi_i(t)\left[\frac{e(x_i(t))}{e^{\mathrm{T}}(x_i(t))p_i(t)} - e_N \right] \middle| p_i(t) = p_i \right\},$$

by the method of stochastic approximation, the following recurrent method constructed using the complementary slackness condition has been obtained:

$$p_i(t+1) = \pi^N_{\varepsilon(t+1)}\left\{ p_i(t) - \gamma(t)\xi_i(t)\left[\frac{e(x_i(t))}{e^{\mathrm{T}}(x_i(t))p_i(t)} - e_N \right] \right\}. \tag{11}$$

Taking into account the solutions at the boundary of a unit simplex, there is a need to weigh the vector condition (10) by the elements of the vector p_i:

$$diag(p_i)[V_i e_N - \nabla V_i] = 0, \tag{12}$$

where $diag(p_i)$ is a square diagonal matrix of an order N composed of vector elements p_i.

Taking into consideration that

$$diag(p_i)(V_i e_N - \nabla_{p_i} V_i) = M\left\{ \xi_i(t)\left[p_i(t) - e(x_i(t)) \right] \middle| p_i(t) = p_i \right\},$$

by using the method of stochastic approximation, the following recurrent method can be implemented based on weighted complementary slackness condition:

$$p_i(t+1) = \pi^N_{\varepsilon(t+1)}\left\{ p_i(t) - \gamma(t)\xi_i(t)\left[e(x_i(t)) - p_i(t) \right] \right\}. \tag{13}$$

Due to such a dynamic reorganization of mixed strategies based on the processing of current losses, methods (9), (11), and (13) provide an adaptive choice of pure strategies over time.

The parameters $\gamma(t)$ and $\varepsilon(t)$ are monotonically descending sequences of non-negative quantities and are used to control the convergence of recurrent methods. These parameters can be calculated as follows:

$$\gamma(t) = \gamma(0)t^{-\alpha},\ \varepsilon(t) = \varepsilon(0)t^{-\beta}, \tag{14}$$

where $\gamma_0, \alpha, \beta > 0$; $\varepsilon(0) \in (0, N^{-1})$.

The convergence of mixed strategies $p_i(t)$, i=1..L to the optimal values with probability 1 or root-mean-square is determined by the ratios of the parameters γ_t and ε_t which must satisfy the fundamental conditions of stochastic approximation (Benveniste et al., 1990; Chung, 1954; Kiefer & Wolfowitz, 1952; Kushner & Yin, 1987, 1987a, 1999, 2003; Nazin & Poznyak, 1986, 1987).

The efficiency (in the sense of fulfilling criteria (4)) of recurrent algorithms is ensured by fulfilling the condition of pseudo-gradient of the vector R for the Lyapunov function $\Delta(p)$ (Nazin & Poznyak, 1986, 1987):

$\left\langle M\{R\{x_i(t), p_i(t), \xi_i(t) \mid p_i(t) = p_i\}, \nabla_{p_i}(\Delta(p))\right\rangle \geq 0$,

where $\langle \cdot,\cdot \rangle$ is the scalar product of vectors in Euclidean space; $p_i \in S^N$; $p \in S = \underset{i=1..L}{\times} S_i^N$.

The Lyapunov function Δ must be differentiated by p_i, i=1..L, must have zeros at the points of asymptotic optimality $\Delta(p^*) = 0$; be positive $\Delta(p) > 0$ on a single combined simplex $\forall p \in S$; $p \neq p^*$. To optimize the function of average wins on a system of unit simplexes, it is assumed that $\Delta(t) = \sum_{i=1}^{L} \left\| p_i(t) - p_i^*(t) \right\|^2$, where $p_i^*(t)$ is the asymptotically optimal solution in mixed strategies for the i-th player.

For the considered recurrent methods, the Lyapunov function $\Delta(t)$ can be defined as the error of the complementary slackness condition (the square of the Euclidean norm of the mixed strategies difference):

$$\Delta(t) = \sum_{i=1}^{L} \left\| p_i(t) - w_i(t) \right\|^2,$$

where $w_i(t) = diag(p_i(t))(\nabla_{p_i(t)} V_i(t)) / (V_i(t))$ is the weighted mixed strategy of the i-th player, calculated from the complementary slackness condition; $p_i, w_i \in S^N$.

Mean squared convergence rate of recurrent methods can be estimated by the asymptotic method of Chung's moments (Chung, 1954; Nazin & Poznyak, 1986, 1987):

$$\overline{\lim_{n\to\infty}} n^{\theta} M \left\{\Delta(t)\right\} \leq \vartheta, \tag{15}$$

where θ is order of the root-mean-square convergence rate, ϑ is the value of the rate of convergence. Higher values θ and lower ϑ correspond to the higher rate of convergence of the game method.

In sign-positive environments for which $V_i(p^{D_i}) > 0$ on the system of unit simplexes, the theoretical order of the root-mean-square convergence rate of methods (9) and (11) is equal $\theta = \min(1 + \beta - \alpha, \alpha - \beta)$ with parameter constraints $\alpha \in (0,1]$; $0 < \beta < \alpha$. The theoretical order of the root-mean-square convergence rate of method (13) is equal $\theta = \min(1 + \beta - \alpha, \alpha)$ with the constraints $\alpha \in (0,1]$; $\beta > 0$ (Kravets et al., 2020, 2021).

The obtained values of the rate parameters of the recurrent methods convergence are approximate, since they are determined on the basis of upper estimates of random processes. It is recommended to specify these parameters during a computer experiment.

The choice of pure strategies (colours of vertices) $x_{i,k}(t)$, i=1..L is carried out by players randomly based on mixed strategies $p_i(t) = \left(p_{i,1}(t), p_{i,2}(t), \ldots, p_{i,N}(t)\right)$:

$$k = \arg\left(\min_{k=1..N} \sum_{j=1}^{k} p_{i,j}(t) > \omega \right) \in \{1..N\}, \tag{16}$$

where $\omega \in [0, 1]$ is a uniformly distributed real random number.

The stochastic game begins with untrained mixed strategies with element values $p_{i,j}(0) = 1/N$ where j=1..N. During the next moments, the dynamics of the vectors of mixed strategies are determined by one of the Markov recurrent methods. Recurrent methods (9), (11), (13) provide adaptation of players' strategies both to changes of random graph realization and calculated (on their basis) in advance unknown current losses.

Therefore, at times t=1,2,..., each player based on the mixed strategy $p_i(t)$ chooses a pure strategy $x_i(t)$ (16) and by the time $t+1$ receives the current loss $\xi_i(t)$ (1), and then calculates the mixed strategy $p_i(t+1)$ according to one of the methods (9), (11), (13).

