Integrating multi-criteria decision making and reinforcement learning for consensus protocol selection

Nurlan Tashatov, Ruslan Ospanov, Yerzhan Seitkulov, Dina Satybaldina, Banu Yergaliyeva
Department of Information Security, Faculty of Information Technologies, Research Institute of Information Security and Cryptology, L.N. Gumilyov Eurasian National University, Astana, Kazakhstan

Article Info

Article history:

Received Nov 14, 2024
Revised Mar 12, 2025
Accepted Jul 5, 2025

Keywords:

Consensus protocol
Control systems
Internet of things
Machine learning
Mathematical modeling
Multi-criteria decision making
Reinforcement learning

ABSTRACT

The rapid progress in artificial intelligence technologies in recent years has been largely driven by advances in reinforcement learning (RL). RL methods have proven to be highly effective in solving many practical problems. Distributed ledger technologies are finding wide application in the internet of things (IoTs), providing new approaches to solving problems of traditional IoT systems. Consensus is a fundamental component of distributed ledger technologies, responsible for ensuring data consistency between nodes, its security and accuracy. This paper is devoted to the study of the optimal choice of blockchain consensus protocol for IoT networks based on a combination of multi-criteria decision making (MCDM) and RL methods. The paper discusses the potential of merging MCDM and RL methods for selecting blockchain consensus protocols in IoT networks. It suggests a combined framework for effective protocol selection and management.

Corresponding Author:

Banu Yergaliyeva
Department of Information Security, Faculty of Information Technologies
Research Institute of Information Security and Cryptology, L.N. Gumilyov Eurasian National University
Astana, Kazakhstan
Email: banu.yergaliyeva@gmail.com

1. INTRODUCTION

Reinforcement learning (RL) methods have demonstrated strong performance across various domains, including control systems and optimization [1]. In these fields, RL has enabled autonomous agents to learn complex behaviors and make real-time decisions, making it a key tool for enhancing robotic capabilities and control mechanisms [2]. Also RL is actively used in the internet of things (IoT) to solve problems related to automation, optimization and control of complex systems. The integration of RL with IoT in robotics and control systems opens up new opportunities for autonomous decision-making, efficient data-driven optimization, and real-time adaptation. IoT connects robots and control systems to a vast network of sensors and devices, enabling the continuous flow of data, which RL can leverage to improve the performance of robotic systems and optimize their behaviors in dynamic environments. The IoT is one of the key application areas of distributed ledger and blockchain technologies. These technologies can offer effective solutions to address the shortcomings of traditional IoT applications [3]-[8]. One of the important components of the distributed ledger reference architecture is the consensus layer, which manages the consensus among network nodes, ensuring the consistency of the ledger state, as well as ensuring the security, accuracy, and protection of data. The choice of consensus protocol has a significant impact on the performance and security of the blockchain system. Today, there are many different consensus protocols with unique characteristics that can be used in IoT blockchain networks [9]-[14]. Therefore, choosing the most suitable consensus protocol for a particular distributed ledger system, especially for IoT blockchain, is a complex task due to conflicting requirements and high uncertainty. To solve such problems, multi-criteria decision making (MCDM) methods are often used to evaluate various alternatives. MCDM techniques can be used in various scenarios, such as robot path planning, task scheduling, and resource allocation [15]. Several works focus on applying these methods within the field of distributed ledger technology (DLT) and blockchain, primarily to assist in selecting a blockchain platform, as well as in choosing a consensus protocol [16]. To select the optimal consensus protocol in the IoT context, it is necessary to consider many criteria, including performance, security, energy consumption, scalability, and resistance to various types of attacks. A multi-criteria approach to decision making allows us to take into account all these aspects and conduct a comprehensive analysis of possible alternatives.

