Neuroagent Game Model of Collective Decision Making in Conditions of Uncertainty

Petro Kravets[1][0000-0001-8569-423X], Volodymyr Pasichnyk[1] [0000-0003-3007-2462], Olga Artemenko[2][0000-0002-4057-1217] Antonii Rzheuskyi[1] [0000-0001-8711-4163]

[1]Information Systems and Networks Department, Lviv Polytechnic National University, Lviv Ukraine
Petro.O.Kravets@lpnu.ua,
vpasichnyk@gmail.com, antonii.v.rzheuskyi@lpnu.ua
[2]Computer Systems and Technologies Department, PHEI “Bukovinian University”, Chernivtsi, Ukraine
olga.hapon@gmail.com

Abstract. The neuroagent game model of collective decision making in conditions of stochastic uncertainty is proposed. Neuroagents are based on artificial neural networks with feedback and learning without a teacher. The current collective decision is obtained after the independent choice of their own decision by all players. Each player generates a current version of decision according to values of the neural network outputs. The choice of decisions is carried out by neuroagents in a random way, regardless of time and other agents. Random choice involves calculation of the probabilities of choosing decisions by optimal design of neural network outputs for a single simplex. After choosing a collective decision, the reaction of decision making environment is determined as a set of values of current neuroagents’ gains. The current gain of each neuroagent is transmitted to the inputs of the appropriate two-layer neural network. Then, the training of neuroagents is carried out by changing the weights of neuronal relationships by one of the learning algorithms without a teacher. The learning process is repeated until the weights of the neurons relationships are stabilized with a given accuracy. The course of training is aimed at maximizing the average gains of neuroagents. The decision of the game is achieved in one of the points of the collective optimum or equilibrium, depending on the values of the parameters of the chosen method of neuroagents training.

Keywords: Collective Decision Making, Uncertainty Conditions, Neuroagent Stochastic Game, Adaptive Learning Methods.

1 Introduction to the problem of collective decision making in conditions of uncertainty

To solve the tasks of distributed management and decision making in technical, economic, informational and social systems, there is a need for a collective choice of decisions that satisfy one of the conditions of multicriteria optimality, such as Nash, Pareto, etc. [1-7]. These conditions in one way or another determine the degree of benefit and fairness of the collectively achieved decision.

The formal model of the system of collective choice of decisions is determined by managed environment and decision making managing agents. The environment model is determined by the structure and functions of the managed system.

The environment perceives agents' managing decisions and generates output signals which are interpreted by agents as estimates of efficiency of realized decisions. Agents get estimations from the environment, choose and implement decisions to maximize their own win or minimize losses. Such an approach to building a decision making system is called optimization [7]. In contrast to the situational approach, when the decision is obtained immediately on the basis of input data, in an optimization approach, the decision is refined through the feedback link.

An agent is an intelligent decision making system, software or technical implementation of the model of decision maker. Generally, the agent consists of the following main subsystems: receptor, intellectual and effector. The receptor subsystem receives signals from the decision making environment, intellectual makes decisions based on current information, self-study and forecasting, effector provides the implementation of decisions made with appropriate impact on the environment.

Collective decision making in a distributed environment is conducted by a set of interacting intelligent agents, which is called the multiagent system (MAS) [8-12]. Each agent makes a decision based on available local information and interaction within the MAS. As an active element of MAS, the agent is able to perceive and analyze data of the information network, negotiate and exchange current information with other agents, make stand-alone decisions, change the states of the network, inform the system and the user about the results of their actions.

The intelligent agents have the following basic properties:

1) autonomy – in pursuit of the goal, the agent can independently make decisions based on his own knowledge, without requiring external management;

2) reactivity – the agent can perceive external information, produce and implement adequate actions;

3) intelligence – the agent's ability to learn, adapt to environment changes, to process data obtained by artificial intelligence methods, to make optimal decisions;

4) specialization – as a rule, the agents perform highly specialized functions;

5) mobility – to achieve the goal, the software agent can move within the information network;

6) coordination – distribution of roles and coordination of actions of agents in solving common tasks;

7) interactivity – the agent interacts with other agents, with the information resources of the network and with the user;

8) communication – the ability of agents to communicate in technical language and understand each other;

9) personality – the unique qualities of the agent, modeling his psychological features, current emotions, etc.;

10) decentralization – each of the agents doesn’t have a complete imagination of the whole system and, therefore, there are no agents who manage the whole system.

When solving a task between agents there are phenomena of cooperation or competition. As it is known, these types of interaction between agents under conditions of uncertainty are studied by the theory of stochastic games [13–15]. Therefore, to study the collective behavior of agents in the decision making process, it is advisable to use the mathematical apparatus of the theory of stochastic games.

The typical features of the game decision making in the MAS are:

1) distribution or multiparameter of the decision making environment;
2) internal stochasticity of the environment;
3) complete or partial absence of a priori information (uncertainty) about the decision making environment;
4) manageability of the environment and the possibility of distributed implementation of management options;
5) multicriteria of management or decision making;
6) discreteness and finiteness of the set of decision making options;
7) stochastic independence of choice of decisions in space and time;
8) the possibility of multiple repetition of variants of decisions implementations in time;
9) distributed locally-dependent character of the formation and collection of information for statistical identification of the decision making environment;
10) the possibility of using a distributed game algorithm, which ensures the achievement of compromise decisions area;
11) implementation of the game algorithm in real time;
12) determination of the moments of stopping the game algorithm for the possibility of its practical application.

The functioning of the MAS decision making is conducted in conditions of a priori uncertainty [16]. Uncertainty can be caused by internal or external MAS factors. We distinguish the following types of uncertainties:

1) structural – unknown composition of the system and connections between its elements;
2) algorithmic – unknown algorithm of the system functioning;
3) information-fuzziness, lack of complete information necessary for decision making;
4) linguistic – ambiguity of statements in the exchange of messages between agents;
5) target – an unknown global purpose of the system;
6) social – due to the collective interaction of agents, when the actions of one of the agents influence the choice of decisions by other agents;
7) stochastic – the influence on the system of uncontrolled external factors.