The vertices colours of the random graph are defined as the mathematical expectation of the possible numbers of colours, rounded to an integer number, calculated for the last step of the stochastic game:

$$\overline{x}_i(t) = \text{int}\left(\sum_{i=1}^{N} p_i(t) x_i(t)\right), \; i=1..L. \tag{17}$$

4.1. Stochastic game algorithm

The stochastic game algorithm is described as following. Steps 1 – 2 determine the initialization of data and perform preparatory actions, and steps 3 – 11 implement a stochastic game for random graph vertices colouring.

Step 1. Set the initial values of the parameters:

- t=0 is the initial time;
- L is the projects number (number of graph vertices, number of agent teams, number of players);
- K is the total number of agents that can be involved in all projects.
- N=L is the number of pure strategies of players (the number of colours of the paint palette);
- $\Omega = \{O_1, O_2, ..., O_m\}$ is the ontologies dictionary;
- $\Pi_i = \{O_{i,1}, O_{i,2}, ..., O_{i,r}\} \subseteq \Omega$, i=1..L is set of ontological knowledge or competencies required for project implementation;
- $A_k = \{O_{k,1}, O_{k,2}, ..., O_{k,s}\} \subseteq \Omega$, k=1..K is set of ontologies that determine the abilities of agents;
- $q_{i,k}$, i=1..L, k=1..K is the probabilities of agents' participation in project implementation;
- $U_i = \{u_{i,1}, u_{i,2}, ..., u_{i,N}\}$, i=1..L is vectors of pure strategies of players;
- $p_i(0) = \left((1/N)_j \,\middle|\, j = 1..N\right)$, i=1..L is the initial values of mixed player strategies;
- $\gamma > 0$ is the learning step parameter;
- $\alpha \in (0,1]$ is the coefficient of the order of the learning step;
- ε is parameter ε-simplex;
- $\beta > 0$ is the coefficient of the order of expansion ε-simplex;
- $\lambda \in [0,1]$ is the weighting factor;
- $t_{\max}$ is the maximum number of method steps.

Step 2. Perform preparatory actions:

2.1. Cover projects with ontologies, involving relevant agents in the implementation of projects.

2.2. Construct a graph whose vertices denote projects (agent teams) and the edges connect those vertices (projects) for which the same agents are involved. Form the initial matrix of adjacencies of the vertices of the graph.

2.3. Associate players with each vertex of the graph who choose the current colours of the graph vertices.

Step 3. Determine the current composition of the agent teams involved in implementing projects with probability $q_{i,k}$ and perform a new mark-up of the graph vertices.

Step 4. Determine the current matrix of adjacencies of the vertices of the graph.

Step 5. Choose pure strategies (colours of graph vertices) $x_i(t) \in X_i$ of players i=1..L according to (16).

Step 6. Calculate the value of current losses $\xi_i(t)$, i=1..L according to (1).

Step 7. Calculate the parameters $\gamma(t)$ and $\varepsilon(t)$ according to (14).

Step 8. Calculate the elements of the vectors of mixed strategies $p_i(t)$, i=1..L according to (13).

Step 9. Calculate the current values of the functions of the average losses $\Xi_i(t)$ (2) of each player and, on their basis, calculate the system function of the average losses $\Xi(t)$ (3) of the stochastic game for the graph colouring.

Step 10. Set the next time $t := t+1$.

Step 11. If $t < t_{\max}$, then go to step 3, otherwise – to step 12.

Step 12. Calculate the average colour values $\bar{x}_i(t)$, i=1..L for the vertices of the graph according to (17). End of the game.

4.2. Test example

For the test example, the following values were adopted:

1) library of ontologies $\Omega = \{O_1, O_2, O_3, O_4, O_5\}$;
2) competencies required for the implementation of projects $\Pi_1 = \{O_1, O_3, O_4\}$, $\Pi_2 = \{O_2, O_3, O_5\}$, $\Pi_3 = \{O_1, O_4, O_5\}$, $\Pi_4 = \{O_1, O_3, O_5\}$;
3) the ability of agents $A_1 = \{O_1, O_4\}$, $A_2 = \{O_2, O_3\}$, $A_3 = \{O_1, O_5\}$, $A_4 = \{O_3, O_4\}$, who can be involved in the implementation of projects.

Based on these data, the following coverage of projects by agents is possible: $\Pi_1 = \{A_1, A_2\}$, $\Pi_2 = \{A_2, A_3\}$, $\Pi_3 = \{A_1, A_3\}$, $\Pi_4 = \{A_3, A_4\}$. Indeed, the following ontology coverage ratios are valid for given projects:

$$\Pi_1 : A_1 \cup A_2 = \{O_1, O_2, O_3, O_4\} \supseteq \{O_1, O_3, O_4\};$$

$$\Pi_2 : A_2 \cup A_3 = \{O_1, O_2, O_3, O_5\} \supseteq \{O_2, O_3, O_5\};$$

$$\Pi_3 : A_1 \cup A_3 = \{O_1, O_4, O_5\} \supseteq \{O_1, O_4, O_5\};$$

$$\Pi_4 : A_3 \cup A_4 = \{O_1, O_3, O_4, O_5\} \supseteq \{O_1, O_3, O_5\}.$$

Figure 1 shows a graph whose vertices are projects Π_i, and the edges are the connections between those projects for which it is planned to involve the same agents. At the graph vertex there is an agents list $A_k, k = 1..K_i$ involved in the implementation of projects. Each vertex (project) is associated with a player who makes moves in the stochastic game to select the current vertex colour, depending on the colours connected by the edges of the graph adjacent vertices.

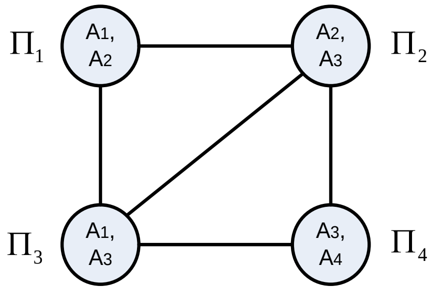

Figure 1. Graph of projects dependencies on executors

Agent refusals result in a change of the composition of project executives and a corresponding change in the relationships between the graph vertices. As a result, a random graph instead of a deterministic graph has been obtained. Several possible implementations are shown in Figure 2.

Figure 2a shows the implementation of a random graph for the case when agent A_1 of project Π_1 failed. The same structure of the graph will be in case of refusal of agent A_1 of project Π_3, or refusal of agent A_1 for both projects Π_1 and Π_3. The result is a loss of connection between projects Π_1 and Π_3.

The refusal of agent A_2 of project Π_2 will implement the random graph shown in Figure 2b. A similar result is obtained in the case of the refusal of agent A_2 of project Π_1, or refusal of agent A_2 of projects Π_1 and Π_2.

