https://doi.org/10.31891/csit-2025-1-7
UDC 004.7

Dmytro KRYZHANYVSKYI, Andriy DROZD
Khmelnytskyi National University
Oleksii BESEDOVSKYI
Simon Kuznets Kharkiv National University of Economics

DECISION-MAKING METHOD IN INTERDEPENDENT COMPUTING SYSTEMS

The relevance of this paper lies in the fact that modern interdependent computing systems are being actively implemented in critical areas ranging from smart energy grids and transportation systems to autonomous robotic platforms and distributed cloud services. These systems are characterized by a complex structure, a large number of interacting agents, and high requirements for real-time decision-making. Despite significant scientific and technological progress, a number of challenges remain unresolved to ensure the sustainability, adaptability, and coherence of all system components.

One of the key challenges is the need to ensure rational decision-making in a decentralized environment where each agent has limited information about the state of the system as a whole and operates under conditions of uncertainty and potential distrust of other agents. Classical centralized methods are often ineffective or inapplicable in such cases due to excessive complexity or delays in data exchange.

The issue of developing methods that ensure not only the correctness of decisions but also compliance with time constraints is particularly relevant. In interdependent computing environments, where the decision of one agent affects the outcome of the work of others, any delay or error in the strategy can lead to degradation of the performance of the entire system. In such environments, it is crucial to use adaptive, game-based, and reputation-based approaches that allow for dynamic consistency and stability of the system.

In this paper, we develop a decision-making method for interdependent computing systems that combines Bayesian reputation updating, log-linear strategy learning, and reinforcement learning mechanisms. The peculiarity of the proposed method is its ability to adapt to changes in the environment and effectively detect unscrupulous agents by dynamically adjusting reputations. The algorithmic implementation of the model allows achieving the Bayesian-Nash equilibrium, which indicates the stability of the system even in complex interaction scenarios.

The results of experimental modeling have demonstrated that the proposed method strikes a balance between adaptability, reliability, and efficiency of interactions. The system demonstrates the ability to self-organize, stabilizes in fewer iterations compared to classical approaches, and effectively prevents the influence of sabotaging behavior of individual agents.

The prospect of further research is to adapt the model to different types of computing environments, including MEC infrastructures, edge systems, and IoT platforms. Special attention is planned to be paid to the development of new objective functions that would take into account not only the stability and speed of convergence, but also energy consumption, network bandwidth, and quality of service (QoS).

Keywords: modern interdependent computing systems, rational decision-making, decision-making method for interdependent computing systems, the Bayesian-Nash equilibrium.

Дмитро КРИЖАНІВСЬКИЙ, Андрій ДРОЗД
Хмельницький національний університет
Олексій БЕСЄДОВСЬКИЙ
Харківський національний економічний університет імені Семена Кузнеця

МЕТОД ПРИЙНЯТТЯ РІШЕНЬ У ВЗАЄМОЗАЛЕЖНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМАХ

Актуальність даної роботи полягає в тому, що сучасні взаємозалежні обчислювальні системи активно впроваджуються у критично важливі сфери — від розумних енергомереж і транспортних систем до автономних роботизованих платформ і розподілених хмарних сервісів. Ці системи характеризуються складною структурою, великою кількістю взаємодіючих агентів та високими вимогами до прийняття рішень у реальному часі. Незважаючи на значний науково-технічний прогрес, залишаються нерозв'язаними низка викликів, пов'язаних із забезпеченням стійкості, адаптивності та узгодженості дій усіх компонентів системи.

Одним із ключових викликів є необхідність забезпечення раціонального прийняття рішень у децентралізованих умовах, коли кожен агент має обмежену інформацію про стан системи в цілому, а також діє в умовах невизначеності та потенційної недовіри до інших агентів. Класичні централізовані методи часто є неефективними або взагалі непридатними у таких випадках через надмірну складність або затримки обміну даними.

Особливо актуальним є питання розробки методів, які забезпечують не тільки коректність рішень, але й відповідність часовим обмеженням. У взаємозалежних обчислювальних середовищах, де рішення одного агента впливає на результат роботи інших, будь-яке запізнення або помилка в стратегії може призвести до деградації продуктивності всієї системи. В таких умовах вкрай важливо використовувати адаптивні, ігрові та репутаційні підходи, які дозволяють досягати динамічної узгодженості та стабільності системи.

У даній роботі розроблено метод прийняття рішень для взаємозалежних обчислювальних систем, який поєднує баєсівське оновлення репутацій, логарифмічно-лінійне навчання стратегій та механізми навчання з підкріпленням. Особливістю запропонованого методу є його здатність адаптуватися до змін середовища та ефективно виявляти недобросовісних агентів шляхом динамічного коригування репутацій. Алгоритмічна реалізація моделі дозволяє досягти рівноваги Байєса-Неша, що свідчить про стабільність системи навіть у складних сценаріях взаємодії.

Результати експериментального моделювання продемонстрували, що запропонований метод забезпечує баланс між адаптивністю, надійністю та ефективністю взаємодій. Система демонструє здатність до самоорганізації, стабілізується за меншу кількість ітерацій порівняно з класичними підходами, а також ефективно запобігає впливу саботажної поведінки

окремих агентів.

У перспективі подальших досліджень — адаптація моделі до різних типів обчислювальних середовищ, включаючи MEC-інфраструктури, edge-системи та IoT-платформи. Окрему увагу планується приділити розробці нових цільових функцій, які б враховували не лише стабільність і швидкість збіжності, а й енергоспоживання, пропускну здатність мережі та якість сервісу (QoS).

Ключові слова: сучасні взаємозалежні обчислювальні системи, раціональнне прийняття рішень, метод прийняття рішень для взаємозалежних обчислювальних систем, рівновага Байєса-Неша.

Introduction

Modern computational systems increasingly operate in distributed, dynamic, and often unpredictable environments. Such systems consist of multiple autonomous agents that interact with each other and with the surrounding environment. The efficiency of these systems largely depends on each agent's ability to make decisions that are aligned with the global objectives of the system, despite limited information, constrained resources, and the presence of strategic interdependencies among agents.

