# The Stochastic Game Model for Solving Self-organization Problem of the Hamilton Graph Cycle

Petro Kravets, Mykola Prodaniuk and Volodymyr Pasichnyk

*Lviv Polytechnic National University, 12 Bandera Street, Lviv, 79013, Ukraine*

**Abstract**

A new application of a stochastic game model for solving the problem of self-organization of the Hamiltonian graph cycle is proposed. To do this, game agents are placed at the vertices of an undirected graph, the pure strategies of which are options for choosing one of the incident edges. All agents' random choice of strategies forms a set of local paths that begin at each vertex of the graph. Current player payouts are defined as loss-making functions that depend on the strategies of neighboring players who control adjacent vertices of the graph. These functions are formed from a penalty for choosing opposing strategies by neighboring players and a penalty for strategies that have shortened the length of the local path.

Computer simulation of a stochastic game provided a pattern of self-organization of agents' strategies in the form of several local cycles or a global Hamiltonian cycle into the graph, depending on the ways in which the current losses of players are formed.

The study results can be used in practice for game solutions of NP-complete problems, transport, and communication problems, building authentication protocols in distributed information systems, and for collective decision-making under uncertainty.

**Keywords**

Self-organization, Behavioral pattern, graph, Hamiltonian cycle, stochastic agent game, Markov recurrent method.

## 1. Introduction

The rational functioning of natural and artificially distributed systems is possible due to the coordinated work of their active elements or agents. In such systems, the agent has a local behavior strategy determined by the survival and reproduction criteria in a changing environment. In the absence of coordination, the agents' actions will be chaotic, and their populations have no chance of survival. Coordination of agents' actions should be aimed at forming a global strategy for the behaviors of the system as a whole – achieving a certain balance of local goals of the system of agents, the deviation from which becomes disadvantageous to individual agents or groups of agents [1, 2].

Coordination can be centralized (vertical) or decentralized (horizontal). Vertical coordination has one center or hierarchically subordinate centers, and horizontal coordination is realized through the coordination of actions directly between all agents or within small groups of agents. Hierarchical centralized coordination requires a high level of the social organization of agents. In centralized systems, information flows flow rapidly, mostly from top to bottom, entropy and reaction time in such systems are minimal, but the cost of maintaining existence can be high. Decentralized coordination does not require a high level of the social organization of agents and is more democratic since information flows are mainly formed at the grassroots levels of the system, costs are minimized, but the entropy and time to produce a coordinated response will be much higher than in centralized systems. Decentralized coordination is a more natural form of coexistence of agents during their organizational evolution [3, 4].

Decentralized coordination of agents' actions is formed during their direct local interaction and multi-round exchange of information among themselves. The result of purposeful decentralized coordination is the unauthorized achievement of global coordination or the self-organization of agents, which manifests itself as established behavioral patterns in the horror of the entire decentralized system.

MoMLeT+DS 2023:5th International Workshop on Modern Machine Learning Technologies and Data Science, June 3, 2023, Lviv, Ukraine
EMAIL: Petro.O.Kravets@lpnu.ua, (P. Kravets); Mykola.M.Prodaniuk@lpnu.ua (M. Prodaniuk); Volodymyr.V.Pasichnyk@lpnu.ua (V. Pasichnyk)
ORCID: 0000-0001-8569-423X (P. Kravets); 0000-0001-9544-3792 (M. Prodaniuk); 0000-0002-5231-6395 (V. Pasichnyk)

Self-organization is a purposeful spatiotemporal process of creating, ordering, or improving the structure and functions of a complex open dynamic system due to its internal factors without organizing external influence. The necessary conditions for the self-organization of complex systems are multivariance, alternativeness, and randomness of agents' actions, guided by the process of self-learning and adaptation to uncertainties [5–7].

The structure of self-organization is a regular (nonchaotic) distribution of structural elements of a system or their states in time and space, which allows a holistic description of the system without characterizing all its constituent elements. Patterns are manifested at the macro level and are a consequence of local interactions of elements at the micro level [8].

Behavioral patterns determine the form of orderly actions of agents that arise from chaos through a process of coordination. The behavioral pattern is a systemic, globally coordinated behavior of a group of agents, which arises based on their local interaction during multi-step adaptive learning [9, 10].

The structural model of a decentralized system is conveniently represented as a graph, the vertices of which denote agents, and the edges determine the possibility of observing the internal states of neighboring agents and the local exchange of information between them. Then, depending on the strategies of agents' behavior, self-organization patterns can manifest themselves in the form of combinatorial entities within the graph – skeleton trees, painted vertices, Eulerian or Hamiltonian cycles, etc. [11, 12].

In practice, those combinatorial formations on graphs whose construction problem belongs to the class of NP-complete have been used, that is, it does not allow finding an exact solution by full enumeration of variants for large-order graphs in an acceptable polynomial time [13,14]. One such problem is the search for Hamiltonians on the cycle of the graph. A Hamiltonian cycle in a graph is a continuous path that passes through all vertices of a graph only once. For example, the travelling salesman problem (the Travelling Salesman Problem) is to find the minimal Hamiltonian cycle in the loaded graph [14, 15].

Not every graph has a Hamiltonian cycle. Unlike Euler cycles, no reliable conditions are known as to whether a given graph has a Hamiltonian cycle. In some cases, they are guided by statements about the dependence of the existence of Hamiltonian cycles on the powers of vertices of the graph [14, 16].

Precise or approximate (heuristic) methods can be used to search for Hamiltonian cycles. Although heuristic methods do not guarantee an optimal solution, they translate the problem into a polynomial complexity class, optimizing the enumeration of possible options.

This class also includes the strode of a stochastic game described in this article for the self-organization of the Hamiltonian cycle of an undirected graph. Its possible advantage is that thanks to self-learning and adaptation to uncertainties, it can find the Hamiltonian cycle in deterministic and random graphs. The disadvantages include a relatively small, power order of the rate of convergence, due to a priori uncertainty of a distributed system.

## 2. Practical applications of Hamiltonian cycles

Hamiltonian graph cycles are widely used in various fields [17]. They are used to solve various variants of the travelling salesman problem for optimal routing [18–22], modelling assemblies and genomes according to its selected fragments [23], identification by fingerprints [24], communication management [25], construction of cryptographic authentication protocols with zero knowledge disclosure [26] and others.

Zero-knowledge disclosure is an interactive method by which one party (provides evidence) can prove to another party (verifying evidence) that it knows some meaning (statement, object, or secret key) without telling the party. This method was first proposed in 1985 by authors S. Goldwasser, S. Micali, and C. Rackoff in “The Complexity of Knowledge of interactive evidence systems”.

Zero-knowledge proof has the following properties:

1. Completeness: if the statement $x$ is true, then A will convince B of this.
2. Correctness: if the statement $x$ is false, even a dishonest A cannot convince, except for $x$ a very small probability.
3. Zero Disclosure: If a statement $x$ is true, then anyone even dishonest A will know nothing, except the fact, that the statement $x$ is true.

Let a graph consisting of *n*-vertices numbered 1, 2 , .., $n$ and it has a Hamiltonian cycle. Then, enumerating all permutations in a symmetric group, we can be found in the Hamiltonian cycle. Since then, such a method of complete enumeration of options can hardly be realized in a reasonable time. It is known that from luck finding the Hamiltonian cycle in a graph is NP-complete.

## 3. Overview of methods for finding Hamiltonian graph cycles

The search for the Hamiltonian cycle is a special case of the travelling salesman problem, which finds the shortest Hamiltonian path of the loaded graph.

Combinatorial nodes of the search for Hamiltonian cycles of the graph are divided into those that provide exact solutions, and heuristics, which give approximate solutions [27, 28].

The methods of the first group are based on a complete enumeration of options or on deterministic calculations and can be applied to problems of relatively small dimensions. Heuristic methods use some improvements, and intuitive guesses to reduce the number of enumerated options. Some of them are probabilistic methods that rely on a random selection of options with the ability to select and memorize better solutions. Heuristic methods provide an approximate optimal solution to the problem in an acceptable polynomial time.