The case of refusal of agent A_3, involved in the implementation of projects Π_2, Π_3 and Π_4, is shown in Figure 2c. This figure shows that agent refusals can disrupt the connectivity of the project graph. Players, controlling the states of isolated vertices, are temporarily out of the game.

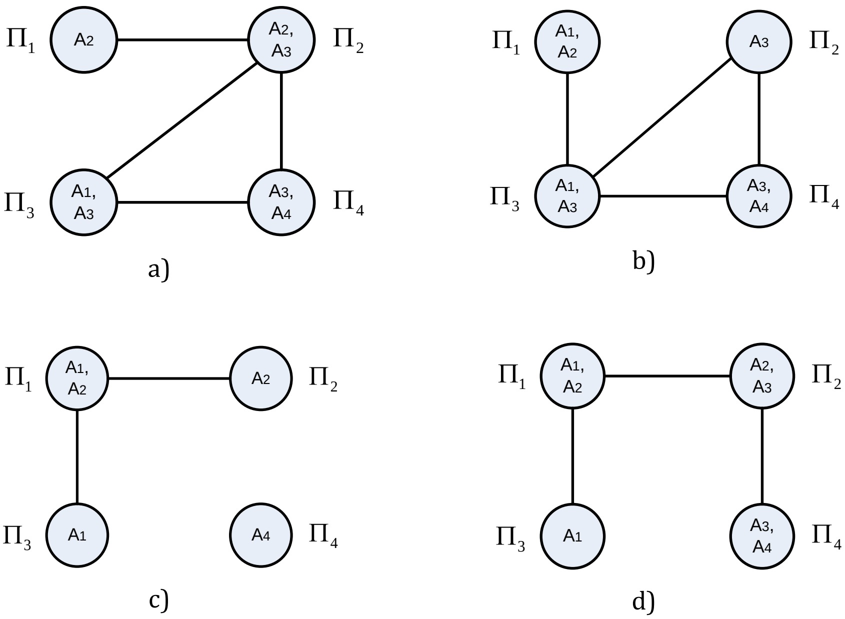

Figure 2. Implementation of a random graph

The implementation of the graph shown in Figure 2d was obtained as a result of the refusal of agent A_3 of project Π_3. For a given random graph, it is necessary to determine the sequence of projects implementation, for which the possible participation of each agent is consistent over time.

4.3. Computer simulation results

To solve the problem of developing a diagram of the project execution sequence, considering the executive agents ontologies, software tools for modelling a stochastic game for colouring a stochastic graph have been developed. The software implementation of the stochastic game is made in the C++ language in the Visual C++ programming environment.

The problem is solved by the stochastic game method (13) for random graph colouring. The following values of the parameters of the stochastic game have been adopted: $L = 4$; $N = L$; $K = 4$; $\gamma_0 = 1$; $\alpha = 0.01$; $\varepsilon_0 = 0.999\, N^{-1}$; $\beta = 2$.

The influence of probabilities $q_k = q$, i=1..K of the agent participation in implementing projects on the convergence of the stochastic game for the problem of colouring random graphs on a logarithmic scale, is shown in Figure 3.

The parameter θ of the order of convergence rate (15) of the game method is determined by the tangent of the angle formed by the linear approximation of the graph of the system function of average losses and the time axis. As can be seen in Figure 3, the average order of the rate of convergence of the game method is close to 1. It practically does not change for different probabilities of participation of agents in the implementation of projects. As the value of these probabilities decreases, the learning time of the stochastic game only increases.

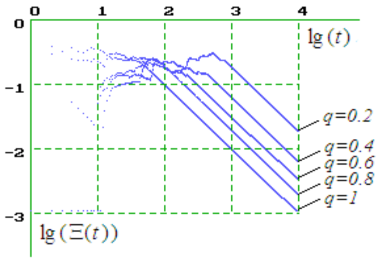

Figure 3. Dependence of average losses system function on the probabilities of agents participation in the implementation of projects

Figure 4 shows the approximate exponential dependence of the average number of steps $\bar{t}$ required for a random graph colouring on the probabilities $q_k = q$, i=1..K (the probabilities of the agent's participation in the game).

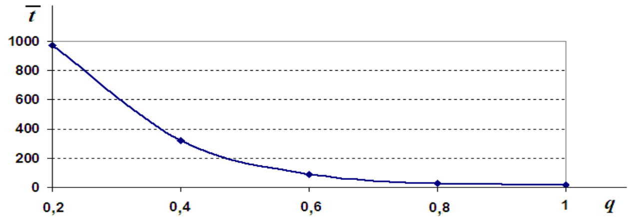

Figure 4. The average number of steps in learning a stochastic game

The increase in the probability of agents involvement in the project implementation ($q \to 1$) leads to the increase in the similarity of random graph $G(t)$ implementations to a given deterministic graph G: $\delta(t) = \mathfrak{M}(G(t), G) \to 0$, where $\delta(t) \in [0,1]$ is the degree of proximity of graphs at moments t=1,2,... (Belova & Pobizhenko, 2017). The result is a reduction in the number of steps in learning a stochastic game. For reliable agents (q=1) it is necessary 10 – 20 steps of the stochastic game for graph colouring correct formation with Fig. 1.

The solution to the stochastic game is shown in Figure 5. This is a coloured graph obtained for the probabilities $q_k = q = 0.8$, i=1..K of the agent's participation in the project's implementation.

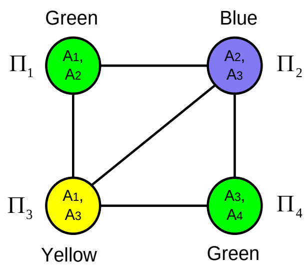

Figure 5. Painted graph of projects

Projects corresponding to vertices of the same colour can be executed simultaneously (in parallel). Two sequences of projects from six possible are shown in Figure 6.

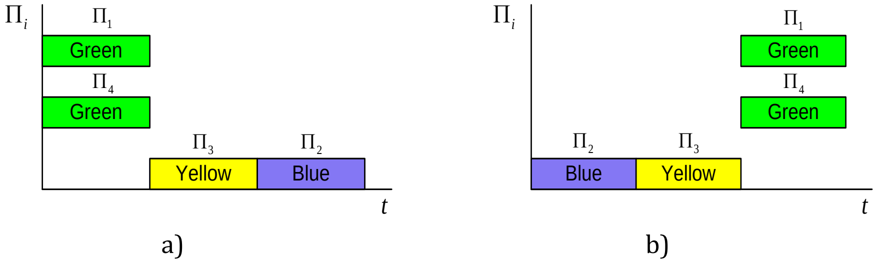

Figure 6. Sequences of project implementation

As shown in Figure 6a and Figure 6b, projects Π_1 and Π_4 can be executed simultaneously and projects Π_2 and Π_3 – in series with these parallel projects. The graph colouring algorithm used determines only the sequence of execution of projects without considering the different duration of the projects.