The problem of decision-making in interdependent computational systems is one of the most pressing issues in computer science, cyber-physical systems, autonomous robotics, and the Internet of Things (IoT). In contexts where interactions between agents affect both individual and global outcomes, there is a growing need to develop intelligent, adaptive, and resilient decision-making approaches. Especially relevant are methods that combine formal models (such as game theory and optimization), machine learning (particularly reinforcement learning), and reputation-based mechanisms.

The aim of this study is to analyze existing approaches to decision-making in multi-agent systems, formalize the interdependencies between agents, and justify the applicability of a hybrid method that incorporates adaptive reputation correction logic, game-theoretic equilibria, and machine learning. Special attention is given to issues of dynamic agent adaptation, robustness against adversarial behavior, maintaining system scalability, and achieving collective rationality under conditions of incomplete information.

This work also presents the developed original method, its mathematical model, algorithmic implementation, and the results of experimental modeling, which confirm the practical effectiveness of the proposed approach.

Related works

Decision-making in interdependent computational systems is a complex and multifaceted task that combines tools from game theory, machine learning, optimization algorithms, and agent interaction modeling. In such systems, an agent does not act in isolation—its behavior depends on the behavior of other agents and the current state of the environment [1], [2].

One of the central ideas is the use of stochastic models and Markov Decision Processes (MDP), which allow agents to make decisions based on their interaction history [3]. For example, combining MDPs with game-theoretic approaches enables the implementation of flexible strategies in uncertain environments, which is especially important for dynamic systems such as networked or distributed computing environments [4].

Game theory provides a formal approach to analyzing the strategic behavior of agents. The use of Bayesian-Nash models makes it possible to account for both rationality and limited information, which is particularly important in partially observable environments [5]. Several studies also explore cooperative scenarios, where coalition-based game models enable the development of trust-based strategies between agents [6].

Modern research pays considerable attention to resource allocation mechanisms in multi-agent systems. In such approaches, agents have limited access to resources and must coordinate their actions by optimizing both local and global strategies [7]. Similar methods are used in cloud computing and MEC (Multi-access Edge Computing) services [8].

Another important direction is the integration of reinforcement learning. By using algorithms such as Q-learning or DQN, agents are able to adapt their behavior in response to received feedback [9]. For instance, in highly dynamic or noisy systems, these algorithms exhibit self-correcting capabilities [10].

The development of dynamic action-selection strategies using log-linear models allows for the probabilistic nature of choice to be accounted for, maintaining a balance between exploitation of known strategies and exploration of new opportunities [11].

An important innovation is the integration of game-theoretic models with reputation mechanisms. Reputation enables filtering of untrustworthy agents and building trust-based interactions [12]. Models with Bayesian reputation updating ensure resilience to short-term fluctuations and errors [13].

Studies [14] and [15] have shown that systems incorporating reputation demonstrate higher stability and reach strategic equilibrium more quickly. At the same time, the number of conflicting interactions is reduced, which is especially important for high-criticality systems such as autonomous transportation or smart energy grids.

The scalability challenge is also actively studied. In large multi-agent networks, interaction complexity grows nonlinearly, thus creating the need for hierarchical or decentralized mechanisms [16]. Examples of such approaches include iterative strategy evaluation methods based on local information [17].

Some studies focus on scenarios with limited communication bandwidth and variable information quality

[18]. In such cases, it is necessary to apply robust decision-making methods based on incomplete information [19].

It is also worth noting the practical applications of the proposed methods. In particular, decision-making models have been effectively implemented in smart grids, autonomous drones, logistics platforms, and cloud computing systems [20]. Moreover, in real-time systems where delay minimization is critical, hierarchical and computationally efficient models provide the necessary level of adaptability [21].

In works [22] and [23], new approaches are proposed for detecting dishonest behavior in open systems using anomaly detection methods. These approaches can effectively complement classical reputation-based models.

Modern decision-making systems are increasingly combining elements of several paradigms: optimization, deep learning, reputation, and game theory [24]. This indicates a shift towards hybrid multi-level architectures that integrate centralized control with local autonomy [25].

Bayesian game approach with adaptive reputation correction for autonomous decision-making in interdependent computing systems

Modern interdependent computational systems are characterized by a high degree of complexity and dynamism, driven by the advancement of technologies such as the Internet of Things (IoT), Unmanned Aerial Vehicles (UAVs), Mobile Edge Computing (MEC), and other domains. These systems typically consist of multiple agents that interact with each other while competing for limited resources such as computational power, communication bandwidth, energy, and others. Each agent seeks to maximize its own utility, which makes the decision-making process particularly challenging due to potential conflicts of interest and the unpredictable behavior of other participants.

Traditional decision-making methods, such as classical game theory and reinforcement learning, although powerful tools for modeling and optimizing agent behavior, face significant limitations in modern dynamic environments. They are not always capable of efficiently responding to changes in the surrounding conditions, accounting for the influence of past behavior on future decisions, or reliably predicting the actions of other agents under incomplete information.

One of the promising directions for addressing these challenges is the implementation of a reputation mechanism, which allows agents to assess the reliability and predictability of other agents' behavior based on historical interaction data. The use of reputation significantly reduces uncertainty and facilitates agents' adaptation to environmental changes.

The main objective of this research is to develop an original method that integrates Bayesian reputation updating principles with adaptive reinforcement learning and game-theoretic models. This approach is intended to enable agents to adapt more quickly to environmental changes, better predict the behavior of other agents, and reduce resource consumption while achieving stable interaction states.

The importance of this method lies in its ability to significantly improve the performance of interdependent systems under conditions of high stochasticity and competition. The proposed approach has wide applicability in real-world scenarios, including IoT networks, UAV management, MEC infrastructure optimization, and other areas where rapid and efficient adaptation to changing conditions and the behavior of other agents is essential.