In scientific research, the uncertainty of decision making is most often modeled with the use of mechanism of random variables, which is the basis of stochastic uncertainty. Partial compensation of uncertainty is provided by the ability of agents to self-learning and adaptive decision making strategies.

In the cybernetic literature, game methods of self-learning based on the adaptive formation of probability distributions of discrete variants of decisions (pure strategies of players) are well studied [14]. After chosing and implementation of pure strategies by all players is completed, each of them receives a current win from the environment, which is used to change the dynamic mixed strategies (conditional probabilities vectors of the choice of decisions), which are the basis for the mechanism of generating random pure strategies. This change means that the probability of choosing a clean strategy increases proportionally to the value of the current win. The method of reconstructing mixed strategies with the passage of time built on the basis of stochastic approximation provides maximization of the functions of average gains on unit simplexes. The intellectual capabilities of agents of such game are limited, because they model only the reflex behavior of biological systems.

Artificial neural networks (ANN) can be used to construct intellectual decision making systems as models of processes of the nervous activity of biological systems with the ability to memorize, analyze and predict behavior [17–20]. Neural networks implement "soft" calculations based on the processes occurring in the human brain and are used as models of objects with unknown characteristics. ANN consist of many neurons and the connections among them. Training of ANN is to correct the synaptic connections among neurons based on information which enters the neural network from the environment. To obtain the necessary structure of connections, some connections among neurons are amplified while others are weakened.

The decision making system, based on ANN, is called a neuroagent. Neuroagent game models of decision making in conditions of uncertainty are insufficiently studied in modern professional scientific literature.

The application of neuroagent game models is a promising direction for increasing the efficiency of collective development and decision making processes in conditions of uncertainty due to the following features:

1) nonlinearity – neural networks allow to obtain a nonlinear dependence of the output signal from the input;

2) adaptability – neural networks have the ability to adapt their synaptic weights to environmental changes;

3) plasticity and resistance to failures – neural networks store information in distributed form over all connections of the neural network. Failure of one or more neurons does not lead to failure of the system as a whole;

4) universality – neural networks do not require special programming, because they allow to solve various tasks of information processing with the same algorithms of neurons training.

The aim of this work is to develop a model of stochastic game of neuroagents for collective decision making in conditions of uncertainty. To achieve this aim it is necessary: to set the task of game decision making in conditions of uncertainty; to set the environment for collective decision making; develop the structure of neuroagents; to choose the method of teaching neuroagents to solve the formulated task; to develop algorithm and software tools for simulation of stochastic game of neuroagents; to analyze the obtained results and make recommendations for their practical use.

2 Setting up of the Game Task of Decision Making In Conditions Of Uncertainty

Matrix stochastic game $G = (I, \{U_i\}_{i\epsilon I}, [V_i])$ of decision making is given by:

1) multitude of agents $I = \{i \,|\, i = 1..L\}$, where $L = |I|$ is a power of multitude or number of players;

2) set of pure strategies of agents $U_i = \{u_i(1), u_i(2), ..., u_i(N_i)\}$, where N_i is quantity of pure player's strategies with number i;

3) matrices of the average gains of agents $[v_i(u)]$ $\forall i \in I$, $\forall u \in U$, where $u \in U = \underset{i \in I}{\times} U_i$ is collective player's strategies.

The stochastic game takes place in discrete moments of time $t = 1, 2, ...$. After the implementation of the collective strategy $u_t = u$ each agent gets a current random gain $\xi_{i,t}(u) \in R^1$ with unknown mathematical expectation $M\{\xi_{i,t}(u)\} = v_i(u)$ and limited dispersion $d_i(u) < \infty$. Here and below the index i is the number of the player, and the index t is the current time.

The obtained current agent's gains are averaged over time to evaluate the effectiveness of the decision making process by each agent:

$$\Xi_{i,t}(\{u_t\}) = t^{-1} \sum_{\tau=1}^{t} \xi_{i,\tau} \quad \forall i \in I . \tag{1}$$

The aim of the agents is to maximize their average payment functions:

$$\varliminf_{t \to \infty} \Xi_{i,t} \to \max_{u_i} \quad \forall i \in I . \tag{2}$$

Thus, based on observations of current gains $\xi_{i,t}$ each agent $i \in I$ must choose current decisions $u_{i,t} = u_i \in U_i$ so that with the passage of time $t = 1, 2, ...$ to ensure maximization of system of target functions (1).

The decision of the multicriterial problem (2) should be sought in the set of points of collective equilibrium (for example, Nash) or optimality (for example, Pareto) depending on the way of agents' choice the sequences of decisions variants.

3 Game Neuroagent Method of Task Solution

Known adaptive sequence generation methods $\{u_{i,t}\}$ $\forall i \in I$, $t = 1, 2, ...$ based on dynamic distributions of discrete random variables, which use mixed players' strategies [14]. In contrast, let's consider the neuroagent method for solving a matrix stochastic game, the scheme of which is shown in Fig. 1

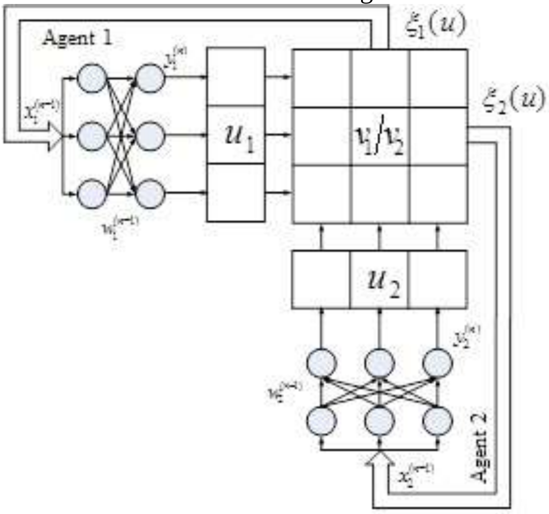

Fig. 1. The structure of the matrix game of neuroagents.

The model of the decision making environment is given by the matrices of the mathematical expectations of random gains v_i $\forall i \in I$. The values of the decision variants are presented to the input of the environment $u_i \in U_i$. The output of the environment is the corresponding values of the current gains $\xi_i(u)$.