In conditions of a priori uncertainty of the projects random graph, it is important to maximize the probabilities of project coverage. The probabilities of project coverage determine the chance of its implementation, taking into account the refusals of executing agents. The scheme for calculating such probabilities is presented below.

Let be $Z_i = 2^{K_i}$ is the total space of the combined states of the agents involved in the execution of the *i*-th project (*i*=1..*L*), where K_i is the number of such agents. The value of the state $s_{i,j} = 1$ signals the participation of the $j \Leftrightarrow k$ -th agent, and the value $s_{i,j} = 0$ is of his refusal to participate in the project Π_i. Here the operation $\Leftrightarrow$ performs a mutually unique display of sequential numbers $j = 1..K_i$ of project Π_i agents A_j and real numbers of agents A_k. The probability of project coverage $p_{\text{cov}}(\Pi_i)$ is then defined as the sum of the probabilities of those combined states $(s_{i,1}s_{i,2}...s_{i,K_i})$ for which the individual states of the agents are equal to 1, and whose united ontologies cover the given project.

For example, for the image shown in Figure 1 column each project Π_i, *i*=1..*L* has the following state space for two agents involved in its implementation:

$$Z_i = \{(s_{i,1}s_{i,2}) \mid s_{i,j} \in \{0,1\}, j = 1,2\} = \{00, 01, 10, 11\}.$$

Obviously, the probability of realization of all combined states is equal to 1: $(1-q_{i,l})(1-q_{i,m}) + (1-q_{i,l})q_{i,m} + q_{i,l}(1-q_{i,m}) + q_{i,l}q_{i,m} = 1$. Here, the first index indicates the project number, and the second – is the agent number.

Given the composition of sets of ontologies, the probability of project coverage, for example, Π_1 is determined as follows: $p_{\text{cov}}(\Pi_1) = q_{1,1}q_{1,2}$, where $\Pi_1 = \{A_1, A_2\}$. If $q_{1,k} = q$, *k*=1,2, then there is a square dependence of the project coverage probability on the agents' refusal probability $p_{\text{cov}}(\Pi_1) = q^2$.

The probability of coverage can be increased by involving redundant agents in the project whose ontologies are partially included in many project ontologies. For example, there is a need to introduce an additional agent A_4 in the project Π_1, which will lead to the appearance in the column of a new connection between the vertices Π_1 and Π_4. As a result, the agent's state space will look like this:

$$Z_1 = \{(s_{1,1}s_{1,2}s_{1,3}) \mid s_{1,j} \in \{0,1\}, j = 1..3\} = \{000, 001, 010, 011, 100, 101, 110, 111\}.$$

For a new set of ontologies $\Pi_1 = \{A_1, A_2, A_4\}$, the probability of project coverage will now be equal to: $p'_{\text{cov}}(\Pi_1) = q_{1,1}(1-q_{1,2})q_{1,4} + q_{1,1}q_{1,2}(1-q_{1,4}) + q_{1,1}q_{1,2}q_{1,4}$. It's easy to check that $p'_{\text{cov}}(\Pi_1) \geq p_{\text{cov}}(\Pi_1)$. In a separate case for the same values of the probabilities of participation of agents in the project, there is $p'_{\text{cov}}(\Pi_1) = 2q^2(1-q) + q^3 \geq q^2$. Probability graphs $p_{\text{cov}}(\Pi_1)$ and $p'_{\text{cov}}(\Pi_1)$ of project coverage Π_1 for $q_{1,k} = q \in [0,1]$, k=1..K are shown in Figure 7.

Excessive project coverage affects the convergence of the stochastic game for the random graph colouring problem differently depending on the composition of the agent teams involved in the projects. Typical implementations of the time-averaged loss function for the project Π_1 for different coverage probabilities are presented in Figure 8. Graph 1 is obtained for $p_{\text{cov}}(\Pi_1) = 0.64$, and graph 2 – for $p'_{\text{cov}}(\Pi_1) = 0.768$, calculated for $q_{1,k} = q = 0.8$, k=1..K.

As shown in Figure 7, excessive coverage of projects due to the involvement of additional agents can increase the probability of coverage of the required project ontologies with existing agent ontologies. However, as shown in Figure 8, for random graphs, this may lead to a deterioration in the game method convergence due to the project's dependence on the same agents.

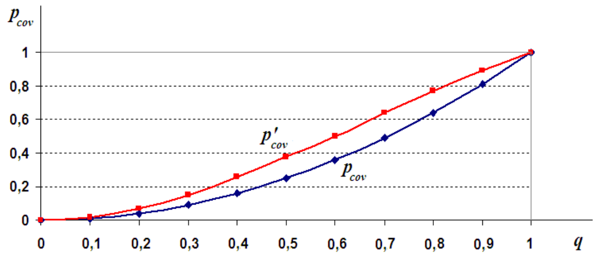

Figure 7. Probabilities of project coverage by ontologies of agents with refusals

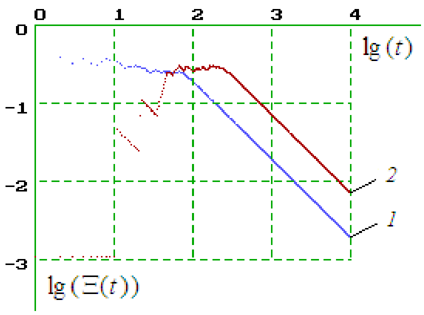

Figure 8. The effect of over-covering the project with agents' ontologies on the convergence of the stochastic game

The result of involving the same additional agents is an increase in the connectivity of the graph of projects. In case of refusal of agents, the graph of projects splits into more subgraphs, and isolated vertices often appear, which weakens the convergence of the stochastic game with the colouring random graph. Increasing the order of graph (the number of graph vertices), which is equivalent to increasing the number of projects, also slows down the convergence of the stochastic game.

The involvement of redundant agents in project implementation can be limited by an additional criterion of minimizing the cost of project implementation.

5. Conclusions

In this article, the problem of planning the sequence of projects implementation based on the self-learning method of the stochastic graph colouring game has been solved. It is highlighted that a random graph is a structural model for assigning staff based on the necessary ontological support for projects in conditions of uncertainty. Moreover, in such conditions, the well-known deterministic methods for solving this problem cannot be applied, since at each step of the game the implementation of a random graph is changed, the structure of which depicts the current coverage of a set of projects by ontologies of executive agents. The method of the multi-step stochastic game adaptively processes a random change in the structure of the graph. It has been shown that due to its adaptive properties, the stochastic game method can be used for random graph colouring, taking into account the probabilities of agents' participation in project implementation. The result of learning the stochastic game is the asymptotically correctly coloured random graph, which allows to determine the sequence of staff assignments to projects. According to the obtained results, it can be seen that the convergence of the stochastic game for the problem of colouring random graphs method is ensured by the balanced ratio of its parameters while observing the fundamental limitations of the stochastic approximation. In addition, increasing the graph order, the graph connectivity, and the probability of refusal of agents lead to the increase in the number of steps required for the convergence of the stochastic game of graph colouring. In our opinion, the stochastic game method for graph colouring can be used to solve similar problems formulated in terms of incomplete information, such as compiling various schedules, parallelization of algorithms, classification, data clustering and others. Moreover, the considered stochastic game has independent value as a model of global self-organization of states of the distributed system, which displays correct random graph colouring in conditions of uncertainty based on the locally collected data processing.