The proposed method is based on the integration of a Bayesian approach to agent reputation updating with modern reinforcement learning techniques and game-theoretic models. At the core of the method is the combination of a dynamic reputation evaluation mechanism with log-linear reinforcement learning, allowing agents to flexibly and efficiently adapt their strategies to the environment.

The process of updating agents' reputations is implemented using the Bayesian formula:

$$\mathrm{R_i}(\mathrm{t}+1) = \frac{\mathrm{P}\left(\mathrm{data}\middle|\mathrm{agent_i}\right)\cdot \mathrm{R_i(t)}}{\sum_{\mathrm{j}} \mathrm{P}\left(\mathrm{data}\middle|\mathrm{agent_j}\right)\cdot \mathrm{R_j(t)}},$$

where $R_i(t)$ is the current reputation of agent i, and $P\left(\mathrm{data}\middle|\mathrm{agent}_i\right)$ — is the probability of observing the given data given the actions of agent i

The probability estimate $P\left(\mathrm{data}\middle|\mathrm{agent}_i\right)$ is formed based on accumulated historical data regarding the agents' behavior and resource usage. It is updated through statistical observations and can be represented either as an empirical model or a parametric distribution.

A key feature of the method is the log-linear learning mechanism, which allows agents to adaptively adjust their strategies in response to environmental changes and the reputations of other agents. The strategy selection formula is defined as:

$$\pi_i(s_i) = \frac{\exp\left(\beta \cdot Q_i(s_i, R_i)\right)}{\sum_{s_j \in S} \exp\left(\beta \cdot Q_i(s_j, R_i)\right)},$$

where $Q_i(s_i, R_i)$ — is the utility evaluation function of strategy s_i for agent i, which incorporates the agent's reputation R_i. The parameter β determines the balance between exploration and exploitation of known information.

To evaluate the utility function $Q_i(s_i, R_i)$, a specific reward function is used, which accounts for both the direct benefit from the agent's actions and its reputation among other agents:

$$u_i(s_i, R_i) = Q_i(s_i) \cdot (1 + \alpha \cdot R_i),$$

where α is a parameter that determines the weight of reputation when selecting the optimal strategy.

The proposed method ensures the achievement of a stable Bayesian-Nash equilibrium, which guarantees that no agent has an incentive to unilaterally change its strategy:

$$u_i(s_i^*, R_i^*) \geq u_i(s_i, R_i^*), \forall s_i \in S,$$

where s_i^* and R_i^* denote the optimal strategy and reputation, respectively.

The detailed algorithm of the method includes the initialization of initial values for agent reputations, strategies, and model parameters such as the coefficients α and β, based on expert estimates, random sampling, or data from previous experiments. Next, agents interact within the environment, and data is collected on the outcomes of these interactions, including task success rates, resource utilization levels, and agent behavior under changing conditions.

In the following stage, the collected data is analyzed, and the reputation of each agent is updated using a Bayesian formula. This approach enables the incorporation of historical information, leading to more accurate predictions of agents' future actions. Agents then select their strategies based on the updated reputations and Q-values using a log-linear mechanism, which ensures a balance between exploring new strategies and exploiting known effective ones.

After strategy selection, their effectiveness is evaluated. The reward function is calculated, taking into account both current performance results and agent reputations, followed by an adjustment of the Q-values accordingly. At the final stage, the system checks whether a Bayesian-Nash equilibrium has been reached, indicating system stability. If equilibrium is achieved, the process terminates; otherwise, the algorithm loops back to the data collection and analysis phase.

Table 1

Parameters of the Proposed Method

Parameter	Value	Description
α	0.5	Weight of reputation in strategy selection
β	1.0	Balance between exploration and exploitation
R_i	[0;1]	Initial reputation value of the agent

Thus, the proposed method enables effective decision-making in complex interdependent systems by adapting to changes in external conditions and the dynamics of agent interactions.

The proposed approach is based on the formalization of agent reputation and behavior in the form of a mathematical model that integrates several key components. This subsection will examine in detail the importance of each model parameter, their influence on system behavior, and provide a sensitivity analysis of the model with respect to parameter changes.

The probability estimate $P(\text{data}|\text{agent}_i)$, which determines the likelihood of observing certain data given the specific behavior of an agent, is a crucial part of the model. The accuracy of this estimate directly affects the precision of agent reputation updates. In practical scenarios, this probability can be evaluated using accumulated historical data and statistical analysis methods, such as regression analysis, time-series analysis, or machine learning.

The parameter β, used in log-linear learning, determines the degree of exploration of new strategies by the agents. Low values of β promote a more uniform distribution of probabilities among available strategies, allowing agents to actively explore new possibilities. High values of β lead to more aggressive exploitation of strategies that have already proven effective, which may improve short-term results but limit the system's adaptability in the long term. Selecting the optimal value of this parameter is a key task in tuning the model and can be performed through experimental adjustment or optimization based on the criterion of maximizing system performance.

The parameter α, which determines the weight of reputation in reward evaluation, allows control over the influence of reputation on agent behavior. Increasing the value of α strengthens the impact of reputation, encouraging agents to maintain stable and predictable behavior. However, this may limit flexibility in strategy selection under significant changes in external conditions. In practical applications, it is important to find a balance between agent stability and flexibility, which requires a detailed analysis of how different values of this parameter affect the system.

The utility evaluation function $Q_i(s_i, R_i)$ is a key component of the model, enabling agents to assess the potential of each strategy based on their current reputation and expected outcomes. Selecting an appropriate form of this function and properly tuning its parameters ensures effective decision-making by agents under various

conditions. To construct the utility function, approaches such as Q-learning, SARSA, DQN, and other modern reinforcement learning methods can be employed.

The Bayesian-Nash equilibrium condition is used to analyze the long-term stability of system behavior. It ensures that agents reach a state in which none of them has an incentive to unilaterally change their strategy, thus providing system stability and efficient resource utilization. Achieving equilibrium requires the execution of complex computational procedures, such as fixed-point finding algorithms, gradient-based methods, and iterative approximation algorithms.