In a series of works [17], [18] a model of a cryptographic system for backup storage of confidential data in secure servers was presented. The model is based on a cryptographic protocol, as a result of which messages are encrypted in such a way that their decryption will be possible only no earlier than a specified exact term. The functionality of the model works through the interaction of a portal for receiving user requests for data encryption, a network of distributed servers for performing all necessary calculations and secure storage of the corresponding secret data and, directly, the user of this system. The user sends, and the portal, accordingly, accepts a request for secure storage of user confidential data for a certain fixed specified term. Then the network of servers perform calculations within the framework of a combination of a number of cryptographic protocols, such as the distributed key generation protocol based on discrete logarithmization on elliptic curves, the Pedersen verifiable threshold secret sharing protocol and the ElGamal encryption algorithm on elliptic curves, ensuring secure storage of the corresponding secret data. An important property of this system is the ability to function even if a certain number of servers fail, not exceeding a certain threshold. The main component in this model is a distributed network of servers that interact by sending each other working messages provided by the system operation protocol. Some of them may be faulty, i.e., either completely fail and never resume operation, or stop sending working messages to other servers, start behaving randomly and arbitrarily. It is assumed that each server receiving a message always knows the identifier of the server that sent the message, and that the communication environment is reliable, i.e., all messages are delivered without errors. Moreover, the network consists of a fixed number of servers.

This paper proposes to consider a version of the model in which the network consists of a non-constant number of servers (but also at least 3), and its implementation using distributed ledger technologies. It is proposed to consider a consensus protocol based on artificial intelligence as a consensus mechanism. The main idea is to use RL to optimize the formation of p2p groups of servers that perform all the required calculations during the operation of a cryptographic backup system for confidential data. RL is a type of machine learning, representing a set of algorithms for solving problems that are expressed in the form of Markov decision-making processes. The main components are the environment and a set of states of the environment, learning agents and a set of actions available to agents, transitions between states, and rewards.

In this article we consider an idea of integration MCDM and RL methods to selection of blockchain consensus mechanisms for IoT networks. We propose a integrated system for selecting and managing consensus protocols in IoT networks. based on MCDM and RL methods. The MCDM-RL-based framework for consensus protocol selection in IoT networks proposed in this paper is a relatively new approach to solving the problem of selecting, managing, and optimizing consensus protocols in complex, dynamic IoT environments.

The paper is organized as follows: section 2 reviews the details about consensus problems, consensus protocols, classifications of consensus protocols, also it is devoted to MCDM and RL methods and the issue of their integration for consensus protocol selection in IoT networks. Section 3 considers the use of RL to optimize the formation of p2p groups of servers that perform all the required calculations during the operation of a cryptographic backup system for confidential data, and introduces the MCDM-RL-based decision framework for blockchain consensus protocol selection for IoT. Finally, section 4 concludes the work.

2. METHOD

In this section, we describe the approach we used to integrate MCDM and RL for selecting consensus protocols in blockchain-based IoT networks. We start by discussing the key challenges associated with consensus mechanisms and providing an overview of existing protocols. Next, we introduce the concepts of MCDM and RL, explaining how each method contributes to the decision-making process. Finally, we discuss a novel approach that combines MCDM and RL, leveraging their complementary strengths to improve the selection of consensus protocols in dynamic IoT environments.

2.1. Consensus problems and consensus protocols

In this section we review the details about consensus problems, consensus protocols, classifications of consensus protocols. Generally, the notion of consensus problem is related with the notion of agreement problem for distributed systems [19]-[22]. The agreement problem can be defined as follows.

Agreement problem. Let $P = \{p_1, p_2, \dots, p_n\}$ some finite set of processors, which interact by sending messages to each other. Some subset $F \subset P$ processors are faulty. Each processor p_i has some value x_i. The processors have to reach agreement on some common value (a common set of values). This problem has the following parameters.