The game problem of assigning staff has the significant impact on industrial and management decisions, providing new approaches to solving complex problems and helping to make more justified and effective decisions. So, it is possible to expect the decision-making improvement in conditions of uncertainty, the increase in the efficiency and competitiveness of projects, the reduce of the staff costs and the increase of staff motivation, the improvement of the company management strategy, in particular, the staff selection and distribution, the stimulation for the development of new innovative approaches to staff selection and management, what will allow the company to increase its competitiveness in the market. The proposed research method can be used in many other cases where it is necessary to solve complex problems of resource allocation. For example, it can be used in manufacturing enterprises to assign workers to different jobs based on their skills and experience. Also, it can be applied in the field of logistics to assign vehicles to transport goods taking into account various factors such as the distance, time, carrying capacity, etc. The managerial game in logistics tasks may help managers to make decisions about the optimal use of resources such as workers, equipment, and materials. This can improve business efficiency and reduce costs. The conducted research has significant potential for further research and application in various areas related to staff management and organization. Some promising areas of research may include the model extension to more complex situations taking into account additional factors of project planning, such as the cost of staff labour, the project duration, technical constraints, etc., the research of the influence of changing conditions on the choice of strategies – changing priorities and goals or changing input data, development of new decision-making methods – combining the game with other methods, such as multi-agent systems, machine learning, information theory, and others. So, it can improve the accuracy and speed of decision-making, which, in turn, will increase the efficiency of enterprises and organizations.

Therefore, the study of the game problem of assigning staff to the project implementation and other similar tasks has significant potential for improving production processes and optimizing the use of resources in various industries.

Author Contributions: Conceptualization, P.K; methodology, P.K.; software, V.V.; validation, V.L.; formal analysis, A.K-S.; investigation, V.L.; resources, P.K.; data curation, P.K.; writing—original draft preparation, O.M., A.K-S.; writing—review and editing, A.K-S.; visualization, V.V.; supervision, V.L.; project administration, V.L.; funding acquisition, A.K-S. All authors have read and agreed to the published version of the manuscript.

Conflicts of Interest*:* The authors declare no conflicts of interest.

Acknowledgments: The research was financed from Statutory research, No. 13/010/BK_23/0072 (Institute of Economics and Computer Science, Faculty of Organization and Management, Silesian University of Technology).

References

Asdre, K., Ioannidou, K., & Nikolopoulos, S. D. (2007). The harmonious coloring problem is NP-complete for interval and permutation graphs. Discrete Applied Mathematics, 155(17), 2377-2382. https://doi.org/10.1016/j.dam.2007.07.005.

Belova, T. G., & Pobizhenko, I. O. (2017). Method for Assessing the Degree of Structural Similarity Connected Undirected Graphs. Information Processing Systems, 1(147), 9-12. https://doi.org/10.30748/soi.2017.147.02.

Benveniste, A., Métivier, M., & Priouret, P. (1990). Adaptive Algorithms and Stochastic Approximations. Stochastic Modelling and Applied Probability, 22. (1st ed.). Heidelberg: Springer Berlin. https://doi.org/10.1007/978-3-642-75894-2.

Blum, Y., & Rosenschein, J. S. (2008). Multiagent graph coloring: Pareto efficiency, fairness and individual rationality. Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, 1, 24-29. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=0e01a90cfe4e502e2b07cef9c1481d0afe021d13 Accessed 12 March 2023.

Bowman, A., Wyman, J. F., & Peters, J. (2002). The operations manual: A mechanism for improving the research process. Nursing Research, 51(2), 134–138. https://doi.org/10.1097/00006199-200203000-00011.

Burov, Y., Vysotska, V., & Kravets, P. (2019). Ontological approach to plot analysis and modeling. CEUR Workshop Proceedings, 2362, 22–31. https://ceur-ws.org/Vol-2362/paper3.pdf Accessed 12 March 2023.

Charteris, J., Berman, J., & Page, A. (2021). Virtual team professional learning and development for practitioners in education. Professional Development in Education, 47(4), 638-650. https://doi.org/10.1080/19415257.2021.1879215.

Chartrand, G., & Zhang, P. (2008). Chromatic graph theory. (1st ed.). New York: Chapman and Hall/CRC. https://doi.org/10.1201/9781584888017.

Chen, B.-S. (2019). Stochastic Game Strategies and Their Applications. (1st ed.). New York: CRC Press. https://doi.org/10.1201/9780429432941.

Chen, J. V., & Wang, Y.-H. (2011). Virtual Community and Online Game Players. In Information Resources Management Association (Ed.), Virtual Communities: Concepts, Methodologies, Tools and Applications (pp. 1511–1519). Hershey, PA: IGI Global. https://doi.org/10.4018/978-1-60960-100-3.ch427.

Chung, K. L. (1954). On a Stochastic Approximation Method. The Annals of Mathematical Statistics, 25(3), 463–483. https://doi.org/10.1214/aoms/1177728716.

De Lima, A. M., & Carmo, R. (2018). Exact Algorithms for the Graph Coloring Problem. Revista de Informática Teórica e Aplicada, 25(4), 57–73. https://doi.org/10.22456/2175-2745.80721.

Denysenko, O. (2019). Overview of Graph Coloring Methods and Algorithms. Λ′ΟΓΟΣ. The art of scientific mind, 7, 27-32. https://doi.org/10.36074/2617-7064.07.00.006.

Dey, A., Agarwal, A., Dixit, P., Long, H. V., Werner, F., Pal, T., & Son, L. H. (2019). A genetic algorithm for total graph coloring. Journal of Intelligent and Fuzzy Systems, 37(6), 7831-7838. https://doi.org/10.3233/JIFS-182816.

Dowsland, K. A., & Thompson, J. M. (2008). An improved ant colony optimisation heuristic for graph colouring. Discrete Applied Mathematics, 156(3), 313–324. https://doi.org/10.1016/j.dam.2007.03.025.

Flesch, J., Thuijsman, F., & Vrieze, O. J. (2003). Stochastic games with non-observable actions. Mathematical Methods of Operations Research, 58(3), 459–475. https://doi.org/10.1007/s001860300293.