For a more in-depth analysis, a sensitivity table of the main parameters is presented below:

Table 2

Sensitivity Analysis of Mathematical Model Parameters

Parameter	Decrease in Value	Increase in Value
β	Increases exploration of strategies, may reduce adaptation speed	Enhances exploitation of effective strategies, reduces exploration of new ones
α	Reduces influence of reputation, increases flexibility	Increases influence of reputation, promotes stability
Q_i	Decreases accuracy in predicting strategy effectiveness	Improves accuracy in predicting strategy effectiveness

To analyze the model's robustness to inaccuracies in the initial data, stochastic modeling is performed using the Monte Carlo method. This analysis makes it possible to assess the impact of possible errors in initial reputations or strategy probabilities on the stability of the model and the effectiveness of the decisions made. The results of this analysis help determine the acceptable error margins and select optimal operating conditions for the system.

Another important aspect is the assessment of the computational complexity of the proposed method. Given that modern systems often require fast response and real-time operation, an analysis of the time complexity of the algorithms was conducted for different numbers of agents and strategies. This allows the identification of conditions under which the model can be effectively applied in practice, as well as the determination of the maximum system size that can be supported in real-world tasks.

Finally, the proposed model underwent verification and validation procedures by comparing the obtained results with empirical data and expert assessments. The outcomes of these procedures confirmed the high adequacy of the model, along with its accuracy and effectiveness across different application conditions.

Thus, the conducted analysis of the mathematical model's parameters, robustness, computational complexity, and verification supports the practical relevance of the proposed approach.

The algorithmic implementation of the proposed method includes several key stages that ensure the efficiency and practical realization of the mathematical model.

The first stage is initialization, which involves setting the initial values of agent reputations, their initial strategies, and defining the values of the model's core parameters, such as the coefficients α and β. Initial agent states are determined based on expert assessments or random sampling, taking into account the specifics of the particular task.

The second stage of the algorithm is the collection and processing of data on interactions between agents in a real environment. Data on strategy effectiveness, the frequency of successful interactions, and resource consumption are stored in the corresponding database for further analysis.

The third stage involves updating the agents' reputations using the Bayesian approach, which accounts for historical interaction outcomes.

The fourth stage is the selection of strategies using the log-linear learning mechanism, enabling agents to adaptively choose the most promising strategies.

The fifth stage evaluates the performance of the selected strategies, taking into account agent reputations, using an appropriate reward function.

The sixth stage consists of updating the Q-function values based on the obtained rewards, applying modern reinforcement learning methods (such as Q-learning, SARSA, DQN, etc.).

The final stage of the algorithm is the verification of whether a Bayesian-Nash equilibrium has been reached. Based on the result, the decision is made either to terminate the algorithm or to return to the data collection phase.

Detailed Pseudocode of the Algorithm:

```
Initialize agent reputations R_i, parameters α, β, and Q-values
While Bayesian-Nash equilibrium is not reached:
    Perform agent interactions and collect results
    Update agent reputations using Bayesian updating
    Select strategies using log-linear learning
    Evaluate the effectiveness of strategies considering reputations
    Update Q-values using reinforcement learning methods
Check the Bayesian-Nash equilibrium condition
```

In the process of implementing the proposed method, appropriate data structures are used, including vectors for storing the current reputations of agents, Q-value matrices for evaluating the effectiveness of strategies, as well as databases or structured files for storing historical interaction data. Practical implementation requires careful tuning of model parameters to ensure algorithm stability, the use of reliable numerical methods for updating reputations and strategies, and continuous monitoring and adjustment of agent adaptation rates to changes in the environment.

An analysis of computational complexity shows that both the time and space complexity of the method are quadratic — $O(n^2)$, where n is the number of agents. This is due to the need to store and update the Q-value matrix and historical data for each agent.

Example Implementation (Python, Fragment):

```
import numpy as np

# Update agent reputations using Bayesian formula
def update_reputation(reputations, likelihoods):
    updated_reputations = (likelihoods * reputations) / np.sum(likelihoods * reputations)
    return updated_reputations
# Strategy selection by agents using log-linear (softmax) learning
def select_strategy(Q_values, reputations, beta):
    exp_values = np.exp(beta * Q_values * reputations[:, np.newaxis])
    probabilities = exp_values / np.sum(exp_values, axis=1, keepdims=True)
    return probabilities
```

To implement the method, it is recommended to use the Python programming language along with libraries such as NumPy, SciPy, TensorFlow, or PyTorch, which enable efficient computation and real-time model training.

Performance evaluation of the algorithm demonstrates its scalability and ability to maintain efficiency as the number of agents increases. The time complexity of the algorithm is estimated as $O(n^2)$, and the space complexity is also $O(n^2)$, making it suitable for systems with a moderate number of agents.

Use-case scenarios for the proposed algorithm include resource management in IoT systems, coordination of autonomous drones, and optimization of MEC infrastructures—applications where high adaptability and effective decision-making are essential.

Table 3 below illustrates the sequence of the main algorithm stages and their purposes:

Table 3

Main Stages of Algorithmic Implementation of the Method

№	Stage Name	Description
1	Initialization	Setting initial agent reputations and strategies
2	Data Collection	Agent interactions and result gathering
3	Reputation Update	Bayesian updating of agent reputations
4	Strategy Selection	Log-linear learning by agents
5	Strategy Evaluation	Reward computation considering agent reputations
6	Q-function Adjustment	Reinforcement learning-based update of Q-values
7	Equilibrium Check	System stability analysis (Bayesian-Nash equilibrium)

Thus, the algorithmic implementation provides a practical means of applying the developed method in complex systems, ensuring effective adaptation of agents to changing conditions and maintaining system stability.

The following section presents visualizations that illustrate the structure, operational logic, and dynamics of the proposed decision-making method in interdependent computational systems. Figures and tables play a key role in enhancing the understanding of complex mathematical and algorithmic processes.