– A processor receiving a message always knows the identity of the processor that sent the message.
– The communication environment is reliable, i.e., all messages are delivered without errors.
– Processors can be non-faulty or faulty. The processor can have three types of faults.
  a. Crash fault: the processor ceases operation permanently and does not restart.
  b. Omission fault: processor "forgets" to send messages to other processors.
  c. Unexplained fault (malicious fault, Byzantine fault): the processor behaves randomly and arbitrarily.
– Processors can be synchronous, asynchronous, or partially synchronous. In the first case, there is a constant $s \geq 1$ such that in any time interval during which any processor makes $s + 1$ steps, every other non-faulty processor must make at least one step. In the second case, a processor may experience an indefinite wait, but the time elapsed between its own steps remains finite. However, if the processor performs only a finite number of steps when the system starts up indefinitely, then it fails. In the third case, either a constant $s \geq 1$ such that in any time interval during which any processor makes $s + 1$ steps, every other healthy processor must make at least one step exists, but is unknown, or such a constant is known, but must be taken into account from some unknown point in time T, called the global stabilization time (GST). In other words, in the third case, the processors are asynchronous until time T, and become synchronous after.
– The communication may be synchronous (i.e., the communication delay may be limited), asynchronous (i.e., the communication delay may be unlimited), or the communication may be partially synchronous. In the first case, there is a constant $\Delta \geq 1$ such that any message sent by the processor must be delivered in Δ steps in real time. In the second case, message delivery may take an unpredictable but finite amount of real time. However, over the course of any infinite system execution, every message is ultimately delivered, ensuring that no messages are lost. In the third case, either a constant $\Delta \geq 1$ such that any message sent by the processor must be delivered in Δ steps in real time exists, but is unknown, or such a constant is known, but must be taken into account from some unknown point in time T, called GST. In other words, in the third case, communication is asynchronous up to time T, and becomes synchronous after.
– Messages passed between processors may be ordered or unordered. In the first case, if some processor p sends a message m_1 to processor r at some real time t_1, and some processor p sends a message m_2 to processor r at real time $t_2 > t_1$, then processor r receives message m_1 before message m_2. In the second case, the messages may be delivered out of order.
  a. Messages can be sent over a point-to-point connection or over a broadcast channel.
  b. Sending and receiving messages can be carried out simultaneously or separately.
  c. Messages passed between processors can be authenticated (signed messages) or unauthenticated (oral messages). In the first case, a faulty processor is unable to fabricate a message or alter the content of a received one. Additionally, it can verify the authenticity of incoming messages. In contrast, in the second case, a faulty processor can create fraudulent messages, falsely attributing them to another processor, or modify the content of received messages before forwarding them. In this case, there is no mechanism for verifying the authenticity of received messages.

The cryptographic protocol for solving the agreement problem is called the agreement protocol. The agreement protocol must have the following properties.

– Property 1. (Agreement) all non-faulty processors must agree on the same value (set of values).
– Property 2. (Validity) if all non-faulty processors have the same initial value (set of initial values), then the agreed value (set of values) of all non-faulty processors must be the same value (set of values).
– Property 3. (Termination) each non-faulty processor must eventually determine the value (set of values).

There are the following variants of the agreement problem:

– The Byzantine agreement problem (also known as the Byzantine generals problem). In the Byzantine agreement problem, the only value that must be agreed upon is initialized by an arbitrary processor, and all non-faulty processors must agree on this value.
– The consensus problem. In the consensus problem, every processor starts with an initial value, and all non-faulty processors must reach an agreement on a single shared value.
– The interactive consistency problem. In the interactive consistency problem, each processor also has an initial value, but all non-faulty processors must agree on a common set of values.

Accordingly, there are the following variants of agreement protocols.

The Byzantine agreement protocol is a cryptographic agreement protocol for solving the Byzantine agreement problem. Accordingly, the protocol must have the following properties.
- Property 1. (Agreement) all non-faulty processors must agree on the same value.
- Property 2. (Validity) if the processor that initializes the single value to be negotiated is healthy, then the consensus value of all non-faulty processors must be the same as the original value of the source.
- Property 3. (Termination) each non-faulty processor must eventually determine the value.

The consensus protocol is a cryptographic agreement protocol for solving the consensus problem. It must have the following properties:
- Property 1. (Agreement) all non-faulty processors must agree on the same (single) value.
- Property 2. (Validity) if all non-faulty processors have the same initial value, then the agreed value of all non-faulty processors must be the same value.
- Property 3. (Termination) each non-faulty processor must eventually determine the value.

The interactive consistency protocol is a cryptographic agreement protocol for solving the interactive consistency problem. The interactive consistency protocol must have the following properties.
- Property 1. (Agreement) all non-faulty processors must agree on the same set of values $X = \{x_1, x_2, \dots, x_n\}$.
- Property 2. (Validity) if the processor P_i is non-faulty and it has an initial value x_i, then all non-faulty processors agree on the value x_i as the i-th element of the set X. If the processor P_j is faulty, then all non-faulty processors can agree on an arbitrary value as j-th element of the set X.
- Property 3. (Termination) each non-faulty processor must eventually determine a set of values X.

All these variants of the agreement problem are closely interconnected. Despite the formal differences, the solution of any of them can be applied to solve the other two problems [19]-[22]. Therefore, the use of the terms "agreement problem" and "consensus problem" in the literature are used interchangeably.