Frieze, A., & Karoński, M. (2015). Introduction to Random Graphs. (1st ed.). Cambridge: Cambridge University Press. https://doi.org/10.1017/cbo9781316339831.

Fudenberg, D., & Tirole, J. (1991). Game theory. (1st ed.). Cambridge, Massachusetts: The MIT Press.

Fudenberg, D., & Tirole, J. (1991a). Game theory. London: The MIT Press. https://dokumen.tips/documents/mit-press-game-theory-fudenberg-tirole-579p.html?page=1 Accessed 12 March 2023.

Gozhyj, A., Kalinina, I., Gozhyj, V., & Vysotska, V. (2019). Web service interaction modeling with colored petri nets. Proceedings of the 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 18-21 September, 2019, Metz, France, 1, 319–323. https://doi.org/10.1109/IDAACS.2019.8924400.

Gupta, S., & Singh, D. P. (2020). Greedy graph coloring algorithm based on depth first search. International Journal on Emerging Technologies, 11(2), 854–862.

Hardy, B., Lewis, R., & Thompson, J. (2018). Tackling the edge dynamic graph colouring problem with and without future adjacency information. Journal of Heuristics, 24(3), 321–343. https://doi.org/10.1007/s10732-017-9327-z.

Hartley, R., & Thuijsman, F. (1994). Optimality and Equilibria in Stochastic Games. The Journal of the Operational Research Society, 45(1), 116–117. https://doi.org/10.2307/2583957.

Heagney, J. (2012). Fundamentals of project management, fourth edition. (4th ed.). New York: Amacon, American Management Association. https://www.nesacenter.org/uploaded/conferences/SEC/2014/handouts/Rick_Detwiler/15_Detwiler_Resources.pdf Accessed 12 March 2023.

Huang, M., & Ma, Y. (2016). Mean field stochastic games: Monotone costs and threshold policies. Proceedings of the IEEE 55th Conference on Decision and Control, CDC, ARIA Resort & Casino, December 12-14, 2016, Las Vegas, USA, 7105–7110. https://doi.org/10.1109/CDC.2016.7799364.

Saoub, K.R. (2021). Graph Theory : an introduction to proofs, algorithms, and applications. Textbooks in Mathematics. (1st ed.). Boca Raton, Florida: CRC Press, Taylor & Francis Group, LLC.

Kay, E., & Christofides, N. (1976). Graph Theory: An Algorithmic Approach. Operational Research Quarterly (1970-1977), 27(4), 1027–1028. https://doi.org/10.2307/3009122.

Keet, C. M. (2018). An Introduction to Ontology Engineering. Version 01. https://open.uct.ac.za/bitstream/handle/11427/28312/OEbook.pdf?sequence=1&isAllowed=y Accessed 12 March 2023.

Kiefer, J., & Wolfowitz, J. (1952). Stochastic Estimation of the Maximum of a Regression Function. Annals of Mathematical Statistics, 23, 462–466. https://doi.org/10.1214/aoms/1177729392.

Kravets, P., Burov, Y., Lytvyn, V., & Vysotska, V. (2019). Gaming method of ontology clusterization. Webology, 16(1), 55–76. http://www.webology.org/2019/v16n1/a179.pdf Accessed 12 March 2023.

Kravets, P., Lytvyn, V., Vysotska, V., & Burov, Y. (2020). Promoting training of multi-agent systems. CEUR Workshop Proceedings, 2608, 364–378. https://ceur-ws.org/Vol-2608/paper28.pdf Accessed 12 March 2023.

Kravets, P., Lytvyn, V., Vysotska, V., Burov, Y., & Andrusyak, I. (2021). Game task of ontological project coverage. CEUR Workshop Proceedings, 2851, 344–355. https://ceur-ws.org/Vol-2851/paper32.pdf Accessed 12 March 2023.

Kushner, H. J., & Yin, G. (1987). Asymptotic Properties of Distributed and Communicating Stochastic Approximation Algorithms. SIAM Journal on Control and Optimization, 25(5), 1266–1290. https://doi.org/10.1137/0325070.

Kushner, H. J., & Yin, G. (1987a). Stochastic approximation algorithms for parallel and distributed processing. Stochastics, 22(3–4), 219–250. https://doi.org/10.1080/17442508708833475.

Kushner, H. J., & Yin, G. G. (1999). Review of stochastic approximation algorithms and applications. Proceedings of the IEEE, 87(4), 688–690. https://doi.org/10.1109/JPROC.1999.752525.

Kushner, H. J., & Yin, G. G. (2023). Stochastic Approximation and Recursive Algorithms and Applications. Stochastic Modelling and Applied Probability, 35. New York: Springer-Verlag. https://doi.org/10.1007/b97441.

Lawrence, R. J. (2022). Success Factors for Global Virtual Team Leaders: A Qualitative Study of Leaders of Global Virtual Teams in a Global Professional Services Firm Employing Grounded Theory. University of Southern California : ProQuest Dissertations Publishing. https://www.proquest.com/openview/5d133f8ba0f39eae08990d3c8fa673b7/1?pq-origsite=gscholar&cbl=18750&diss=y Accessed 12 March 2023.

Lim, J., & Lee, D. (2020). A load balancing algorithm for mobile devices in edge cloud computing environments. Electronics, 9(4), 686. https://doi.org/10.3390/electronics9040686.

Liu, W. (2020). Production scheduling and equipment matching of flexible workshops based on multi-objective and multi-process hybrid optimization algorithm. Academic Journal of Manufacturing Engineering, 18(4), 158–163. https://ajme.ro/PDF_AJME_2020_4/L21.pdf Accessed 12 March 2023.

Li, Z., Zhu, E., Shao, Z., & Xu, J. (2018). NP-completeness of local colorings of graphs. Information Processing Letters, 130, 25–29. https://doi.org/10.1016/j.ipl.2017.09.013.

Makris, S. (2021). Dynamic Safety Zones in Human Robot Collaboration. Cooperating Robots for Flexible Manufacturing, Springer Series in Advanced Manufacturing. Springer, Cham, 271–287. https://doi.org/10.1007/978-3-030-51591-1_14.

Makris, S. (2021a). Introduction to Cooperating Robots and Flexible Manufacturing. Cooperating Robots for Flexible Manufacturing, Springer Series in Advanced Manufacturing. Springer, Cham, 3–32. https://doi.org/10.1007/978-3-030-51591-1_1.

Maynard, M. T., Mathieu, J. E., Gilson, L. L., R. Sanchez, D., & Dean, M. D. (2019). Do I Really Know You and Does It Matter? Unpacking the Relationship Between Familiarity and Information Elaboration in Global Virtual Teams. Group and Organization Management, 44(1), 3–37. https://doi.org/10.1177/1059601118785842.