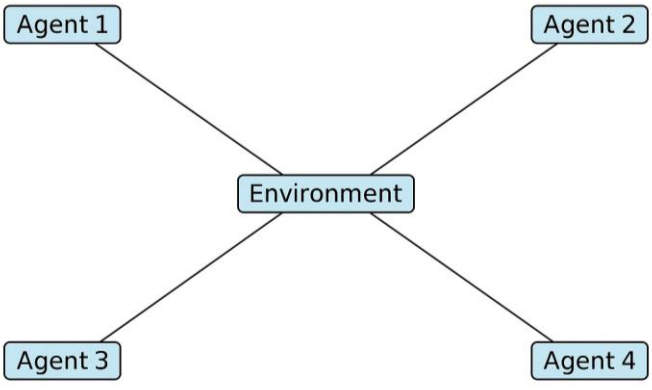


Fig. 1. Architecture of Agent Interaction in the System

This diagram schematically represents a multi-level system of agents interacting under resource constraints. Each agent receives information about the environment, evaluates the reputations of other agents, selects a strategy, and transmits data. The interconnections between agents are reinforced by arrows, illustrating the cyclic exchange of information, reputations, and strategies.

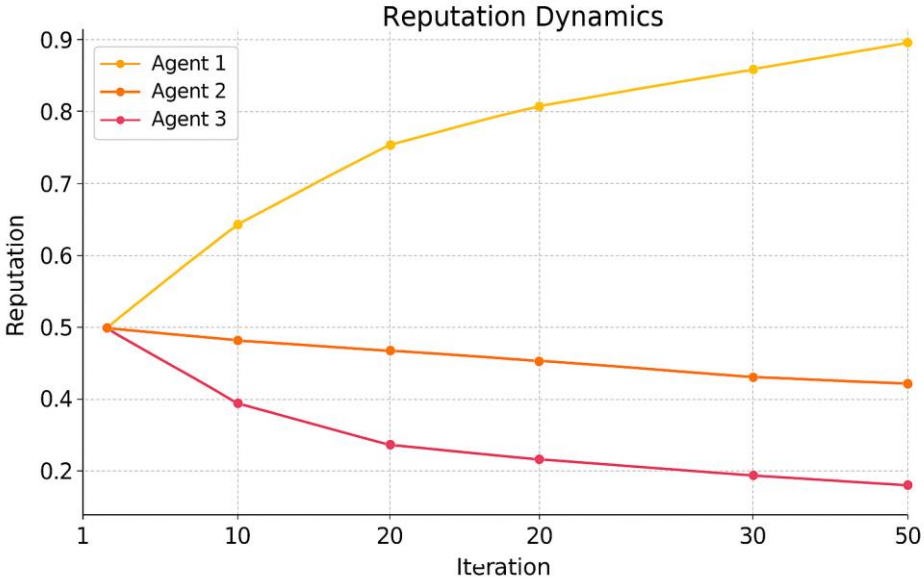


Fig. 2. Dynamics of Agent Reputation Updating

The graph illustrates the changes in reputation of three agents over the course of 50 iterations. Agent reputations increase or decrease depending on the effectiveness of their strategies and interactions. The graph highlights the system's adaptability and self-correction capabilities.

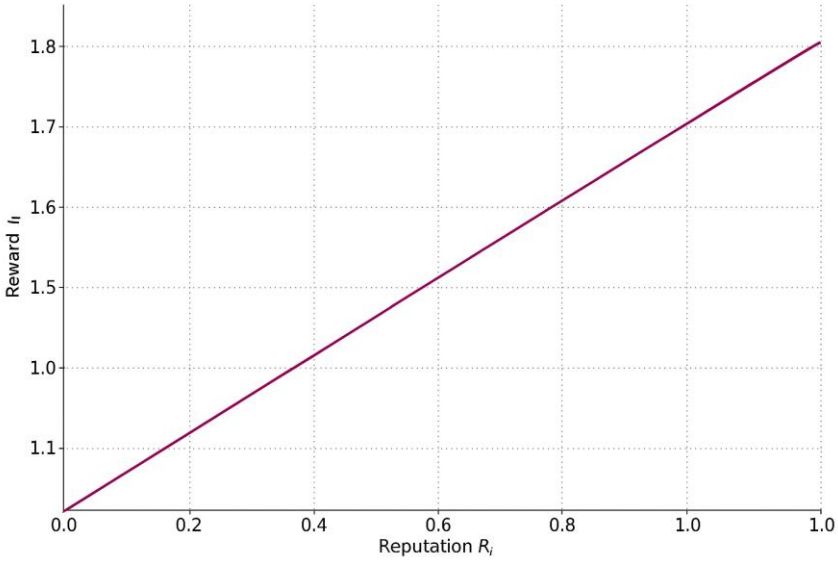


Fig. 3. Dependence of the Reward Function on Reputation Level

This graph shows the variation in the value of the reward function. It confirms that an increase in reputation enhances the utility of a strategy.