The formulation of the travelling salesman problem as a linear programming problem was first performed by N. Deo and S. L. Hakimi in 1965.

Let $\left[c_{i,j} > 0 | i,j = 1..n\right]$ be a matrix of weights (or distances) of edges of order graph $n$, and $\left[x_{i,j} | i,j = 1..n\right]$ be a matrix of displacements between vertices of the graph. The element $x_{i,j} = 1$, if an edge between vertices $i$ and $j$ ($i \neq j$) is included in the Hamiltonian path, and $x_{i,j} = 0$ – otherwise. The traveling salesman's tasks can be formulated as follows.

The objective function that minimizes the length of the Hamiltonian cycle is:

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j} x_{i,j} \rightarrow \min_{x}. \quad (1)$$

System of restrictions:

$\sum_{i=1}^{n} x_{i,j} = 1$ $(1 \leq j \leq n)$ there can be only one entrance to each vertex;

$\sum_{j=1}^{n} x_{i,j} = 1$ $(1 \leq i \leq n)$ there can be only one exit from each vertex;

$\sum_{i \in S} \sum_{j \notin S} x_{i,j} \geq 1$ $(S \neq \emptyset, S \subset \{1, \ldots, n\})$ is the exclusion of isolated cycles that together cover all vertices of the graph, there must be only one (Hamiltonian) cycle; for any subset of vertices $S$, there exists at least one arc that leads to other vertices of the graph.

$x_{i,j} \in \{0,1\}$ $(1 \leq i,j \leq n)$ is the desired matrix, the elements of which denote the Hamiltonian path.

The formalization of the problem (1), although it provides an exact solution by integer programming methods, for example, by the method of branches and boundaries, is cumbersome, contains many variables and constraints, and requires many calculations. A number of other methods can be used to accurately solve the traveling salesman problem, for example, dynamic programming, Lagrange multipliers, the modified method of branches and boundaries (Little's method), clipping planes (Danzing's-Falkerson-Johnson method).

Heuristic methods provide a solution close to optimal for large-dimensional problems within acceptable time limits. These include methods of nearest neighbor, greedy, insertion, local optimization of *k-opt* to improve the initial solution, Lin-Kernighan, decomposition, and cross-linking [29], elastic network, artificial neural networks, genetic, ant, and others.

Similar exact and approximate methods are used to solve the problem of finding Hamiltonian cycles of an unloaded graph.

Accurate methods include brute force, backtracking, algebraic method, Roberts-Flores method, linear programming, integer programming (Gomori, branch and boundary method), dynamic programming and others.

Since the problem of determining Hamiltonian cycles is NP-complete, the application of the brute force method can only be used for small-order graphs.

To speed up the enumeration of options, you can use the backtracking method (search with return), which excludes from consideration a significant number of options by one check, building a decision tree and traversing it in depth [30].

First, select an arbitrary vertex of the graph, for example, the first and one of the edges incidents to it, along which we proceed to the next vertex. We remember the vertices passed. Suppose that the $k$ -th vertex of the loop has already been found. If is equal to the number $n$ of vertices of the graph and there is an edge that connects the $k$ -th vertex to the first, then the loop is found. If there are edges extending from the $k$ -th vertex to the still unviewed vertices, then we include one of them in the solution and continue the search from it. If such edges do not exist, then go back one step to the $(k-1)$-th vertex and continue the search. So, all the vertices of the Hamiltonian cycle will be found.

In [31] a new method and corresponding polynomial algorithm for solving the problem of finding the Hamiltonian cycle of a graph are proposed. Based on a given graph, another graph of the shortest paths is constructed. To construct a graph of shortest paths, Dijkstra's algorithm is used. The search for the Hamiltonian cycle in the graph is reduced to the search for a closed path in the shortest path graph. The enumeration space for solutions to a problem consists of solutions that are constructed from each vertex of the graph in the shortest path graph.

An algebraic method based on multiple symbolic multiplications of a modified adjacency matrix $B$ of order $n$ (single elements of an adjacency matrix are replaced by the literal notation of graph vertices) by a matrix of sums of internal (without extreme vertices) products of notation of vertices of nonzero chains between vertices of a graph: $P_{i+1} = B \cdot P_i$ [11]. Multiplication is performed until a matrix $P_{n-1}$ containing all Hamiltonian chains between all pairs of vertices is computed. Hamiltonian cycles are obtained from those chains whose edges $P_{n-1}$ or vertices are connected by arcs. The disadvantage of this method is that for large orders of the graph, each element of the matrix $P_i$ will consist of many constituents' sub-elements, which require large amounts of memory to store.

Unlike the algebraic method, which ensures obtaining all existing Hamiltonian cycles of a graph, the Roberts-Flores enumeration method works with only one path, which is extended until a Hamiltonian cycle is found, or it turns out that this path cannot lead to a Hamiltonian cycle [11]. After that, the path is modified in some systematic way and the cycle search continues for it. By reducing the number of computations required, this method is efficient for large graphs.

For practical applications, heuristic methods for finding Hamiltonian cycles are more effective, which reduces the complete enumeration of options. These include the nearest neighbor, greedy, annealing simulations, banned search, Hopfield's artificial neural network, evolutionary, genetic, ant colony, and others.

In the context of self-organization, those heuristic methods that have the property of self-learning are important. Among these, it is worth noting the methods of neural networks, genetics, ant colony, and stochastic play, which have polynomial complexity.

The problem of finding the Hamiltonian cycle can be solved using the Hopfield neural network, which implements learning without a teacher [32]. To do this, the conditions of problem (1) must be translated into parameters of connections between neurons.

For the Hopfield network to determine the Hamiltonian cycle, the following requirements must be imposed:

1. The neural network should consist of $N = n \times n$ neurons, which can be considered as square matrix of $n$ rows and $n$ columns, where $n$ is the order of the graph.
2. There are connections between all pairs $(k, l)$ of neurons to which weightings are attributed $W_{k,l}$.
3. Not all grid weights can be negative at the same time.
4. The active neuron in each column specifies the corresponding vertex of the graph (or another route city for the traveling salesman problem).
5. The network response must contain only one active neuron in each row and each column. For this, the network weights must be constructed so that each neuron interferes with the activation of other neurons in its row and in its column.
6. To minimize the path length, it is necessary that the neuron in the $j$ -th column the more actively interferes with the activation of neurons in the $(j+1)$ -th and $(j-1)$ -th columns, the greater the distance between them (required to solve the traveling salesman problem).

All these conditions are satisfied by the following formula for calculating the weight between the neuron corresponding to the $x$-city (row) at the $i$-th position (column) and the neuron corresponding to the $y$-city (row) at the $j$-th position (column):

$$W_{k,l} = W_{xi,yj} = -A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy}) - Cd(x,y)(\delta_{i,j+1} + \delta_{i,j-1}) + D,$$

where $k = (x,i)$, $l = (y,j)$ are the coordinates of the connections between the neurons of the matrix; $A, B, C, D$ are some constants; $d(x,y)$ is the distance between cities $x$ and $y$; $\delta_{xy}$ is the Kronecker symbol taking the value 1 if $x = y$ and the value 0 is otherwise. As it is easy to see, the first term is equal $-A$ for all connections in the same line ($x = y$), except the connection of the neuron with itself (at $i = j$). The second term is equal $-B$ for all relationships in the same column ($i = j$), except the relation to itself ($x = y$). The third term is proportional to the distance between cities $x$ and $y$ if these cities are adjacent in the route ($i = j - 1$ or $i = j + 1$).

Starting from the initial random state, the Hopfield network can provide a suboptimal solution to the problem (for the traveling salesman problem, the resulting cycle may differ from the optimal one). The experiment can be conducted several times and choose the best solution.

Heuristic nodes based on genetic algorithms and models are the evolutionary process of natural reflection, inheritance, and mutation. Each Hamiltonian cycle is treated as a chromosome. At the initial stage, there is a set of such chromosomes (the initial population), and at each subsequent cycle, a new one is produced from the existing population by pairwise crossing and mutations [27, 33].