Medeni, T. D., Elwell, M., Cook, S. A., & de Souza, E. G. M. (2011). Learning Reflection and Creativity in Online Game Communities. In Information Resources Management Association (Ed.), Virtual Communities: Concepts, Methodologies, Tools and

Applications (pp. 1760–1772). Hershey, PA: IGI Global. https://doi.org/10.4018/978-1-60960-100-3.ch512.

Meraihi, Y., Ramdane-Cherif, A., Mahseur, M., & Achelia, D. (2019). A Chaotic Binary Salp Swarm Algorithm for Solving the Graph Coloring Problem. In Chikhi, S., Amine, A., Chaoui, A., & Saidouni, D.E. (Eds.) Modelling and Implementation of Complex Systems. MISC 2018. Lecture Notes in Networks and Systems, vol. 64 (pp. 106–118). 0 Cham, Switzerland: Springer. https://doi.org/10.1007/978-3-030-05481-6_8.

Monasson, R. (2004). On the Analysis of Backtrack Proceduresfor the Colouring of Random Graphs. In Ben-Naim, E., Frauenfelder, H., & Toroczkai, Z. (Eds.) Complex Networks. Lecture Notes in Physics, vol. 650 (pp. 235–254). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-540-44485-5_11.

Nash, J. F. (1950). Equilibrium points in n-person games. Mathematics, 36(1), 48–49. https://doi.org/10.1073/pnas.36.1.48.

Nazin, A. V., & Poznyak, A. S. (1986). Adaptive Control of Finite Markov Chains Employing Stochastic Approximation Method. IFAC Proceedings Volumes, 19(5), 365–369. https://doi.org/10.1016/s1474-6670(17)59825-0.

Nazin, A. V., & Poznyak, A. S. (1987). Adaptive Control of Finite Markov Chains Employing Stochastic Approximation Method. IFAC Symposia Series, 2, 365-369. https://doi.org/10.1016/b978-0-08-033452-3.50071-5.

Neogy, S. K., Bapat, R. B., & Dubey, D. (2018). Mathematical Programming and Game Theory. (1st ed.). New Delhi, India: Indian Statistical Institute, Springer Nature Singapore Pte Ltd. https://link.springer.com/content/pdf/10.1007/978-981-13-3059-9.pdf Accessed 12 March 2023.

Neogy, S. K., Bapat, R. B., Dubey, D., & Parthasarathy, T. (2020). Special Issue on-Operations Research and Game Theory: Modeling and Computation. International Game Theory Review, 22(2). New-York: World Scientific Publishing Co Pte Ltd. https://doi.org/10.1142/S0219198920020016.

Neyman, A., & Sorin, S. (2003). Stochastic Games and Applications. Nato Science Series C:, 570. New York: Springer Science, Business Media. https://doi.org/10.1007/978-94-010-0189-2.

Osborne, M. J. (2000). An Introduction to Game Theory. (1st ed.). Oxford: Oxford University Press.

Osborne, M. J. (2000a). An Introduction to Game Theory by Please send comments to Department of Economics This version : Notes. https://www.academia.edu/40093924/An_Introduction_to_Game_Theory_by_Please_send_comments_to Accessed 12 March 2023.

Osborne, M. J. (2000b). Selected chapters from draft of: An introduction to game theory. https://mathematicalolympiads.files.wordpress.com/2012/08/martin_j-_osborne-an_introduction_to_game_theory-oxford_university_press_usa2003.pdf Accessed 12 March 2023.

Osborne, M. J. (2004). An Introduction to Game Theory. (2nd ed.). New Delhi: India.

Osborne, M. J. (2010). An Introduction to Game Theory. (International Edition). Oxford: Oxford University Press.

Osborne, M., & Rubinstein, A. (1994). Solution Manual for A Course in Game Theory. Ariel. https://docplayer.net/51517226-Solution-manual-for-a-course-in-game-theory-by-martin-j-osborne-and-ariel-rubinstein-with-the-assistance-of-wulong-gu.html Accessed 12 March 2023.

Panagopoulou, P. N., & Spirakis, P. G. (2008). A game theoretic approach for efficient graph coloring. In Hong, SH., Nagamochi, H., Fukunaga, T. (Eds.), Algorithms and Computation. Lecture Notes in Computer Science, vol 5369 (pp. 183–195). Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-540-92182-0_19.

Panik, M. J. (2017). Linear programming and resource allocation modeling. Hoboken: John Wiley & Sons, Inc. https://doi.org/10.1002/9781119509479.

Philipsen, W. J. M., & Stok, L. (1991). Graph coloring using neural networks. Proceedings of the IEEE International Symposium on Circuits and Systems, Singapore, 1597–1600. https://doi.org/10.1109/iscas.1991.176684.

Puterman, M. L. (2008). Markov decision processes: Discrete stochastic dynamic programming. Hoboken: John Wiley & Sons, Inc. https://doi.org/10.1002/9780470316887.

Raigorodskii, A. (2016). Models of random graphs and their applications to the web-graph analysis. Communications in Computer and Information Science, 573, 101–118. https://doi.org/10.1007/978-3-319-41718-9_5.

Raigorodskii, A. M. (2017). Small subgraphs in preferential attachment networks. Optimization Letters, 11(2), 249–257. https://doi.org/10.1007/s11590-015-0945-9.

Raigorodskii, A. M. & Karas V. S. (2022). Asymptotics of the independence number of a random subgraph of the graph G(n,r,<s). Mathematical Notes, 111, 124–131. https://doi.org/10.1134/S0001434622010138.

Romanuke, V. (2022). Pareto-Efficient Strategies in 2-Person Games In Staircase-Function Continuous And Finite Spaces. Decision Making: Applications in Management and Engineering, 5(1), 27–49. https://doi.org/10.31181/dmame0316022022r.

Romanuke, V. V. (2021). Refinement of acyclic-and-asymmetric payoff aggregates of pure strategy efficient Nash equilibria in finite noncooperative games by maximultimin and superoptimality. Decision Making: Applications in Management and Engineering, 4(2), 178–199. https://doi.org/10.31181/dmame210402178r.

Sahu, A., & Tapadar, R. (2006). Solving the Assignment problem using Genetic Algorithm and Simulated Annealing. Lecture Notes in Engineering and Computer Science. https://www.iaeng.org/IJAM/issues_v36/issue_1/IJAM_36_1_7.pdf Accessed 12 March 2023.

Shimizu, K., & Mori, R. (2022). Exponential-Time Quantum Algorithms for Graph Coloring Problems. Algorithmica, 84(12), 3603–3621. https://doi.org/10.1007/s00453-022-00976-2.

Shiryaev, A. N. (1996). Convergence of Probability Measures. Central Limit Theorem. Graduate Texts in Mathematics, 95, 308–378. https://doi.org/10.1007/978-1-4757-2539-1_4.