Each neuroagent is assigned an artificial network with $n = 2$ layers of neurons. The number of elements of each layer is the same, equal to the number of decisions $N_i = |U_i|$. The vectors of the parameters of the neuroagents are presented to the inputs $x_i^{(n-1)}$, calculated on the basis of outputs of the environment $\xi_i(u)$ $\forall i \in I$. The outputs of neuroagents are vectors of the parameters $y_i^{(n)}$, on the basis of which decision variants are determined $u_i \in U_i$ $\forall i \in I$. Weights $w_i^{(n-1)}$ indicate the value of synaptic connections among neurons of i -th agent. The positive values of weights of connections correspond to exciting and negative values correspond to inhibitory synapses. Zero value of weights means the absence of connection among the neurons.

The functioning of a neuroagent is carried out by one of the adaptive algorithms of unsupervised learning, for example, Hebb, Kohonen or others [19]. Unsupervised learning or self-learning is by nature closest to the brain as its biological prototype.

Self-learning is not oriented to the presence of correct outputs of neural network. The self-learning algorithm independently detects the internal structure of input data, rebuilding the weights of synaptic connections so that close (by some metric) sets of input signals cause sufficiently close output sets of signals. In fact, the process of neural network self-learning solves the task of data clustering, identifying the statistical properties of learning multitudes and grouping similar output multitudes into clusters. By entering a vector from given class to the input of trained neural network, we obtain the characteristic output vector for this class. The output vector is not known in advance. Its formation is due to the structure of the training sample, random distribution of initial values of weights of connections among neurons and a combination of excited neurons of the output layer of the neural network.

Neuroagents carry out a random choice of decisions $u_{i,t} \in U_i$ independently of each other $\forall i \in I$ and in time $t = 1, 2, ...$. To do this, each neuroagent builds a vector of conditional probabilities $p_{i,t}$ choice of decisions $u_{i,t}$ by designing the output vector $y_{i,t}^{(n)}$ on a single N_i -measurable ε -simplex:

$$p_{i,t}(u_{i,t} \mid u_{i,\tau}, \xi_{i,\tau}, \tau = 1, 2, ..., t-1) = \pi_{\varepsilon_t}^{N_i}\{y_{i,t}^{(n)}\} \quad \forall i \in I \ , \tag{3}$$

where $\pi_{\varepsilon_t}^{N_i}$ is a projector on a single ε -simplex $S_{\varepsilon_t}^{N_i} \subseteq S^{N_i}$ [14]. Parameter ε_t adjusts the speed of expansion of ε -simplex $S_{\varepsilon_t}^{N_i}$ to a single simplex S^{N_i} and can be used as an additional factor in controlling of convergence of game neuroagent decision making method. The obtained probability vector is used to construct an empirical distribution of discrete random variables, on the basis of which a choice of decisions is made:

$$u_{i,t} = \left\{ u_i[k] \,\middle|\, k = \arg\left(\min_k \sum_{j=1}^{k} p_{i,t}(u_{i,t}[j]) > \omega \right),\ k = 1..N_i \right\} \quad \forall i \in I \ , \tag{4}$$

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

The response of decision making environment to the chosen version is the value of a random variable with an unknown distribution Z , which is interpreted as the agent's current gain:

$$\xi_i(u_t) \sim Z(v_i(u_t), d_i(u_t)) \ ,$$

where $v_i(u_t)$ is mathematical expectation, $d_i(u_t)$ is dispersion.

The obtained current gains $\xi_{i,t}(u_t)$ are submitted to the inputs of neuroagents

$$x_{i,t}^{(n-1)} = e\xi_i(u_t) \quad \forall i \in I \ , \tag{5}$$

where $e = \left(1 \mid \forall u_i \in U_i\right)$ is the vector, all elements of which are equal to one. If necessary, normalization of vector elements is carried out $x_{i,t}^{(n-1)}$, for example, such as:

$$x_{i,t}^{(n-1)} = e\xi_i(u_t)/|\xi_{\max}| \ \ \forall i \in I \ ,$$

where $\xi_{\max}$ is the maximum value of current gains. Normalization can reduce the number of steps required to train the neuroagent.

Total inputs $\overline{x}_{i,t}^{(n)}$ of neurons of n-th layer are calculated based on outputs $y_{i,t}^{(n-1)}$ of neurons of $(n-1)$ -th layer:

$$\overline{x}_{i,t}^{(n)}[k] = \sum_{j=1}^{N_i} w_{i,t}^{(n-1)}[j,k] y_{i,t}^{(n-1)}[j] \ \ \forall i \in I \ , \ k = 1..N_i \ , \qquad (6)$$

where $w_{i,t}^{(n-1)}[N_i, N_i]$ is the matrix of the weights of connections among the nodes of the neural network, calculated at the moment of time t. Here $w_{i,t}^{(n-1)}[j,k]$ indicates the weight of connection between j -th node $(n-1)$ -th layer and k -th node of n -th layer. To calculate outputs $y_{i,t}^{(n)}$ of neuroagent a transfer function $\varphi()$ of neuron is used:

$$y_{i,t}^{(n)}[k] = \varphi(\overline{x}_{i,t}^{(n)}[k]) \ , \qquad (7)$$

where $k = 1..N_i$.

Depending on the task being solved and the type of neural network, the transfer function $\varphi()$ can be threshold, linear with saturation, sigmoidal, sinusoidal, radially symmetrical, and so on. Most often for modeling an artificial neural network, linear

$$y_{i,t}^{(n)}[k] = \begin{cases} 0, & if \ \ \overline{x}_{i,t}^{(n)}[k] \le \theta, \\ \eta(\overline{x}_{i,t}^{(n)}[k] - \theta), & if \ \ \overline{x}_{i,t}^{(n)}[k] > \theta \end{cases} \qquad (8)$$

bipolar sigmoidal

$$y_{i,t}^{(n)}[k] = -0.5 + 1/(1 + \exp(-\eta(\overline{x}_{i,t}^{(n)}[k] - \theta))) \qquad (9)$$

transfer functions are used. Parameter $\eta > 0$ defines the tangent of the slope angle for the linear transfer function and the level of steepness for the sigmoidal transfer function. Parameter $\theta \ge 0$ defines the threshold of neuron activation.