A simplified genetic algorithm consists of the following steps:

1. Creation of an initial population with $N$ chromosomes of length $n + 1$, where $n$ is the order of the graph.
2. Calculation of fitness functions (route lengths) $S_i$ ($i = 1 \ldots N$) for all individuals of the population.
3. Realization of the original selection of the parents of chromosomes (with the best value of $S_i$), they're pairwise crossing, random mutation, and the formation of a new generation of chromosomes.
4. If at least one Hamiltonian path is found, then go to step 5, otherwise return to step 2.
5. Choose the best solution found to the problem (for a loaded graph in the traveling salesman problem).

The genetic algorithm allows you to get more than one solution in a single application and select the best one. You can run several runs of the algorithm and calculate the average value of the best results. As the graph order increases, the probability of obtaining an optimal solution decrease.

The ant algorithm is based on the behavior of an ant colony and simulates the evaporation of ovation of pheromones. Ants find their way between an anthill and a food source by the smell of pheromones they leave behind. The more ants use the same path, the higher the concentration of pheromones on it. This "ant logic" allows you to choose a shorter path between the end points of the route [27, 34].

For each ant, the transition from point $i$ to point $j$ is determined by its memory $M$ (the list of points that can still be visited), the visibility $\mu_{i,j}$ between points (the presence of an edge $e_{i,j}$ in the graph), and the pheromone trail $\tau_{i,j}$. The selection of the next vertex of the graph is carried out with probability.

$$p = \chi(j \in M) \cdot \frac{\mu_{i,j}^{\alpha} \tau_{i,j}^{\beta}}{\sum_{k \in M} \mu_{i,k}^{\alpha} \tau_{i,k}^{\beta}},$$

where $\chi(j) \in \{0,1\}$ is the indicator function of the event; $\alpha$, $\beta$ are the weight parameters.

For the considered neural, ant, and genetic algorithms, it is problematic to choose the initial values of the parameters that will ensure their convergence to the optimal solution.

Even though the described heuristic methods provide the search for an approximate optimal Hamiltonian cycle, their methodological value is in demonstrating different implementations of self-learning of active systems for solving NP-complete problems without the need to use classical algorithms. Such methods implement "soft" calculations that do not require traditional programming and can be easily adapted to solve other problems. The stochastic game method also has similar properties.

In this article, we propose a new application of the stochastic game method [35] for the self-organization of Hamiltonian cycles of an unloaded graph.

## 4. The purpose of the work

The aim of the work is to solve the stochastic game problem of self-organization of strategies for constructing a Hamiltonian cycle of an unloaded undirected graph. Comprehension of the goal is provided using the adaptive method in solving a stochastic game, proper adjustment of its parameters, planning a computer experiment, developing a software model and a stochastic game, analyzing the results and making recommendations for their practical application.

## 5. Formulation of the game problem of finding Hamiltonian cycles

Let the undirected graph $G = (V, E)$ be given by a finite set of vertices $V$ and edges $E$ .The graph is ordered, that is, it's edges is numbered in ascending order, for example, clockwise. We assume that the graph is simple (has no loops and multiple edges) and is connected (does not contain isolated vertices). Furthermore, given the possibility of covering the cycle of all vertices without loss of generality, it can be assumed that the minimum degree of vertices of a deterministic graph is greater than 1 (graph without leaves).

At each vertex of the graph, we place the game agent $A_i$, who can choose one of the incident vertices of $i \in V$ edges belonging to the local set $E_i \subseteq E$. To do this, each agent with a number $i \in V$ has $N_i \geq 2$ pure strategies $X^i = \left(x^i[1], x^i[2], \dots, x^i[N_i]\right)$, where $x^i \in E_i$ is the edge number and the $N_i$ value is determined by the power of the corresponding vertex. Hereinafter, superscript is not a power of a number, except for symbols with a minus sign.

The variants of the possible choice can be given by an oriented graph of players' strategies. Let the arcs coming out of the vertex with the number $i$, denote the strategies of the $i$ -th player regarding the choice of one of the neighboring players, and those arcs that enter the $i$ -th vertex denote the dependence of the winnings or losses of the $i$ -th player on the strategies of neighboring players. The correspondence of given undirected graph and formed on its basis-oriented graph of strategies is shown in Figure 1.

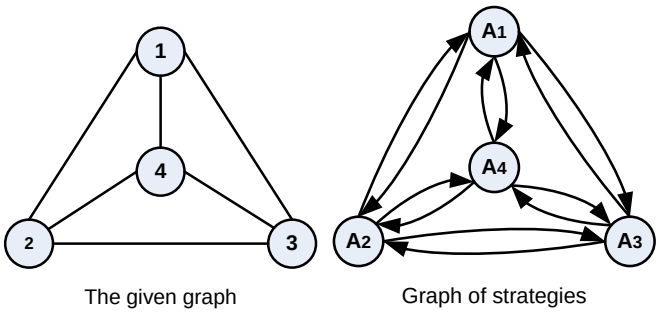



**Figure 1:** Correspondence of graphs

The choice of pure strategies is made by the players independently and synchronously at discrete moments in time $n = 1,2, \dots$. The choice made is estimated by the current loss $\zeta_n^i$, which is formed depending on how much the agent's action led to an approach to the target solution of $n = 1,2, \dots$ the problem.

To do this, each game agent $i \in V$ makes an independent choice of one of $N_i$ own pure strategies $x_n^i = x^i \in X^i$ according to conditional probabilities.

$$p_n^i(j) = P\{x_n^i = x^i(j) | x_t^i, \zeta_t^i (t = 1,2, \dots, n-1)\},\ j = 1..N_i,$$

where $\{x_t^i | t = 1,2, \dots, n-1\}$ is the background of the strategies chosen by the player with the number $i$; $\{\zeta_t^i | t = 1,2, \dots, n-1\}$ is the background of the losses received for this.

The probability set $p_n^i = \left(p_n^i(1), p_n^i(2), \dots, p_n^i(N_i)\right)$ $\forall i \in V$ specifies the vectors of mixed strategies of players who take values $p_n^i \in S^{N_i}$ on $N_i$-dimensional unit simplexes:

$$S^{N_i} = \left\{ p \left| \sum_{j=1}^{N_i} p(j) = 1;\ p(j) \geq 0\ \ (j = 1..N_i) \right. \right\}.$$

To select pure strategies $\{x_n^i\}$, a random mechanism is used, built on a discrete dynamic distribution, which is given by vectors of mixed strategies $p_n^i$:

$$x_n^i = \left\{ x^i(l) | l = arg\ \min_l \sum_{k=1}^{l} p_n^i[k] > \omega\ (k, l = 1..N_i) \right\}, \quad (2)$$

where $\omega \in [0, 1]$ is a real random variable with uniform distribution.

After completing the selection of pure strategies by all players, they receive random current losses $\zeta_n^i = \zeta_n^i(x_n^{V_i})$, whose values are a function of the combined strategies $x^{V_i} \in X^{V_i} = \underset{j \in D_i}{\times} X^j$ of neighboring players from subsets $V_i \subseteq V$ $(V_i \neq \emptyset)$ $\forall i \in V$. It is assumed that current losses have a constant mathematical expectation and a limited second point, which are not known to agents a priori.

Depending on the selected criteria and initial parameters of the game method, several autonomous loops or one global loop may occur in the graph.

To create autonomous disjoint cycles of a graph, the condition must be satisfied that the choice of an edge in the direction of the $i$ -th agent ($\forall i \in V$) can be carried out only by one of its neighbors agent from the set $V_i$.

Let $k_n^i = \sum_{j \in V_i} \chi(x_n^j \to x_n^i)$ be the current number of neighboring agents whose strategies are directed to the vertex of the graph where the $i$ -th agent is located (in the direction of the $i$ -th agent). The heterogeneity of directing strategies within a local subset $V_i$ is considered by the difference penalty:

$$\xi_n^i[1] = C_i^{-1} \sum_{j \in V_i} |k_n^i - k_n^j|, \quad (3)$$

where $\xi_n^i[1] \in R_+^1$ is a real number; $C_i = |V_i|$ is the number of agents of the local subset $V_i$ .