Shiryaev, A. N. (1996a). Mathematical Foundations of Probability Theory. Graduate Texts in Mathematics, 95, 131–307. https://doi.org/10.1007/978-1-4757-2539-1_3.

Shoham, Y., & Leyton-Brown, K. (2008). Multiagent systems: Algorithmic, Game-Theoretic, and logical foundations. Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511811654.

Thevenin, S., Zufferey, N., & Potvin, J. Y. (2017). Makespan minimisation for a parallel machine scheduling problem with preemption and job incompatibility. International Journal of Production Research, 55(6), 1588-1606. https://doi.org/10.1080/00207543.2016.1181285.

Thuijsman, F., Tijs, S. H., & Vrieze, O. J. (1991). Perfect equilibria in stochastic games. Journal of Optimization Theory and Applications, 69(2), 311–324. https://doi.org/10.1007/BF00940646.

Thuijsman, F., & Vrieze, K. (1993). Stationary ε-optimal strategies in stochastic games. Operations-Research-Spektrum, 15(1), 9–15. https://doi.org/10.1007/BF01783412.

Thuijsman, F., & Vrieze, O. J. (1987). The bad match; a total reward stochastic game. Operations-Research-Spektrum, 9(2), 93–99. https://doi.org/10.1007/BF01732644.

Thuijsman, F., & Vrieze, O. J. (1991). Easy Initial States in Stochastic Games. Theory and Decision Library, 7, 85–100. https://doi.org/10.1007/978-94-011-3760-7_8.

Thuijsman, F., & Vrieze, O. J. (1999). The Power of Threats in Stochastic Games. In Bardi, M., Raghavan, T.E.S., Parthasarathy, T. (Eds.) Stochastic and Differential Games. Annals of the International Society of Dynamic Games, vol 4 (pp. 343–357). Boston, MA: Birkhäuser. https://doi.org/10.1007/978-1-4612-1592-9_8.

Torabi Yeganeh, F., & Zegordi, S. H. (2020). A multi-objective optimization approach to project scheduling with resiliency criteria under uncertain activity duration. Annals of Operations Research, 285(1–2), 161–196. https://doi.org/10.1007/s10479-019-03375-z.

Umoren, R. A., Poore, J. A., Sweigart, L., Rybas, N., Gossett, E., Johnson, M., Allen, M., Scott, P. J., Truman, B., & Das, R. (2017). Teamstepps virtual teams: Interactive virtual team training and practice for health professional learners. Creative Nursing, 23(3), 184-191. https://doi.org/10.1891/1078-4535.23.3.184.

Ungureanu, V. (2018). Nash equilibrium conditions as extensions of some classical optimisation theorems. In Pareto-Nash-Stackelberg Game and Control Theory. Smart Innovation, Systems and Technologies, vol 89 (pp. 43–55). Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1_2.

Ungureanu, V. (2018a). Nash equilibrium set function in dyadic mixed-strategy games. In Pareto-Nash-Stackelberg Game and Control Theory. Smart Innovation, Systems and Technologies, vol 89 (pp. 115–127). Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1_6.

Ungureanu, V. (2018b). Pareto-Nash-Stackelberg Game and Control Theory: Intelligent Paradigms and Application. Smart Innovation, Systems and Technologies, 89. Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1.

Ungureanu, V. (2018c). Sets of nash equilibria in bimatrix 2×3 mixed-strategy games. In Pareto-Nash-Stackelberg Game and Control Theory. Smart Innovation, Systems and Technologies, vol 89 (pp. 115–127). Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1_4.

Ungureanu, V. (2018d). Sets of nash equilibria in polymatrix mixed-strategy games. In Pareto-Nash-Stackelberg Game and Control Theory. Smart Innovation, Systems and Technologies, vol 89 (pp. 83–96). Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1_3.

Ungureanu, V. (2018e). Stackelberg equilibrium sets in polymatrix mixed-strategy generalized stackelberg games. In Pareto-Nash-Stackelberg Game and Control Theory. Smart Innovation, Systems and Technologies, vol 89 (pp. 129–165). Cham: Springer. https://doi.org/10.1007/978-3-319-75151-1_7.

Neumann, J. V., & Morgenstern, O. (2007). Theory of games and economic behavior. (1st ed.). Princeton: Princeton University Press. https://uvammm.github.io/docs/theoryofgames.pdf Accessed 12 March 2023.

Wang, Z., Li, S., Wang, Y., & Li, S. (2009). The research of task assignment based on ant colony algorithm. Proceedings of the IEEE International Conference on Mechatronics and Automation, ICMA, August 9–12, Changchun, China, 2334–2339. https://doi.org/10.1109/ICMA.2009.5246570.

Wang, Z., & Man, Y. (2021). Machine learning-based intermittent equipment scheduling model for flexible production process. In Ren, J., Shen, W., Man, Y., & Dong, L. (Eds.), Applications of Artificial Intelligence in Process Systems Engineering, 2021 (pp. 473–495). Amsterdam: Elsevier Inc. https://doi.org/10.1016/B978-0-12-821092-5.00018-8.

Whillans, A., Perlow, L., & Turek, A. (2021). Experimenting during the shift to virtual team work: Learnings from how teams adapted their activities during the COVID-19 pandemic. Information and Organization, 31(1), 100343. https://doi.org/10.1016/j.infoandorg.2021.100343.

Wu, S. S., & Sweeting, D. (1994). Heuristic algorithms for task assignment and scheduling in a processor network. Parallel Computing, 20(1), 1–14. https://doi.org/10.1016/0167-8191(94)90109-0.

Yang, G., & Chen, S. (2020). Resource allocation algorithm and job scheduling of virtual manufacturing workshop. Academic Journal of Manufacturing Engineering, 18(2), 155–161. https://ajme.ro/PDF_AJME_2020_2/L22.pdf Accessed 12 March 2023.

Zais, M., & Laguna, M. (2016). A graph coloring approach to the deployment scheduling and unit assignment problem. Journal of Scheduling, 19(1), 73–90. https://doi.org/10.1007/s10951-015-0434-0.

Zhang, H., Seshadri, R., Prakash, A. A., Antoniou, C., Pereira, F. C., & Ben-Akiva, M. (2021). Improving the accuracy and efficiency of online calibration for simulation-based Dynamic Traffic Assignment. Transportation Research Part C: Emerging Technologies, 128, 103195. https://doi.org/10.1016/j.trc.2021.103195.

Zhu, A., & Yang, S. X. (2006). A neural network approach to dynamic task assignment of multirobots. IEEE Transactions on Neural Networks, 17(5), 1278–1287. https://doi.org/10.1109/TNN.2006.875994.

Zhukovskii, M. E., & Raigorodskii, A. M. (2015). Random graphs: models and asymptotic characteristics. Russian Mathematical Surveys, 70(1), 33. https://doi.org/10.1070/rm2015v070n01abeh004936.