Teaching the neuroagent is carried out by changing the weights $w_{i,t}^{(n-1)}$ of synaptic connections among neurons. Recalculation of the weights of connections is performed according to Hebb’s signal method, Kohonen’s method or another method of unsupervised learning.

Learning by Hebb’s method leads to increase of relationships among excited neurons:

$$w_{i,t}^{(n-1)}[j,k] = w_{i,t-1}^{(n-1)}[j,k] + \gamma_t(y_{i,t-1}^{(n-1)}[j] * y_{i,t-1}^{(n-1)}[k]) \ , \ \ j = 1..N_i \ , \ k = 1..N_i \ , \qquad (10)$$

where γ_t is the parameter of neuroagent training step.

Excited are called neurons for which the value of the total input $x_i^{(n)}$ exceeds the activation threshold θ.

Training by Hebb's differential method leads to increasing of connections among those neurons, the outputs of which have changed the most:

$$w_{i,t}^{(n-1)}[j,k] = w_{i,t-1}^{(n-1)}[j,k] + \gamma_t (y_{i,t}^{(n-1)}[j] - y_{i,t-1}^{(n-1)}[j]) * (y_{i,t}^{(n-1)}[k] - y_{i,t-1}^{(n-1)}[k]),$$
$$j = 1..N_i, \ k = 1..N_i. \qquad (11)$$

Training by Kohonen's method is based on the mechanism of competition, the essence of which is to minimize the difference between the input signals of a neuron-winner, coming from the outputs of the neurons of the previous layer, and the weight coefficients of its synapses:

$$w_{i,t}^{(n-1)}[j,k_i^*] = w_{i,t-1}^{(n-1)}[j,k_i^*] + \gamma_t (y_{i,t-1}^{(n-1)}[j] - w_{i,t-1}^{(n-1)}[j,k_i^*]), \ j = 1..N_i, \qquad (12)$$

where k_i^* is index of the neuron-winner of i-th agent.

In contrast to Hebb's method, in which simultaneously several neurons of the same layer can be excited, in Kohonen's method, neurons of the same layer compete with each other for the right of activation. This rule is known in literature on machine learning as "winner takes it all".

According to Kohonen's method the restructuring of weights of connections is carried out only for neuron-winner. The winner is the neuron whose synapse values are as similar as possible to the input image.

The neuron-winner is determined by calculating the distance between the vectors $y_{i,t-1}^{(n)}$ and $w_{i,t-1}^{(n-1)}$:

$$D_{i,t-1}[k] = \sqrt{\sum_{j=1}^{N_i} \left(y_{i,t-1}^{(n-1)}[j] - w_{i,t-1}^{(n-1)}[j,k] \right)^2}, \ j = 1..N_i.$$

The winner is the neuron with the smallest distance:

$$k_i^* = index \min_{k=1..N_i} (D_{i,t-1}[k]).$$

Another way to determine the neuron-winner is to maximize the outputs of $y_{i,t-1}^{(n)}$ neurons of n-th layer:

$$u_{i,t-1} = \left(u[k] \middle| k = \arg \max_{j=1..N_i} y_{i,t-1}^{(n)}[j] \right).$$

In this case, the index of the neuron-winner is a serial number of the chosen variant of decision $u_{i,t-1}$:

$$k_i^* = index(u_i[k] \mid \chi(u_i[k] = u_{i,t-1}), k = 1..N_i) \ \ \forall i \in I, \qquad (13)$$

$\chi(\ ) \in \{0,1\}$ is the indicator function of event.

The training radius can be set around the neuron-winner R in space of the vectors of weights of neurons:

$$r_{i,t}[k] = \left\| w_{i,t}^{(n-1)}[k_i^*] - w_{i,t}^{(n-1)}[k] \right\|, \; k = 1..N_i \,,$$

where $w_{i,t}^{(n-1)}[k_i^*]$ is the vector of weights of the neuron-winner; $\|*\|$ is Euclidean vector norm.

Each neuron whose distance from the vector of weights to the vector of weights of the neuron-winner is less than the radius of training ($r_{i,t}[k] < R$), takes part in the calculation of weights of synapses. The weights of neurons that are outside the training radius do not change. The training radius decreases in time so that at the end of the training process correction of the weights of connections can be carried out only by one neuron-winner.

The parameters γ_t y (10) – (12) and ε_t y (3) determine the learning rate of neuroagents.To ensure the convergence of neuroagents training process, these parameters are set as positive monotonously decreasing values:

$$\gamma_t = \gamma_0 / t^\alpha \,, \; \varepsilon_t = \varepsilon_0 / t^\beta \,, \qquad (14)$$

where $\gamma_0, \alpha > 0$; $\varepsilon_0, \beta > 0$.

The choice of decision options continues to a specified number of steps $t \le t_{\max}$, or to fulfillment the condition of the training accuracy:

$$\delta_t = | I |^{-1} \sum_{i \in I} \left\| w_{i,t}^{(n-1)} - w_{i,t-1}^{(n-1)} \right\| < \varepsilon \quad \forall i \in I \,, \qquad (15)$$

where ε is the neuroagents training accuracy, which is determined by Euclidean norm of change the weights of connections among neurons for two consecutive moments of time.

4 Indicators of Stochastic Game Effectiveness

For practical applications, it is necessary to determine the indicators of the game effectiveness, which can be used to evaluate the convergence of the game method. In the absence of direct information exchange between players, such indicators are formed on the basis of the collective equilibrium condition according to Nash [1, 14, 21]. To do this, let’s define the functions of the average players’ gains:

$$V_i(p) = \sum_{u \in U} v_i(u) \prod_{j \in I; u_j \in u} p_j(u_j) \,, \qquad (16)$$

where $p \in S^I$, $S^I = \prod_{j \in I} S^{N_j}$, $v_i(u) = M\{\xi_{i,t}(u)\}$, and the values $p_j \in S^{N_j}$ are determined according to (3).