This criterion provides the possibility of forming cycles of movement of agents on the graph.

To form a Hamiltonian cycle, each agent must choose a strategy (incident to its vertex edge) to maximize the length of the local path emanating from the agent-controlled vertex of the graph. Let $r_n^i$ be a set of vertices (or sequence of numbers of vertices) that defines such a path with a root vertex with a number $i$, formed at the current time $n$ by the strategies (directions of movement) $x_n^i$ of players. The path $r_n^i$ consists of the values $m := x_n^m$ determined by the players' pure strategies $x_n^m$. The recurrent procedure for determining the current path that starts at the vertex with number $i$, will be:

$$r_n^i(t) = r_n^i(t-1) \bigcup_{m=i}^{\{m\} \cap r_n^i(t-1) \neq \emptyset} \{m |\ m := x_n^m\},\ \forall i \in V, \quad (4)$$

where $r_n^i(0) = \emptyset$.

The notation of the operation of combining sets into (4) should be read so that this operation is performed repeatedly, starting with the value $m = i$ until the element $r_n^i$ reappears during the formation of the path $m \in r_n^i(t-1)$. For a connected graph with vertex powers greater than 1, this condition will always hold – following the players' strategies from the $i$ -th vertex ($\forall i \in V$), entry into a local loop or a global Hamiltonian cycle will occur. The minimal local loop covers two edge-connected vertices of the graph and is formed by the opposite strategies of two neighboring players. The maximum cycle covers all vertices of the graph and is one of the Hamiltonian cycles.

The deviation of the length of the local path from the maximum possible is considered by the penalty:

$$\xi_n^i[2] = C^{-1} \left( C - S(r_n^i) \right), \quad (5)$$

where $\xi_n^i[2] \in R_+^1$ is a real number; $C = |V|$ is a number of game agents; $S(r_n^i) = |r_n^i|$ is the length of the local path (the number of vertices of the graph on the way to entering the cycle), starting at the $i$ -th vertex ($0 \leq S(r_n^i) \leq C$).

This criterion approximates the length of the path $r_n^i$, leaving the vertex $i$, to the number of vertices $V$ of the graph $G$ (to the length of the Hamiltonian cycle).

Penalties (3) and (5) can be used individually or comprehensively. The total current losses of players are calculated after the selection of moving options $x_n^i$ $\forall i \in V$ is completed:

$$\zeta_n^i = \lambda\xi_n^i[1] + (1-\lambda)\xi_n^i[2], \quad (6)$$

where $\lambda \in [0,1]$ is the weighting factor that determines the share of penalty criteria in the formation of agents' losses.

Due to the random selection of pure strategies, current losses $\zeta_n^i \ \forall i \in V$ are random variables with a priori unknown stochastic characteristics. To formulate criteria for player behavior, time-averaged losses are used:

$$Z_n^i = \frac{1}{n}\sum_{t=1}^{n}\zeta_t^i,\ n = 1,2,\dots,\ \forall i \in V\ . \quad (7)$$

Strategies of players should be aimed at minimizing their own functions of average losses (7):

$$\overline{\lim_{n\to\infty}} Z_n^i \to \min_{x_n^i}\ \forall i \in V. \quad (8)$$

The stochastic game of finding a Hamiltonian cycle is that each player $i \in V$ must learn to choose pure strategies $\{x_n^i\}$ (2) at points in time $n = 1,2,\dots$ based on the observation of locally determined current losses $\{\zeta_n^i\}$ (6) so as to ensure that the system of criteria (8) is met.

The method of forming a sequence of strategies $\{x_n^i\}\ \forall i \in V$ $(n = 1,2,\dots)$ of stochastic game will determine the fulfillment of one of the conditions of collective optimality, for example, Nash, Pareto or another [36].

The learning process of a stochastic game, which leads to self-organization of strategies for moving game agents, is evaluated by the following characteristics.

1. The system function of average losses, which is the averaging of individual functions of average losses of players (7):

$$Z_n = C^{-1}\sum_{i\in V} Z_n^i\ . \quad (9)$$

2. The strategy coordination coefficient, which is the relative number of coordinated strategies of the players:

$$K_n = (nC)^{-1}\sum_{t=1}^{n}\sum_{i\in V}\chi\left(\left|\zeta_t^i\right| \le \delta\right), \quad (10)$$

where $\chi(*) \in \{0,1\}$ is the indicator function of the event: if the condition holds, then $\chi(*) = 1$, otherwise – $\chi(*) = 0$; $0 < \delta << 1$ is a small positive real number.

The self-organization of the Hamiltonian cycle will be indicated by a decrease in the function $Z_n \ge 0$ of system losses and an increase in the coordination coefficient $K_n \in [0,1]$ of agents' strategies.

## 6. Method for solving a stochastic game.

Formation of sequences $\{x_n^i\}$ with the desired properties will be performed using the recurrent method in changing the vectors of mixed strategies [37, 38]:

$$p_{n+1}^i = \pi_{\varepsilon_{n+1}}^{N_i}\{p_n^i - \gamma_n R(p_n^i, x_n^i, \zeta_n^i)\}, \quad (11)$$

where $\pi_{\varepsilon_{n+1}}^{N_i}$ is a projector on unit $\varepsilon$-simplex $S_\varepsilon^{N_i} \subseteq S^{N_i}$ [37]; $\gamma_n$ is a monotonically descending sequence of positive values, which regulates the step size of the method; $R$ is a method step; $\varepsilon_n$ is a monotonically decreasing sequence of positive quantities that regulates the rate of expansion of $\varepsilon$-simplex.

The change in the elements of the vector of mixed strategies is constructed in such a way that when choosing a strategy $x_n^i(j)$, the element $p_n^i(j)$ decreases in proportion to the amount of the current loss $\zeta_n^i$. The other elements of the vector of mixed strategies do not change or grow proportionally of $\zeta_n^i$. After recalculating the vectors of mixed strategies, they are normalized by the method of projection on the unit $\varepsilon$-simplex. As a result, smaller displacements of the vectors of mixed states on the unit $\varepsilon$-simplex.

For theses in the recurrent method, we use the results of the theory of stochastic approximation [38]. For this, suppose that the mathematical expectation of random losses $M\{\zeta_n^i(x)\} = l^i(x)$ is constant for all $x \in X = \bigotimes_{i\in V} X^i$. Then the average loss function of the matrix game is calculated as:

$$L^i(p^{D_i}) = \sum_{x^{V_i} \in X^{D_i}} l^i(x^{V_i}) \prod_{j \in V_i; x^j \in x^{V_i}} p^j(x^j) \quad (12)$$

where $p^{V_i} \in S^{V_i} = \prod_{j \in V_i} S^{N_j}$; $p^i \in S^{N_i}$.

The goal of the players is to minimize their average loss functions for mixed strategies $p^i$:

$$L^i(p^{V_i}) \to \min_{p^i}.$$

Let the expectation of the motion vector of method (11) be the gradient of the mean loss function (12):

$$M\{R(p_n^i, x_n^i, \zeta_n^i)\} = \nabla_{p^i} L^i(p^{V_i}).$$

Considering that

$$\nabla_{p^i} L^i = M\left\{\frac{\zeta_n^i}{e^T(x_n^i)p_n^i} e(x_n^i) \middle| p_n^i = p^i\right\},$$

where $e(x_n^i)$ is the unit vector–indicator of choice of pure strategy $x_n^i \in X^i$, based on stochastic approximation we obtain a gradient method for solving the game problem:

$$p_{n+1}^i = \pi_{\varepsilon_{n+1}}^{N_i}\left\{p_n^i - \gamma_n \frac{\zeta_n^i e(x_n^i)}{e^m(x_n^i)p_n^i}\right\}. \quad (13)$$

The parameter $\gamma_n$ reduces the step size of the method to achieve optimal collective solutions of the game: $\|p_n^i - p_*^i\| \to 0$, and the parameter $\varepsilon_n$ extends the $\varepsilon$ -simplex: $S_{\varepsilon_{n+1}}^{N_i} \to S^{N_i}$.