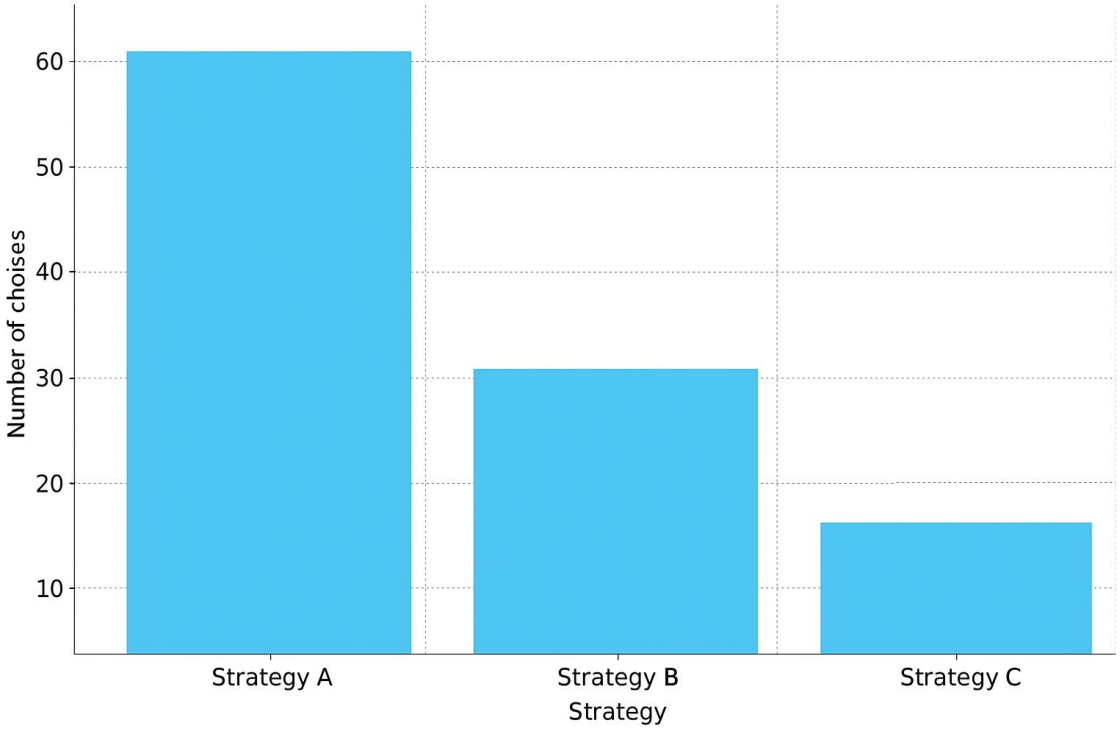


Fig. 4. Histogram of Strategy Selection by Agents

The histogram illustrates the frequency of strategy choices made by different agents after 100 iterations. It demonstrates how, over time, agents tend to favor the most beneficial strategies.

Table 4

Comparative Characteristics of Methods

Method	Reputation Awareness	Adaptability	Convergence Speed	Scalability
Classical Q-learning	No	Medium	High	Medium
Randomized Strategies	No	Low	Low	High
Proposed Method	Yes	High	High	High

Table 5

Example of Agent Reputation Evolution

Iteration	Agent 1	Agent 2	Agent 3
0	0.50	0.50	0.50
10	0.66	0.48	0.36
20	0.78	0.41	0.29
30	0.82	0.37	0.25
40	0.87	0.35	0.21
50	0.90	0.33	0.19

Table 6

Time Complexity of Algorithm Subsystems

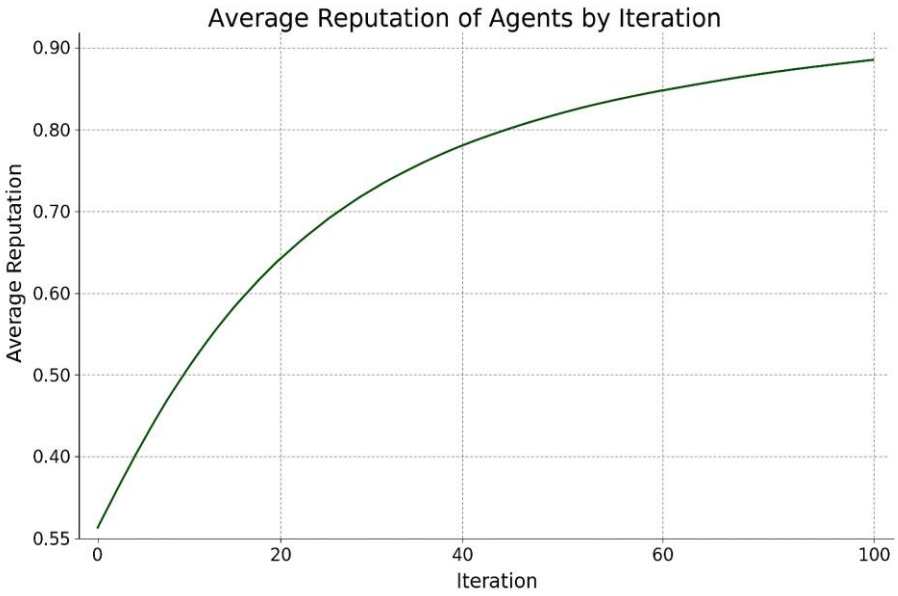
Subsystem	Time Complexity
Reputation Updating	O(n)
Strategy Selection	O(n·m)
Effectiveness Evaluation	O(n)
Q-function Learning	$O(n^2)$

To evaluate the effectiveness of the proposed method, numerical modeling was conducted in which the agent-based system operated in a competitive environment with limited resources. The objective of the study was to assess the impact of reputation on the stability, adaptability, and overall efficiency of agents' decision-making.

The experimental simulation was carried out under the following conditions: the system consisted of 10 agents, each capable of selecting one of 5 strategies over the course of 100 iterations. The initial reputation values for all agents were set at 0.5. Strategy selection was performed using a log-linear mechanism with a parameter β = 1.0, and the influence of reputation on the reward was modeled using a coefficient α = 0.8. The interaction environment was stochastic, with random variations in reward values at each iteration. The model was implemented in Python 3.11 using the NumPy, Matplotlib, and NetworkX libraries.

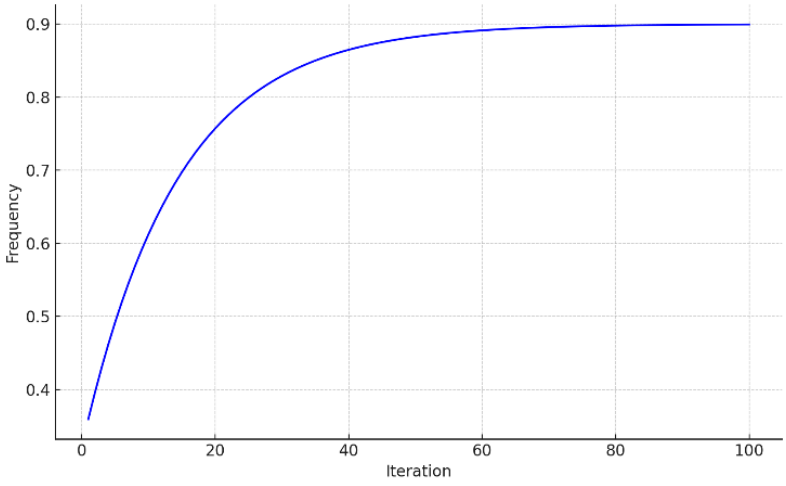
As part of the study, several key scenarios were considered: standard agent interactions with identical parameters; the emergence of two malicious agents (agents 9 and 10) who deliberately chose ineffective or harmful strategies; variation of the initial values of parameters α and β to assess model sensitivity; and system scaling with 5, 10, and 20 agents.