The Nash equilibrium determines the following strategies for solving the game, for which the condition is performed:

$$\forall i \in I \quad V_i(p^*) - V_i(p^{I \setminus i*}, p_i) \geq 0\,, \tag{17}$$

where $p^* \in S^I$ is the optimal collective mixed players' strategy; $V_i(p^{I \setminus i*}, p_i)$ is the function of average gains, defined on the simplex S^I at random deviation of the mixed strategy of the i -th player from the equilibrium point by Nash within the unit simplex. The optimal by Nash leveling mixed strategies can be obtained from the condition of complementary non-rigidity [21]:

$$\nabla_{p_i} V_i = V_i e^{N_i} \quad \forall i \in I\,, \tag{18}$$

where $p_i \in S^{N_i}$ is the mixed strategy of the i -th player; $\nabla_{p_i} V_i$ is the gradient of polylinear function of average gains (16); $e^{N_i} = (1_j \mid j = 1..N_i)$ is a vector, all elements of which are equal to 1.

To account for solutions on the boundary of a simple simplex, let's perform the weighing of the vector condition of complementary non-rigidity with the elements of the vectors of mixed strategies:

$$diag(p_i)(\nabla_{p_i} V_i(p) - e^{N_i} V_i(p)) = 0 \;\; \forall i \in I\,, \tag{19}$$

where $diag(p_i)$ is the square diagonal matrix of N_i order, constructed from elements of the vector p_i ; $p \in S^I$ is combined mixed players' strategies, set on convex simplexes of S^I .

Let's define the Lyapunov's function as the total current players' error during the search for a point of equilibrium according to Nash:

$$\Delta_t = |I|^{-1} \sum_{i \in I} \Delta_{i,t}\,, \tag{20}$$

where $\Delta_{i,t} = \left\| p_{i,t} - \tilde{p}_{i,t} \right\|^2$, $p_{i,t}, \tilde{p}_{i,t} \in S^{N_i}$. Values $\Delta_t \geq 0 \;\; \forall t = 1,2,...$ turns to zero at the points of equilibrium according to Nash, which can be achieved both inside and on the vertices of single simplexes. The vectors $p_{i,t}$ are determined according to (3), and $\tilde{p}_{i,t}$ is calculated so:

$$\tilde{p}_{i,t}(j) = V_{i,t}(j) / V_{i,t}\,, \;\; j = 1..N_i\,,$$

where $V_{i,t}(j) = p_{i,t}(j) \nabla V_{i,t}(j)$.

Direction Δ_t to zero at $t = 1,2,...$ will indicate the convergence of the game method to one of the equilibrium points according to Nash in mixed strategies or taking into account (19) the achievement of the game solution in pure strategies.

The order θ and value ϑ of the convergence rate of the game method can be evaluated using the asymptotic method of Jung's moments [22]:

$$\overline{\lim_{t\to\infty}} n^{\theta} M\{\Delta_t\} = \vartheta . \tag{21}$$

The optimization procedure will be better, than greater θ and less ϑ.

In practice, we can use an approximate evaluation to determine θ, performing the logarithm of the expression (21):

$$\theta \approx \frac{\lg(M\{\Delta_t\})}{\lg(t)} . \tag{22}$$

That is the parameter θ can be defined as an angle tangent ϕ of slope of the linear approximation of the function $M\{\Delta_t\}$ in a logarithmic coordinate system. The course of a stochastic non-antagonistic game can be traced also by changing the function of average player's gains:

$$\Xi_t = |I|^{-1} \sum_{i \in I} \Xi_{i,t} . \tag{23}$$

5 Kohonen's Algorithm of Neuroagent Functioning

1. Set the initial parameters values:
$t = 0$ is the initial time point;
I is a multitude of players;
N_i $\forall i \in I$ is a number of decisions options;
$U_i = \{u_i[1], u_i[2], ..., u_i[N]\}$ $\forall i \in I$ is a multitude of decisions options;
$[v_i]$ $\forall i \in I$ is the matrices of mathematical expectations of gains;
$[d_i]$ $\forall i \in I$ is the matrices of dispersions of gains;
$w_{i,0}^{(n-1)}[N_i, N_i]$ $\forall i \in I$ is the matrix of initial weights of connections among nodes of the neural network;
η, θ is parameters of the transfer function of the neurons;
γ_0 is parameter of training step;
$\alpha \in (0,1]$ is the order of training step;
ε_0 is parameter of ε-simplex;
β is the order of the speed of expansion of ε-simplex;
$t_{\max}$ is the maximum number of steps of the method;
ε is accuracy of training.

2. Perform choices of decision variants $u_{i,t}$ $\forall i \in I$ according to $(3)-(4)$.

3. Obtain the value of current gains of neuroagents as random variables with normal distribution $\xi_i(u_t) \sim Normal(v_i(u_t), d_i(u_t))$ $\forall i \in I$.

4. Calculate the inputs of neuroagents $x_{i,t}^{(n-1)}$ $\forall i \in I$ according to (5) and the corresponding outputs of $y_{i,t}^{(n-1)}$ $\forall i \in I$ neurons $(n-1)$-th layer according to (7).

5. Calculate the total inputs of neurons $\overline{x}_{i,t}^{(n)}$ $\forall i \in I$ (6) and the corresponding outputs $y_{i,t}^{(n)}$ $\forall i \in I$ (7) for neurons of n-th layer.

6. Calculate parameter value of γ_t according to (14).

7. Define k_i^* indexes of the winning neurons $\forall i \in I$ according to (13).

8. Calculate the weight of connections to neurons winners $w_{i,t}^{(n-1)}[j,k_i^*]$ $\forall i \in I$, $j = 1..N_i$ according to (12).

9. Calculate the characteristics of the quality of collective decision making Ξ_t (23), Δ_t (20), θ (22).

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

11. If the condition (15) of the end of the game is not met, then go to step 2, otherwise – the end.

6 Results of Computer Modeling

Let's solve the stochastic game of two neuroagents $|I| = 2$ with two clean strategies $N_i = 2$, $i = 1..2$. Average gains matrices $[v_i]_{2\times 2}$ $i = 1..2$ such a game is given in Table. 1.

Table. 1. Player gains matrix

Strategies	**First Player**		**Second Player**	
$\times$	$p_2(u_2[1])$	$p_2(u_2[2])$	$p_2(u_2[1])$	$p_2(u_2[2])$
$p_1(u_1[1])$	0.1	0.6	0.1	0.9
$p_1(u_1[2])$	0.9	0.4	0.9	0.1

The view of slices of functions of average gains of neuroagents (17) with values on the unit simplex, corresponding to Table 1, is shown in Fig. 2.