The values of these parameters can be calculated as follows:

$$\gamma_n = \gamma n^{-\alpha},\ \varepsilon_n = \varepsilon n^{-\beta}, \quad (14)$$

where $\gamma > 0$, $\alpha \in (0,1]$, $\varepsilon > 0$, $\beta > 0$.

The convergence of stochastic game strategies to collective-optimal values $p_*^i$ is determined by the ratios of parameters $\gamma_n$ and $\varepsilon_n$ (14), which must meet the fundamental conditions of the stochastic approximation [37, 38].

As shown in [39], to ensure the root mean square (rms) convergence of the game method (13) to the Nash point in the positive environment, the following relations must be fulfilled:

$$0 < \beta < \alpha < 1. \quad (15)$$

The theoretical order of the asymptotic rate of convergence of method (13) is $n^{-\theta}$, where $\theta = min\{\beta, \alpha - \beta, 1 - \alpha\}$ is the order parameter. The maximum value of parameter $\theta = \frac{1}{2}$ is reached for $\alpha - \beta = \frac{1}{2}$.

The stochastic game begins with untrained mixed agent strategies: $p_0^i = \left(\frac{1}{N_i}, \dots, \frac{1}{N_i}\right)$ $\forall i \in V$. In moments of time $n = 1, 2, \dots$ mixed strategies are dynamically rearranged according to (13) for adaptive selection of pure strategies.

One step of repeating stochastic play is that at a point in time $n$ each player $i \in V$ chooses a pure strategy $x_n^i$ (2) and by time $n$+1 receives a current loss $\zeta_n^i$ (6), which is used to calculate the new mixed strategy $p_{n+1}^i$ (13).

The stochastic agent game implements adaptive learning by trial and error, which requires a significant number of trials to find the desired solution. The learning process can be significantly accelerated by using a computer implementation of a stochastic game with the appropriate adjustment of its parameters.

## 7. A game algorithm for finding the Hamiltonian cycle of a graph.

Step 1. Set initial values for parameters:

$C = |V|$ – the number of players, which is equal to the number of vertices of the graph;

$M^{C \times C}$ – a matrix of adjacencies of the graph;

$N_i$ – the number of pure strategies of the $i$ -th player, which is equal to the power of the $i$ -th vertex of the graph;

$X^i = \left(x^i[1], x^i[2], \ldots, x^i[N_i]\right)$, $i = 1..C$ – vectors of players' pure strategies, where is the number of the edge incident to the vertex of the graph;

$p_0^i = \left(\frac{1}{N_i}, \ldots, \frac{1}{N_i}\right)$, $i = 1..C$ – initial mixed player strategies.

$\gamma > 0$ – a parameter of the learning step;

$\alpha \in (0,1]$ – an order of the learning step;

$\varepsilon$ – a parameter of $\varepsilon$ -simplex;

$\beta > 0$ – an order of the expansion rate of $\varepsilon$ -simplex;

$\lambda \in [0,1]$ – a weighting factor that determines the share of penalty criteria in the formation of player losses.

$n = 0$ – an initial time moment;

$n_{max}$ – a maximum number of the method steps.

Step 2. Select action options $x_n^i \in X^i$, $i = 1..C$ according to (2).

Step 3. Get the value of current losses $\zeta_n^i$ according to (6).

Step 4. Calculate the parameter values $\gamma_n$, $\varepsilon_n$ according to (14).

Step 5. Calculate the elements of the mixed strategy vectors $p_n^i$, $i = 1..C$ according to (13).

Step 6. Calculate the self-organization characteristics of the stochastic game $Z_n$ according to (9) and $K_n$ according to (10).

Step 7. Set the next time moment $n := n + 1$.

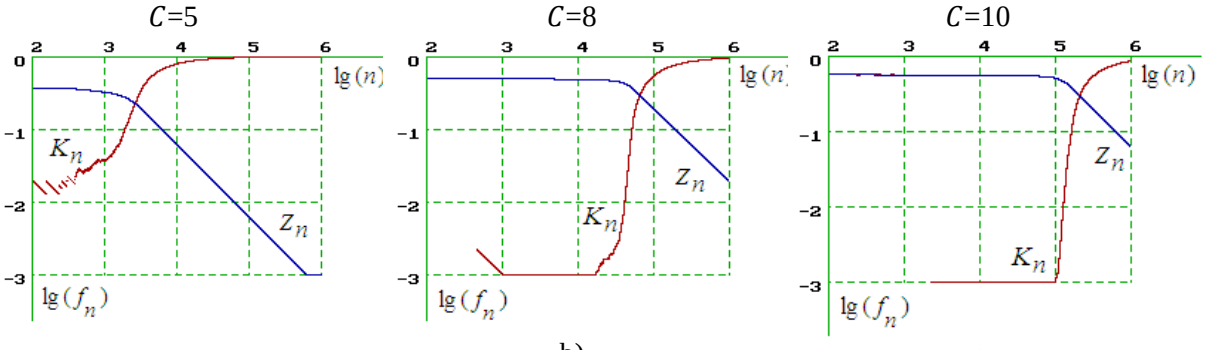
Step 8. If $n < n_{max}$, then go to step 2, otherwise the algorithm stops.

## 8. Results of a computer experiment

First, consider an undirected full-connected graph $G = (V, E)$ without loops. In the program, the graph is conveniently specified either by the matrix of adjacencies or by the incident matrix. In each vertex of the graph, we place one agent. In this case, the count will control $C = |V|$ the agents. In a repetitive stochastic game, one of the agents can knockout and rat one of the incident edges of the graph. Linked by strategies with the combined choice of several agents, forms a local path. The task of the game is the adaptive fusion of local paths of agents into one global Hamiltonian cycle.

To solve a stochastic game, we use the stochastic game method (13) with the following parameters: $\lambda = 0.5$, $\gamma = 1$, $\varepsilon = \frac{0.999}{N_i}$, $\alpha = 0.5$, $\beta = 0.25$. In addition to the values of parameters satisfying condition (15), for a stochastic game to converge, it is necessary that the graph has a Hamiltonian cycle.

The number of vertices of the graph and the powers of its vertices will significantly affect the order of the rate of convergence of the graph and the rate of convergence of the game method. The functions of average player losses $Z_n$ (9) and strategy coordination ratio $K_n$ (10) characterize the course of a stochastic game of agent movement. Graphs of these functions for different values of order $C = |V|$ (number of vertices) of a full-connected graph are shown in Figure 2 on a logarithmic scale. The image is bounded by the coordinates of the rectangular output area of the graphs.




a) b) c)

**Figure 2:** Self-organization indicators of a stochastic game for different values of the order of the full graph

A decrease in the average loss function $Z_n$ indicates that the target conditions (6) of the convergence of the stochastic game are met. The approximation of the value of the coordination coefficient $K_n$ to 1 (that is, to logarithmic zero) indicates the self-organization of the players' strategies. For the given parameters of the game method, increasing the order $C$ of a full-connected graph leads to a significant increase in the number of learning steps of the stochastic game. Thus, for $C = 5$, the rapid growth of the coordination coefficient begins at about $n = 10^3$ steps of learning the stochastic game, for $C = 10$ it takes a little more than $n = 10^4$ steps, and for $C = 15$ it takes about $n = 10^5$ steps.

In addition to the order of a Hamiltonian graph, the convergence time of a stochastic game will also be determined by its connectivity and the parameters of the game method. Reducing the connectivity of the graph accelerates the self-organization of cycles.

Depending on the selected criteria and initial parameters of the game method, several autonomous loops or one global loop may appear in the graph.