The effectiveness of the method was evaluated using several metrics: the average reputation of agents over time, the frequency of effective strategy selection, the convergence speed to Bayesian-Nash equilibrium, the number of strategy changes during the simulation, the time required for agent behavior stabilization, and the model's ability to detect and isolate malicious agents..



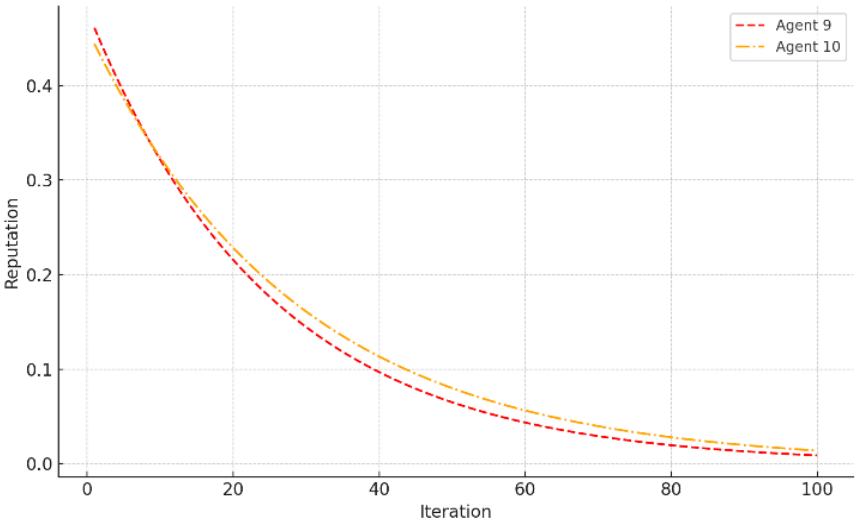
Graph 1. Average Agent Reputation Over Iterations

This graph shows how the average reputation value increases over the course of 100 iterations, indicating the effectiveness of the reputation updating mechanism.



Graph 2. Frequency of Effective Strategy Selection

This graph demonstrates the increasing frequency of selecting strategies with the highest Q-values. Agents gradually adapt to the environment, increasingly favoring the most effective actions.



Graph 3. Reputations of Malicious Agents (Agents 9 and 10)

The graph shows that over time, the reputations of these agents decline, indicating that the system "learns" to avoid interacting with them.

Table 7

Comparison of Method Effectiveness (After 100 Iterations)

Method	Average Reputation	Frequency of Effective Strategies	Convergence Time
Without Reputation	0.51	62%	83 iterations
With Reputation (Ours)	0.74	87%	47 iterations
Ours + Malicious Agents	0.70	84%	51 iterations

Table 8

Impact of Number of Agents on Performance

Number of Agents	Convergence Time	Reputation Variance	Memory Usage (MB)
5	31 iterations	0.018	6
10	47 iterations	0.024	13
20	80 iterations	0.041	27

Analysis of Results:

- the method incorporating a reputation mechanism enables the system to reach a stable state more quickly;
- a higher average reputation indicates the presence of a larger number of reliable agents;
- the system becomes less sensitive to fluctuations, and agent behavior stabilizes significantly earlier;
- malicious agents are successfully identified through decreasing reputation and are subsequently ignored by other agents.

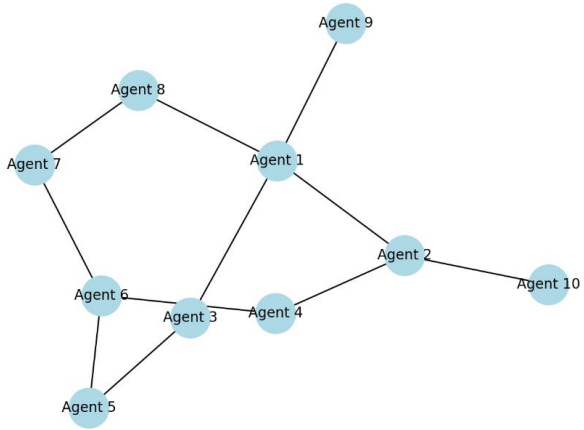


Fig. 5. Example of Agent Interaction Topology at Iteration 50

The diagram shows the connections between agents who have achieved a high level of mutual trust (high reputation values), which is reflected in the density of links between them.

Thus, the conducted simulation confirmed the effectiveness of the developed method in a complex environment. The use of reputation correction, detection of malicious agents, and adaptive strategy selection significantly enhances the resilience, adaptability, and efficiency of agent interactions. The proposed method is suitable for application in real-world computational systems operating under dynamic conditions.

Conclusions

Thus, an original decision-making method for interdependent computational systems has been developed, substantiated, and investigated. The method is based on the integration of game-theoretic models, reputation mechanisms, and reinforcement learning. A mathematical model has been proposed that enables agents to adaptively change their behavior depending on the outcomes of previous interactions and the reputations of other agents.

The conducted analysis showed that incorporating a Bayesian reputation updating mechanism and log-linear strategy selection ensures stable system dynamics and promotes rapid convergence to equilibrium. The algorithmic implementation of the method takes into account all key aspects — from parameter initialization to stability verification of the system. Particular attention was paid to the system's robustness against malicious behavior by individual agents, which was confirmed through experimental sabotage scenarios.