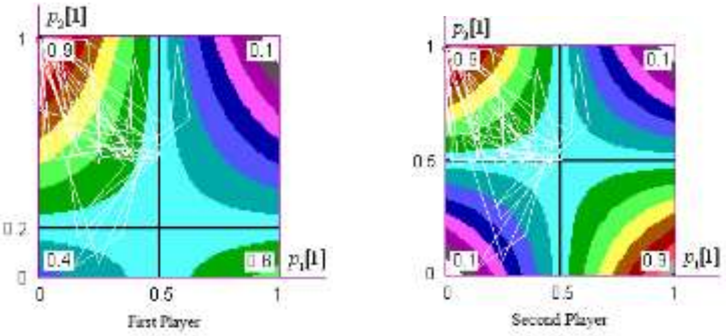

Fig. 2. Functions of average gains of neuroagents

Analysis of average win functions shows that the game has one Nash solution in mixed $(p_1[1], p_2[1]) = (0.5, 0.2)$ and two solutions in pure strategies $(p_1[1], p_2[1]) = (0,1)$, $(p_1[1], p_2[1]) = (1,0)$.

Under conditions of uncertainty, the elements of matrices of average winnings $\left[v^i(u)\right]_{\forall u \in U}$ are unknown and available for observation in the form of random current values $\xi_i(u)$. To simulate random gains we choose the normal distribution law:

$$\xi_i(u) = Normal(v_i(u), d_i(u)),$$

where $v^i(u)$ is a mathematical expectation; $d_i(u)$ is a dispersion. Normally-distributed random variables are calculated using the sum of uniformly distributed random real numbers $\omega \in [0,1]$:

$$\xi_i(u_t) = v_i(u_t) + \sqrt{d_i(u_t)}\left(\sum_{j=1}^{12} \omega_{i,t}[j] - 6\right).$$

Initial values of link weights $w_0^{(n-1)}$ between neurons are random variables uniformly distributed in the interval [0, 1]. The linear dependence of the output on the total inputs is chosen as the transfer function of neurons (8).

The convergence of the gaming neuroagent method is determined by the exact ratio of parameters, which in the general case must satisfy the basic conditions of stochastic optimization [22–28]. The parameters of the neuroagent training method take the following values: $\gamma = 1$; $\alpha = 0{,}1$; $N_i = N = 2$; $\varepsilon = 0.999 / N$; $\beta = 1$.

The graphs of functions of average gains are shown in logarithmic scale in Fig. 3 $\Xi_{i,t}$ and the error of achieving an optimal collective solution at one of the equilibrium points $\Delta_{i,t}$ according to Nash.

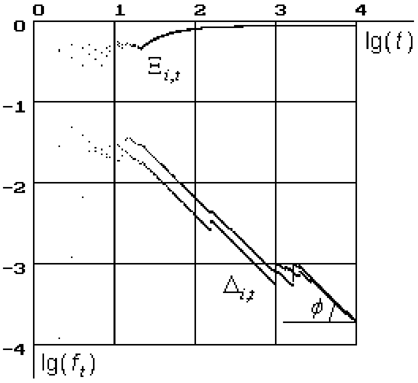

Fig. 3. Characteristics of neuroagent training

Increase $\Xi_{i,t}$ and reduction $\Delta_{i,t}$ in time indicate the convergence of the neuroagent method of decision-making in the sense of fulfilling the condition of complementary rigidity (19).

The trajectories of change of conditional probabilities of collective choice of variants of decisions within a single simplex are shown in Fig. 2. From the obtained data it can be seen that for the given parameter values, the game neuroagent method (12) provides the solution of stochastic game on top of a single simplex $(p_1[1], p_2[1]) = (0,1)$ and has close to1 order $\theta \approx tg(\phi)$ of convergence speed.

For the convergence of the stochastic game of neuroagents to Nash equilibrium points with probability 1 or in medium-quadratic, it is necessary that the ratio of parameters of the recurrent game method primarily satisfies the fundamental conditions of stochastic approximation [29, 30]. In conditions of uncertainty the theoretical study of convergence is based on the stochastic Robins-Monroe approximation, the results of the Robin-Sigmund’s lemma, and depends on both the parameters of the environment and the values of the parameters of the game method [31].

Recurrent game algorithms provide the order degree of convergence rate convergence rate [31] and are easy to program. The low rate of convergence is due to the lack of a priori information about players' payment matrices and the lack of direct exchange of information between players. In addition, each player does not have information about the structure of the game (the number of players, the number of strategies, dependence of own payments on the strategies of other players). During the multi-step game, each player learns to choose pure strategies so as to optimize own function of average payments.

The learning process is a reorganization of one's own mixed strategies over time – by increasing the probability of choosing pure strategies that, on average, produce the best results.

Theoretical evaluations of the conditions of convergence of game algorithms in conditions of uncertainty are complex and difficult to analyze in a rigorous analytical way. In addition, such convergence conditions are evaluations from above and therefore, there is a question about their accuracy. In this regard, the theoretical results will not always be satisfactory and require experimental clarification. It is experimentally established that scaling the system in the direction of increasing the number of players and the number of pure strategies leads to an increase in the entropy of a stochastic game, and as a result is a slowing down of its rate of convergence.

Let's study the dependence of the learning time of stochastic game of neuroagents on the basic parameters of the algorithm. We define training time as the minimum number of steps required to train neuroagents with a given accuracy $\varepsilon > 0$:

$$t_{out} = (t = t_{\min} \mid \delta_t < \varepsilon) ,$$

where the current training accuracy δ_t is calculated according to (15).

Due to the random choice of solutions, we need to average the training time of neuroagents for different sequences of random variables:

$$\overline{t} = \frac{1}{k_{\exp}} \sum_{j=1}^{k_{\exp}} t_{out}[j],$$

where $k_{\exp}$ is a number of experiments.

The average number of training steps $\overline{t}$ depends on the parameters of the training algorithm of neuroagents and the parameters of the decision-making environment.