If the parameter $\lambda$ of the complex penalty (6) takes values close to 1, then the dominant influence on the course of the game will be the penalty criterion (3), which minimizes the selection of each vertex of the graph by neighboring players. The application of this criterion can lead to the formation of several separate (autonomous) cycles in the graph, as shown in Figure 3. For the parameter values given above, we obtain solutions of a stochastic game in pure strategies. The stochastic game provides a multivariate solution from the problem of constructing graph cycles.

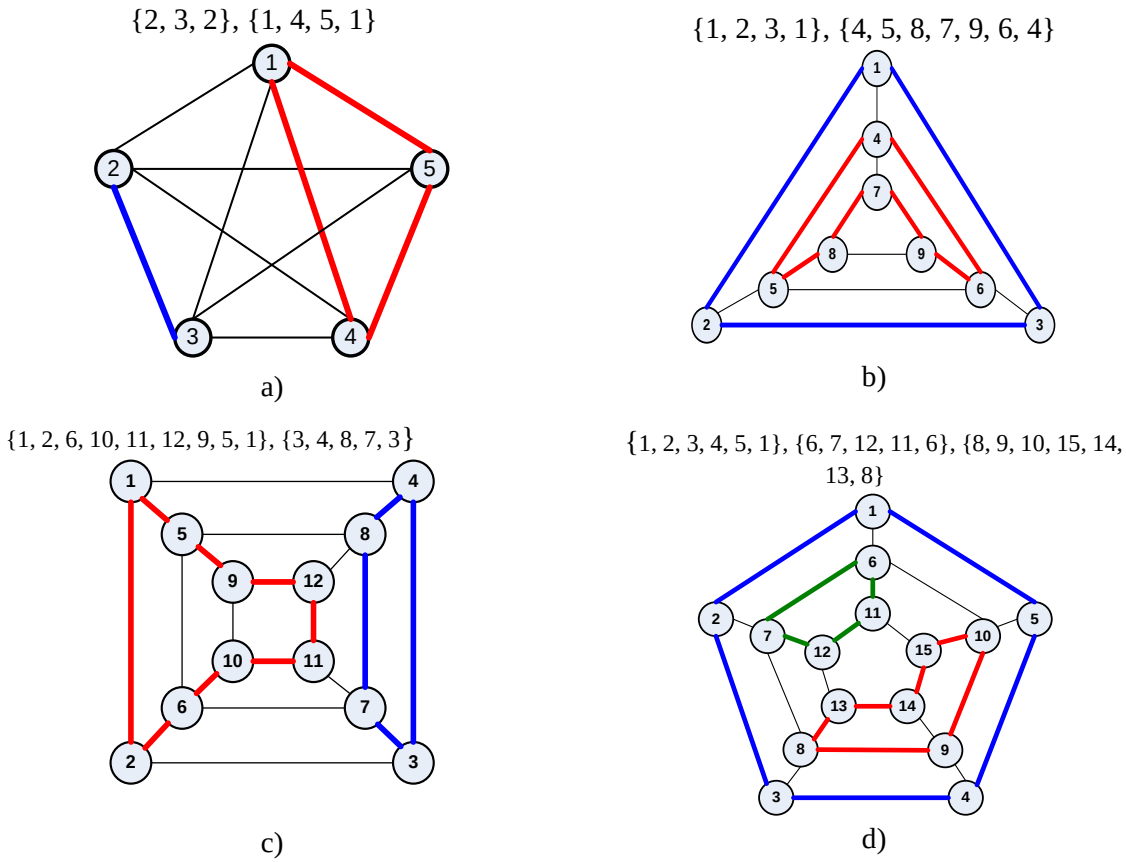



**Figure 3:** Self-organized isolated cycles of graphs

In Figure 3 a shows two possible variants of self-organizing isolated loops for a full-connected graph. An edge (2, 3) is a degenerate variant of a loop (minimal loop) where agent 2 has chosen vertex 3 and agent 3 has chosen vertex 2. Figures 3b – 3d shows variants of isolated cycles for inferior graphs of different structures.

To determine the Hamiltonian cycle, you must set the parameter close to 0. Then criterion (5) will have a predominant effect on the course of the game $\lambda$, which maximizes the lengths of local paths and leads to their asymptotic fusion in time into one global, Hamiltonian cycle.

Variants of Hamiltonian cycles, as a result of self-organization of the stochastic game, are shown in Figure 4 for different graphs. The game problem has a multivariate solution in pure strategies.

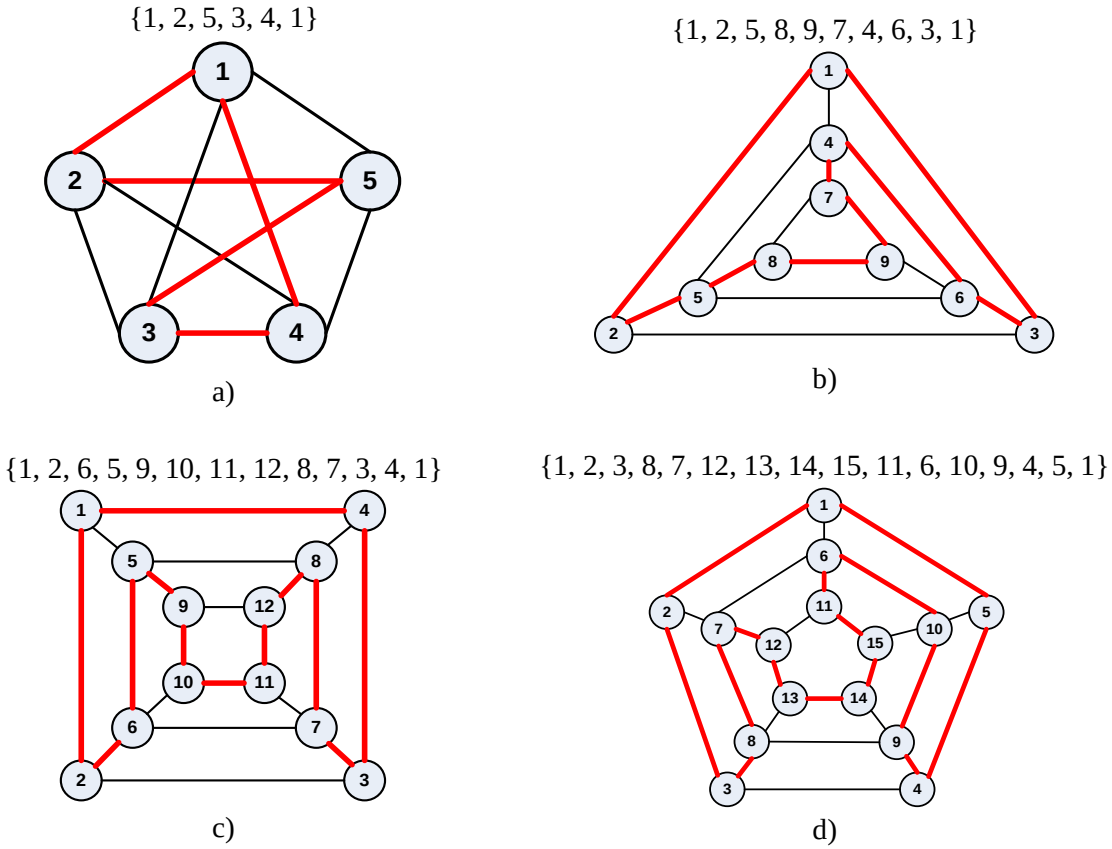



**Figure 3:** Self-organized Hamiltonian graph cycles

From the obtained results the simple interaction of agents within local subsets in the course of learning a stochastic game leads to a complex, coordinated behavior of the system, resulting in self-organizing patterns in the form of several autonomous or one Hamilton in the cycle of the graph. In the process of learning, the stochastic game moves from the initial chaotic choice of agents' strategies to their purposeful choice in the form of cycles.