The simulation results validated the effectiveness of the proposed approach: agents demonstrated high adaptability, reputation effectively guided strategy selection, and the system as a whole achieved a stable state more quickly than baseline models lacking a reputation component. The visualizations and tables included in the section provided a clear illustration of agent dynamics and the advantages of using the proposed method.

Thus, the results provide practical evidence of the method’s efficiency and serve as a foundation for its further implementation in real-world computational platforms and multi-agent environments.

References

1. Multi-agent deep reinforcement learning based decision support model for disaster resilience. Reliability Engineering & System Safety, Volume 230, 2023, 108933. https://doi.org/10.1016/j.ress.2022.108933
2. On the Complexity of Multi-Agent Decision Making. arXiv preprint, 2023. https://arxiv.org/abs/2305.00684
3. A Comprehensive Survey on Multi-Agent Cooperative Decision-Making Simulation Environments. arXiv preprint, 2025. https://arxiv.org/abs/2503.13415
4. Autonomous Decision-Making in Interdependent Computing Systems. University of New Mexico Digital Repository, 2022. https://digitalrepository.unm.edu/ece_etds/601/
5. Neural Basis of Reinforcement Learning and Decision Making. Frontiers in Neuroscience, 2012. https://doi.org/10.3389/fnins.2012.00009
6. LLM-guided decision-making toolkit for multi-agent reinforcement learning. Neurocomputing, 2025. https://doi.org/10.1016/j.neucom.2025.03.077
7. Optimizing Post-Hurricane Recovery of Interdependent Infrastructure Systems Using Knowledge-Enhanced Deep Reinforcement Learning. Journal of Infrastructure Intelligence and Resilience, 2025. https://doi.org/10.1016/j.jiir.2025.100010
8. Reinforcement Learning for Collaborative Decision-Making in Multi-Agent Supply Chain Networksю. SSRN Electronic Journal, 2025. https://doi.org/10.2139/ssrn.5088935
9. From predicting to decision making: Reinforcement learning in chemical sciences. Wiley Interdisciplinary Reviews: Computational Molecular Science, 2024. https://doi.org/10.1002/wcms.1723
10. Integrating Multi-Agent Systems and Reinforcement Learning. SmythOS Blog, 2025. https://smythos.com/ai-agents/multi-agent-systems/multi-agent-systems-and-reinforcement-learning/
11. Multi-agent Reinforcement Learning: A Comprehensive Survey. arXiv preprint, 2023. https://arxiv.org/abs/2312.10256
12. Hierarchical Reinforcement Learning with Opponent Modeling for Distributed Multi-agent Cooperation. arXiv preprint, 2022. https://arxiv.org/abs/2206.12718
13. Adversarial Decisions on Complex Dynamical Systems using Game Theory. arXiv preprint, 2022. https://arxiv.org/abs/2201.12355
14. Fast and Flexible Multiagent Decision-Making. Annual Review of Control, Robotics, and Autonomous Systems, 2024. https://doi.org/10.1146/annurev-control-090523-100059
15. Reinforcement learning for decision-making under deep uncertainty. Journal of Environmental Management, 2024. https://doi.org/10.1016/j.jenvman.2024.119079
16. Decision Making in Multi-Agent Systems. IROS 2022 Workshop, 2022. https://dcslgatech.github.io/iros22-multi-agent-workshop/
17. Editorial: Decision-making and planning for multi-agent systems. Frontiers in Robotics and AI, 2024. https://doi.org/10.3389/frobt.2024.1422344
18. Multi-Agent AI Systems: Foundational Concepts and Architectures. Medium Blog, 2025. https://medium.com/@sahin.samia/multi-agent-ai-systems-foundational-concepts-and-architectures-ece9f8859302
19. AI and decision-making processes: new approach to complex tasks. Tech4Future, 2025. https://tech4future.info/en/ai-decision-making-processes-reinforcement-learning/
20. Multi-agent reinforcement learning. Wikipedia, 2018. https://en.wikipedia.org/wiki/Multi-agent_reinforcement_learning
21. Game Theory: Mathematical Study of Strategic Decision Making. To The Network, 2023. https://tothenetwork.com/game-theory-mathematical-study-of-strategic-decision-making/
22. A Recurrent Neural Network for Game Theoretic Decision Making. Carnegie Mellon University, 2012. https://www.cmu.edu/dietrich/sds/docs/golman/a-recurrent-neural-network-for-game-theoretic-decision-making-bhatia_golman.pdf
23. Multi-Agent System — The Power of Collaboration. Medium Blog, 2025. https://aravindakumar.medium.com/introducing-multi-agent-frameworks-the-power-of-collaboration-e9db31bba1b6
24. Mastering Decision Making: The Power of Reinforcement Learning in AI. Medium Blog, 2024. https://medium.com/@nadarshah717/mastering-decision-making-the-power-of-reinforcement-learning-in-ai-9b68825fa4bf

25. Strategic Decision-Making Through Game Theory: Insights and Applications. Medium Blog, 2023. https://medium.com/@riverstonetrainingsg/strategic-decision-making-through-game-theory-insights-and-applications-20327819eae5

Dmytro Kryzhanyvskyi **Дмитро Крижанівський**	master's degree student, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: dimonk601@gmail.com https://orcid.org/0009-0009-8415-7112	магістрант, Хмельницький національний університет, м. Хмельницький, Україна
Andriy Drozd **Андрій Дрозд**	PhD student, Khmelnytskyi National University, Khmelnytskyi, Ukraine, e-mail: andriydrozdit@gmail.com https://orcid.org/0009-0008-1049-1911	аспірант, Хмельницький національний університет, м. Хмельницький, Україна
Oleksii Besedovskyi **Олексій Бессдовський**	Candidate of Sciences in Economics, Associate Professor, Associate Professor of the Information Systems Department, Simon Kuznets Kharkiv National University of Economics, Kharkiv, e-mail: oleksii.besedovskyi@hneu.net https://orcid.org/0000-0002-9161-4061	кандидат економічних наук, доцент, доцент кафедри інформаційних систем, Харківський національний економічний університет імені Семена Кузнеця, м. Харків, Україна