Graph of average time dependence $\overline{t}$ of neuroagents training from the parameter α is shown in Fig. 4. The results obtained for dispersion value of current gains $d = 0.01$. The parameter $\alpha \in (0,1]$ determines the order of monotonic decrease of the value $\gamma_t > 0$ (14), which regulates the speed of training of neuroagents. As the value α increases, the γ_t value decreases.

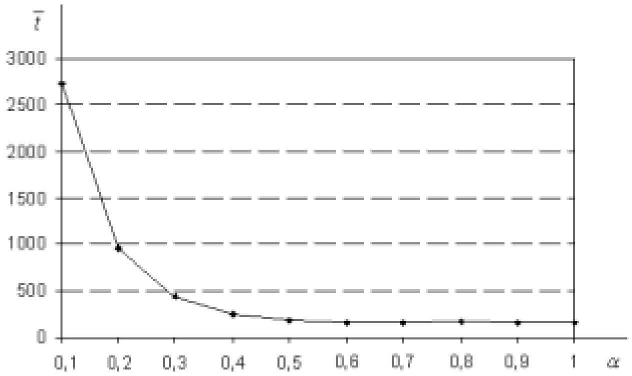

Fig. 4. The influence of α parameter on the training time of neuroagents

The accuracy of neuroagents training of game is $\varepsilon = 10^{-3}$. The data are averaged by $k_{\exp} = 100$ experiments. In all experiments, the choice of pure decision strategies for which condition (19) is met is provided with high probability.

As can be seen from the simulation results, the increase of α parameter, leads to a decrease in the average number of training steps $\overline{t}$ of neuroagent training with accuracy ε.

The dependence of the average number of steps $\overline{t}$ of training neuroagents on the dispersion d of the stochastic environment (dispersion of evaluations of variants of solutions) is presented in Fig. 5. The results are obtained for the value of order of step of the neuroagent training $\alpha = 0.5$.

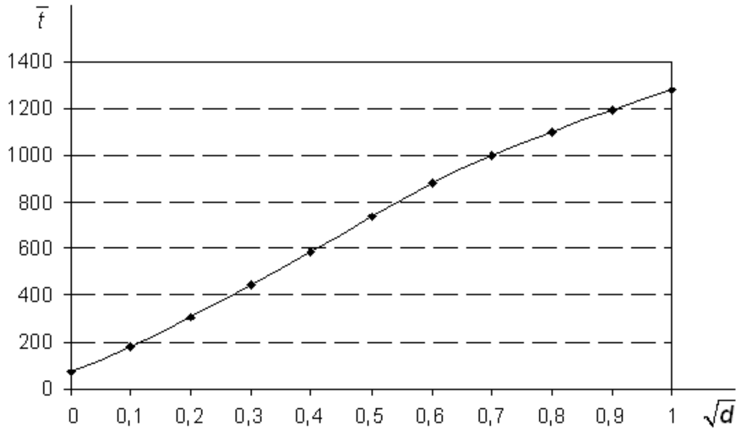

Fig. 5. The influence of dispersion of the estimates of variants of decisions for training of neuroagents.

As dispersion increases, the average number of steps required for training neuroagents increases. A significant increase in dispersion may lead to an incorrect determination of the optimal decision. Modeling of stochastic game with other matrices of average gains of neuroagents can show a different result, preserving the obtained dependencies among the parameters, required to optimize the payment functions of the players.

Conclusions

The neuroagent game model of collective decision making in conditions of stochastic uncertainty is proposed in this article. Neuroagents are based on artificial neural networks with feedback and learning without a teacher. The current collective decision is obtained after the independent choice of their own decision by all players. Each player generates a current version of decision according to values of the neural network outputs. The choice of decisions is carried out by neuroagents in a random way, regardless of time and other agents. Random choice involves calculation of the probabilities of choosing decisions by optimal design of neural network outputs for a single simplex. After choosing a collective decision, the reaction of decision making environment is determined as a set of values of current neuroagents' gains. The current gain of each neuroagent is transmitted to the inputs of the appropriate two-layer neural network. Then, the training of neuroagents is carried out by changing the weights of neuronal relationships by one of the learning algorithms without a teacher. The learning process is repeated until the weights of the neurons relationships are stabilized with a given accuracy. The course of training is aimed at maximizing the average gains of neuroagents. The decision of the game is achieved in one of the points of the collective optimum or equilibrium, depending on the values of the parameters of the chosen method of neuroagents training.

The developed software model confirms the convergence of the game neuroagent method (12) of decision making. The efficiency of the method is estimated by means of characteristic functions of average gains and errors of collective choice of the optimal decision making variant. The convergence of the neuroagent game method depends on the number of players, decisions, and relationships among the method parameters and decision making environment parameters.

The reliability of the obtained results has been confirmed by the repetition of the values of the calculated characteristics of the game neuroagent decision making method for different sequences of random variables.

The results of this work can be used to construct distributed control and decision making systems in conditions of uncertainty.

The conducted researches can be continued in the direction of applying of another configuration of neuroagents and other methods of their training, information exchange between agents of the stochastic game, growth of the number of players and the number of their pure strategies, definition of theoretical conditions of convergence of the game neuroagent method.

References

1. Uncertainty Index for Evaluating and Comparing Solutions for Stochastic Multiple Objective Problems, https://www.sciencedirect.com/science/article/abs/pii/S0377221720300047, last accessed 2020/04/12.
2. A multi-objective model for Pareto optimality in data envelopment analysis cross-efficiency evaluation, https://www.sciencedirect.com/science/article/abs/pii/S0377221719310446, last accessed 2020/04/12.
3. Generalized Pareto copulas: A key to multivariate extremes, https://www.sciencedirect.com/science/article/pii/S0047259X19300296, last accessed 2020/04/12.
4. A Heuristic Algorithm Combining Pareto Optimization and Niche Technology for Multi-Objective Unequal Area Facility Layout Problem, https://www.sciencedirect.com/science/article/abs/pii/S0952197619303458, last accessed 2020/04/12.
5. Xiushan, Jiang, Senping, Tian, Tianliang, Zhang, Weihai, Zhang: Pareto optimal strategy for linear stochastic systems with H∞ constraint in finite horizon. Information Sciences 512, 1103-1117 (2020).
6. An equilibrium in group decision and its association with the Nash equilibrium in game theory, https://www.sciencedirect.com/science/article/pii/S0360835219306072, last accessed 2020/04/12.
7. An extremum seeking-based approach for Nash equilibrium seeking in N-cluster noncooperative games, https://www.sciencedirect.com/science/article/pii/S0005109820300133, last accessed 2020/04/12.