Complicate the problem by expanding it to random graphs [40]. We apply the game method (13) to find the Hamiltonian cycle of a random graph. To do this, we assign to each vertex (or game agents) the probability $q^i$ of failure. Restorative failure of the vertex leads to the temporary loss of all its incident ribs. The agent corresponding to such a vertex skips the current move of the stochastic game, and neighboring players do not consider his choice strategy (since it is absent) to calculate their own current losses.

Criterion (3) is calculated only for those players (or vertices of the graph) and adjacent to them who have not failed for the current step of the game. Criterion (5) for determining the length of the local path, in addition to the condition of entering the cycle, additionally considers the condition of reaching the player who refused and was the first to happen on the way.

Several implementations of player strategies for a random full-connected graph are shown in Figure 5. In addition to vertex failures, it is similarly possible to introduce failures of the edges of the graph. In case of failures, a temporary violation of the connectivity of the graph is allowed. Dashed lines with arrows indicate strategies that participants in the game can choose when forming a path towards the rejected players. Used as a limiting condition for calculating current penalties (5).

In Figure 6 the graphs of the coefficient $K_n$ of coordination of players' strategies in the course of game self-organization of Hamilton cycles of a random graph, which is an implementation of a full-connected graph with $|V| = 5$ vertices, are given. The weighting factor of criteria (3) and (5) in

convolution (6) is $\lambda = 0.5$. The failure probabilities $q^i = q \forall i \in V$ are given the same for all vertices of the graph.

The graphs in Figure 6 are obtained for the following values of failure probabilities $q \in \{0; 0.05; 0.1; 0.15; 0.2\}$. For higher probability failures, the convergence time of the game method increases significantly.

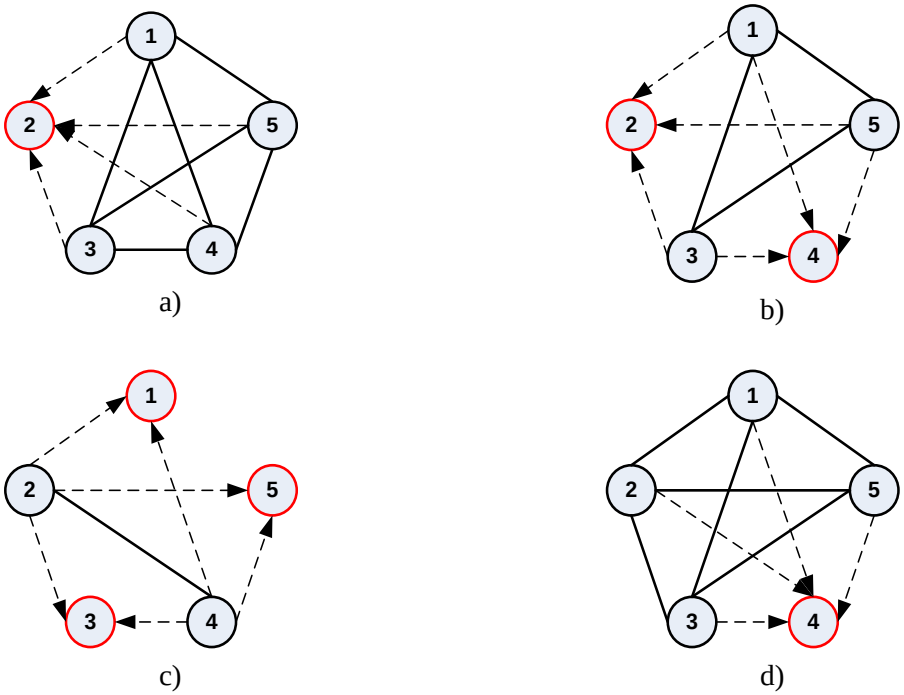



**Figure 5:** Implementations of a random graph of player strategies

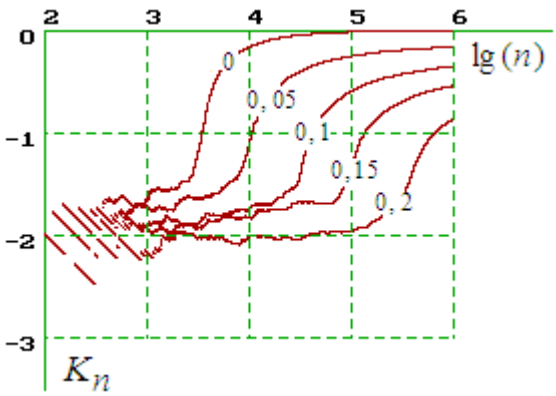



**Figure 6:** The dependence of the coordination coefficient on the probability of failure of vertices of a random graph

As can be seen in Figure 6, inmove players lead to an increase in the number of searches for their steps necessary for self-organization of the Hamilton cycle. This is due to the fact that the game method must additionally adapt to random implementations of a given graph.

To isolate the Hamiltonian cycle of a random graph foreach vertex, the conditional expectation of the edge number chosen by the corresponding agent at the current time is determined rounded to an integer value:

$$\bar{x}_n^i = int\left(\sum_{i=1}^{N_i} p_n^i x_n^i \,|x_t^i, \zeta_t^i (t = 1,2, \dots, n-1)\right).$$

During the adaptive game, at the $n \to \infty$ limit values of the mathematical expectations, the edge numbers are sent to one of the deterministic Hamiltonian cycles of the graph, for example, shown in Figure 4a.

The stochastic game method cannot compete in efficiency with the well-known meta odes of constructing Hamiltonian cycles and determinates graphs. It has other advantages and purposes – it can find Hamiltonian cycles of random graphs under conditions of uncertainty (the probability of failure of vertices of a graph is not known a priori). On the other hand, the method of stochastic play is a good illustration of self-organization Hamiltonian cycle based on the collection and processing of local data without exchanging information between all players. The solution of a stochastic game manifests itself in the form of a global pattern of self-organization of player strategies, which is one of the Hamiltonian cycles of the graph. The slow (power) convergence of the game is explained by its stochastic nature and the lack of information among players about the full structure of the graph or its random implementations. A stochastic game simulates the evolutionary process of self-organization of the Hamiltonian cycle through self-training of game agents.

Consequently, the self-organization of the considered stochastic game consists in the formation of patterns of strategies of game agents in the form of Hamiltonian cycles that arise in a deterministic or random graph in the process of learning the recurrent method (13) based on local interaction between agents, which leads to global coordination of the entire distributed system.

## 9. Conclusions

1. The complex problem of self-organization of Hamiltonian cycles of undirected graph based on model of stochastic game is solved.
2. Global Hamiltonian cycles arise because of purposeful locally conditioned choice of conflict-free strategies during adaptive learning of a stochastic game.
3. Self-organization of Hamiltonian cycles of a graph is possible if the constraints on the parameters of the game method are derived from the general conditions of stochastic approximation.
4. The considered game method provides a power order of the speed of convergence, requires a significant number of learning steps, since it works in conditions of incomplete a priori information.
5. Increasing the graph order leads to the deployment of the search process over a longer period and requires proper configuration of the parameters of the stochastic game.
6. The method of stochastic play as a method of random tests with adaptive data processing requires more game steps than known deterministic methods, but it can work with random graphs with a priori unknown distributions.
7. Compared to deterministic graphs, the search for Hamiltonian cycles in random graphs requires more steps of the game method, since at each step of the game another implementation of the connections between vertices of the graph is possible.
8. The stochastic game method of self-organization of Hamiltonian cycles can be used to build cryptographic protocols for exchanging keys, in systems of proof with non-disclosure of knowledge, for solving distributed flow and transport problems and collective decision-making under uncertainty.
9. A promising study in this direction is the game simulation of self-organization of Sensory networks, ring oscillation of signals in neural networks for the application of results in artificial intelligence systems.

## 10. References


[1] S. Gamazine, J.-L. Deneubourg, N. R. Frank, J. Sneyd, G. Theraula, E. Bonabeau, Self-Organization in Biological Systems, Princeton University Press, 2020.

[2] Z. Sun, Cooperative Coordination and Formation Control for Multi-agent Systems, Springer, 2018.