8. Consensus of Fractional-Order Multiagent System via Sampled-Data Event-Triggered Control, https://www.sciencedirect.com/science/article/abs/pii/S0016003218301595, last accessed 2020/04/12.
9. Tauberian Theorems For General Iterations of Operators: Applications to Zero-Sum Stochastic Games, https://www.sciencedirect.com/science/article/abs/pii/S0899825618300204?via%3Dihub, last accessed 2020/04/12.
10. Evolution of cooperation in stochastic games, https://www.semanticscholar.org/paper/Evolution-of-cooperation-in-stochastic-games-Hilbe-%C5%A0imsa/ff49474432459bf295a7d2a775b1719c4fa09755, last accessed 2020/04/12.
11. Zhan, Bu, Guangliang, Gao, Hui-Jia, Li, Jie Cao: CAMAS: A cluster-aware multiagent system for attributed graph clustering. Information Fusion 37, 10–21 (2017).
12. Chaoxu, Mu, Qian, Zhao, Zhongke, Gao, Changyin, Sun: Q-learning solution for optimal consensus control of discrete-time multiagent systems using reinforcement learning. Journal of the Franklin Institute 356 (13), 6946–6967 (2019).
13. Commutative Stochastic Games https://pubsonline.informs.org/doi/10.1287/moor.2014.0676, last accessed 2020/04/12.
14. Semi-algebraic Tools for Stochastic Games 2015, https://www.semanticscholar.org/paper/Semi-algebraic-Tools-for-Stochastic-Games-Frederiksen/97d4bc8b3800fd14215f03aaddb6f4e3570255cd, last accessed 2020/04/12.
15. A Stochastic Game Framework for Analyzing Computational Investment Strategies in Distributed Computing with Application to Blockchain Mining Mathematics, https://www.semanticscholar.org/paper/A-Stochastic-Game-Framework-for-Analyzing-in-with-Dhamal-Chahed/0702f46e00dc3171ed0b38fdc827b21b120423ee, last accessed 2020/04/12.
16. CSI Neural Network: Using Side-channels to Recover Your Artificial Neural Network Information https://www.semanticscholar.org/paper/CSI-Neural-Network%3A-Using-Side-channels-to-Recover-Batina-Bhasin/905ad646e5745afe6a3b02617cd8452655232c0d, last accessed 2020/04/12.
17. Many Body Physics: Solving the Quantum Many Body Problem with Artificial Neural Networks, https://www.semanticscholar.org/paper/MANY%E2%80%90BODY-PHYSICS%3A-Solving-the-quantum-many%E2%80%90body-Carleo-Troyer/e4a85af3f5dc41e13dc2cae9ee851953709b764e, last accessed 2020/04/12.
18. Integration of New Evolutionary Approach with Artificial Neural Network for Solving Short Term Load Forecast Problem, https://www.semanticscholar.org/paper/Integration-of-new-evolutionary-approach-with-for-Singh-Dwivedi/0a5e2c346f61d9f68f323bc9946dee6988f72687, last accessed 2020/04/12.

19. State-of-the-art in Artificial Neural Network Applications: A survey 2018, https://www.semanticscholar.org/paper/State-of-the-art-in-artificial-neural-network-A-Abiodun-Jantan/efdb2aa8d8dadc182b139623911157fa158648ad, last accessed 2020/04/12.
20. An Artificial Neural Network as a Troubled-Cell Indicator, https://www.semanticscholar.org/paper/An-artificial-neural-network-as-a-troubled-cell-Ray-Hesthaven/2f8af28a213d614ec3b7f7d3f407095d838c944c, last accessed 2020/04/12.
21. Particle Filtering Methods for Stochastic Optimization with Application to Large-Scale Empirical Risk Minimization, https://www.sciencedirect.com/science/article/abs/pii/S0950705120300083, last accessed 2020/04/12.
22. DSCTool: A Web-Service-Based Framework for Statistical Comparison of Stochastic Optimization Algorithms,, https://www.sciencedirect.com/science/article/pii/S1568494619307586, last accessed 2020/04/12.
23. Identifying Practical Significance through Statistical Comparison of Meta-Heuristic Stochastic Optimization Algorithms, https://www.sciencedirect.com/science/article/pii/S156849461930643X, last accessed 2020/04/12.
24. Random Gradient Extrapolation for Distributed and Stochastic Optimization, https://www.semanticscholar.org/paper/Random-gradient-extrapolation-for-distributed-and-Lan-Zhou/263edefd27860664c6a596563f5e30dfd01c48e2, last accessed 2020/04/12.
25. Tomashevskyi, V., Yatsyshyn, A., Pasichnyk, V., Kunanets, N., Rzheuskyi A.: Data warhouses of hybrid type: features of construction. Advances in Intelligent Systems and Computing II (AISC) 938, 325–334 (2019).
26. Kazarian, A., Kunanets, N., Pasichnyk, V., Veretennikova, N., Rzheuskyi, A., Leheza, A., Kunanets, O.: Complex information e-science system architecture based on cloud computing model. CEUR Workshop Proceedings 2362, 366–377 (2019).
27. Methods of Statistical Research for Information Managers, https://ieeexplore.ieee.org/document/8526588/authors#authors, last accessed 2020/04/12.
28. Kaminskyi, R., Kunanets, N., Pasichnyk, V., Rzheuskyi, A., Khudyi, A.: Recovery gaps in experimental data. CEUR Workshop Proceedings 2136, 108–118 (2018).
29. Stochastic Approximation and Recursive Algorithms and Applications, https://www.springer.com/gp/book/9780387008943.
30. Adaptive Algorithms and Stochastic Approximations, https://www.springer.com/gp/book/9783642758966
31. Nazin A., Poznyak A.: Adaptive Choice of Variants: Recurrence Algorithms. Moscow: Science. (1986).