[3] P. Kravets, Game strategies for decision making in hierarchical systems: I. Mathematical model of the stochastic game, System Research, and Information Technologies, 3 (2019) 63–75. doi: 10.20535/SRIT.2308-8893.2019.3.06.
[4] P. Kravets, Game strategies for decision making in hierarchical systems: II. Computer simulation of the stochastic game, System Research, and Information Technologies, 4 (2019) 105–118. doi: 10.20535/SRIT.2308-8893.2019.4.11.
[5] W.J. Zhang (Ed.), Self-organization: Theories and Methods, USA: Nova Science Publishers, 2013.
[6] P. Kravets, Game model of self-organizing of multiagent systems, Bulletin of "Lviv polytechnic national university", Series: "Information systems and networks", 829 (2015) 161–176.
[7] P. Kravets, Game self-organization of agent's system with an individual estimation of strategies, Bulletin of "Lviv Polytechnic national university", Series: "Computer systems and networks", 546 (2005) 75–85.
[8] F. Schweisguth, F. Corson, Self-Organization in Pattern Formation: Review, Developmental Cell, 49(5) (2019) 659–677. doi: 10.1016/j.devcel.2019.05.019.
[9] P. Kravets, R. Jurinets, Y. Kis, Patterns of self-organizing strategies in the game of mobile agents, Bulletin of "Lviv polytechnic national university", Series: "Information systems and networks", 7 (2020) 24–34. doi: 10.23939/sisn2020.07.024.
[10] P. Kravets, Self-organizing strategies in the game of agent movement, Bulletin of "Lviv polytechnic national university", Series: "Information systems and networks", 9 (2021) 131–141. doi: 10. 23939.sisn2021.09.131.
[11] N. Christofides, Graph theory: an algorithmic approach, New York: Academic Press, 1975.
[12] K. R. Saoub, Graph Theory: An Introduction to Proofs, Algorithms, and Applications, Chapman and Hall/CRC, 2021.
[13] M. R. Garey, D. S. Johnson, R. Endre, The Planar Hamiltonian Circuit Problem is NP-Complete, SIAM Journal on Computing, 5(4) (1976) 704–714. doi: 10.1137/0205049.
[14] W. Alhalabi, O. Kitanneh, A. Alharbi, Z. Balfakih, A. Sarirete, Efficient solution for finding Hamilton cycles in undirected graphs, SpringerPlus, 5(1192) (2016) 1–14. doi 10.1186/s40064-016-2746-8.
[15] B. Korte, J. Vygen (Eds.), The Traveling Salesman Problem, in: Combinatorial Optimization: Algorithm and Combinatorics, Springer, Berlin, Heidelberg, 21 (2008) 527–562. doi: 10.1007/978-3-540-71844-4_21.
[16] A. Kumar, D. Mishra, A survey on existing conditions of Hamiltonian graphs, Journal of Mathematical Control Science and Applications, 7(1) (2021) 28–35.
[17] Ł. Waligóra, Application of Hamilton's graph theory in new technologies, World Scientific News, 89 (2017) 71–81.
[18] J.H. Seo, H. Lee, M.S. Jang, Optimal Routing and Hamiltonian Cycle in Petersen-Torus Networks, in: Proceedings of the Third 2008 International Conference on Convergence and Hybrid Information Technology, 2008, pp. 303–308.
[19] F. Sharifov, G. Jun, G. Kandiba, Optimization of routes of aircraft performing Argo-aviation works, Science-intensive technology, 3(23) (2014) 319–325.
[20] V. Lytvyn, D. Ugrin, Methods of solving problems of finding optimal tourist routes by ant colony imitation algorithms, Bulletin of the National Technical University "Kharkiv Polytechnic Institute" Collection of scientific works, Series: Computer Science and Modeling, Kharkiv: NTU "Kharkiv Polytechnic Institute", 21(1193) (2016) 47–60.
[21] Q. Zhang, R. Cheng, Z. Zheng, Energy-efficient renewable scheme for rechargeable sensor networks, EURASIP Journal on Wireless Communications and Networking, 74 (2020) 1–13. doi: 10.1186/s13638-020-01687-4.
[22] N. Xiong, W. Wu, C. Wu, An improved Routing Optimization Algorithm Based on Travelling Salesman Problem for Social Networks, Sustainability, 9 (2017) 1–15. doi: 10.3390/su9060985.
[23] P. Medvedev, M. Pop, What do Eulerian and Hamiltonian cycles have to do genome assembly? PLoS Computational Biology, 17(5):e1008928 (2021) 1–5. doi: 10.1371/journal.pcbi.1008928.
[24] O. Melkozerova, S. Rassomakhin, Identification of fingerprints based on Hamiltonian cycle of distribution of local features, Bulletin of V. N. Karazin Kharkiv National University, Series: Mathematical modelling, Information technology, Automated control systems, 44 (2019) 51–65. doi: 10.26565/2304-6201-2019-44-06.

[25] S. Kavun, I. Revak, Application of graph theory in communication management problems, Scientific Bulletin of Lviv State University of Internal Affairs, 2 (2015) 225–240.
[26] B. Barak, An Intensive Introduction to Cryptography, Chapter 13: Zero knowledge proofs, 2021. URL: https://intensecrypto.org/public/lec_14_zero_knowledge.html.
[27] L. Gulyanytsky, O. Mulesa, Applied methods of combinatorial optimization: Tutorial, Kyiv: Publishing and printing center "Kyiv University", 2016.
[28] Y. Peng, B. Choi, J. Xu, Graph Learning for Combinatorial Optimization: A Survey of State-of-the-Art, Data Science and Engineering, 6 (2021) 119–141. doi: 10.1007/s41019-021-00155-3.
[29] R. Kutelmakh, B. Uhrynovskyi, Investigation of the efficiency of common edges decomposition algorithm for solving large size traveling salesman problem, Young Scientist, 12(52) (2017) 1–5.
[30] J. Sleegers, D. Berg, Backtracking (the) Algorithms on the Hamiltonian Cycle Problem, arXiv:2107.00314v1 [cs.DS], 2021. URL: https://arxiv.org/pdf/ 2107.00314v1.pdf.
[31] V. Prokopenkov, A new method for finding a Hamiltonian cycle on a graph, Bulletin of the National Thecnical University "Kharkiv Polytechnic Institute", Series: Strategic management, portfolio management, and projects, 2 (2020) 43–49. doi: 10.20998/2413-3000.2020.2.6.
[32] T. Tambouratzis, Solving the Hamiltonian cycle problem via an artificial neural network, Information Processing Letters, 75(6) (2000) 237–242. doi: 10.1016/S0020-0190(00)00116-2.
[33] E. Ponce-de-Leon, A. Ochoa, R. Santana, A genetic Algorithm for a Hamiltonian Path Problem, in: Industrial and Engineering Application of Artificial Intelligence and Expert Systems, 2020, pp. 13–19.
[34] O. Muliarevych, V. Golembo, A modification of the ant colony method for solving the problem of a salesman by a team of autonomous agents, Bulletin of" Lviv polytechnic national university", Series: "Computer systems and networks", 717 (2011) 24 – 30.
[35] B.-S. Chen, Stochastic Game Strategies and their Applications, CRC Press, 2020.
[36] V. Ungureanu, Pareto-Nash-Stackelberg Game and Control Theory: Intelligent Paradigms and Applications, Springer, 2018.
[37] K. Najim, A.S. Poznyak, Learning Automata: Theory and Applications, Elsevier, 2014.
[38] H.J. Kushner, G.G. Yin, Stochastic Approximation Algorithms and Recursive Algorithms and Applications, Springer, 2013.
[39] P. Kravets, Convergence of the game gradient method in sign-positive environments, Bulletin of " Lviv polytechnic national university", Series: "Computer systems and networks", 438 (2001) 83 – 89.
[40] A. Frieze, Hamilton Cycles in Random Graphs: a bibliography, arXiv:1901.07139v20 [math.CO], 2021. URL: https://arxiv.org/pdf/1901.07139v20.pdf