Being an implementation of a distributed system, the blockchain system relies on a consensus protocol to ensure that all nodes in the network agree on a single chain of transaction history, taking into account the adverse impact of faulty and malicious nodes. Currently, there are more than a thousand initiatives in the field of distributed ledger technologies, including over a hundred of different consensus protocols. In recent years, there has been a noticeable increase in publications focusing on blockchain and distributed ledger technologies. Given that consensus mechanisms are a core aspect of these systems, it's not surprising that more research has been dedicated to the development, evaluation, comparison, and selection of consensus protocols. For instance, studies such as [9]-[14], [19]-[25] have reviewed, analyzed, and compared various consensus protocols using different criteria. These works offer comprehensive insights into the characteristics of the protocols and the factors influencing their performance and security. For understanding and analyzing this huge amount of consensus algorithms researchers make its classifications, taxonomies and ontologies.

Consensus protocols can be classified based on their execution, resource use, system design, and other factors. By execution type, protocols are deterministic, depending only on initial values and failed processors' behavior; randomized, using random steps; or probabilistic, relying on system assumptions [19]-[22]. In terms of resource use, compute-intensive protocols consume high energy, capability-based protocols select miners based on factors like wealth or trust, and voting-based protocols reduce energy consumption and address wealth dominance [19]. Protocols can also be categorized by effort, wealth, reputation, or representation [10]. Effort-based protocols require computational proof, wealth-based rely on staked resources, reputation-based reward good behavior, and representation-based involve elected nodes acting on behalf of others. Mechanism-based classifications include competitive methods where only one solution is accepted, comparative approaches based on stake, vote-based systems, non-linear alternatives like directed acyclic graph (DAG) or sidechains, and collaborative models combining algorithms [13]. Other classifications identify traditional approaches like Paxos, hybrid designs incorporating proof or Byzantine fault tolerance (BFT), and novel alternatives addressing protocol deficiencies [9]. High-level distinctions include proof-based systems like proof of work (PoW) and proof of stake (PoS) or voting-based systems like BFT and CFT, which address node failures and malicious behavior [20]. Process models outline phases like leader election, block addition, and transaction confirmation, with modes such as leader-based, voting-based, committee-based, and fair accounting [22]. Leader election strategies vary, including proof-based methods requiring qualifications, voting-based balanced systems, randomness-based selections, alliance-based representative groups, or combinations of these [23]. Structurally, protocols can use linear models, such as PoW and PoS, or DAG-based designs like blockDAG and txDAG [24], [25]. These classifications help optimize consensus protocols for energy efficiency, fairness, and fault tolerance.

Numerous traditional consensus mechanisms exist; however, they fail to fully satisfy the requirements of the IoT. According to Wen *et al.* [14], blockchain consensus mechanisms tailored for IoT networks can be classified into four categories: security-focused mechanisms, scalability-oriented mechanisms, energy-efficient mechanisms, and performance-enhancing mechanisms.

2.2. Multi-criteria decision making

The integration of MCDM and RL for consensus protocol selection in IoT networks is a relatively new approach. However, each component has been separately explored in various contexts. MCDM methods are used to evaluate and select the optimal option among several alternatives based on multiple criteria. They are applied when a decision must take into account several factors, often conflicting ones. These methods find wide application in management, planning, engineering, and other fields where it is important to consider multiple criteria to make optimal decisions. MCDM provides a structured methodology for evaluating and ranking alternatives based on multiple criteria. It has been well established in decision theory and operations research for resolving trade-offs between competing objectives [15], [16].

The selection of blockchain consensus protocol is formalized as an MCDM task as follows [16]. The primary objective of decision-making is to identify the best alternative or create a ranked list of alternatives by utilizing multi-criteria decision-making methods, based on the decision matrix, criterion weights, and any applicable constraints. While the number of alternatives is generally not limited, certain methods in multi-criteria decision-making are more effective when applied to a limited set of criteria and alternatives. As a result, selecting the most appropriate list of alternatives is a critical task. For choosing consensus protocols in IoT networks, existing research and surveys on blockchain consensus mechanisms for IoT networks can be referenced [10], [13], [14]. Before applying a multi-criteria decision-making method, the collected data must be processed to meet the input requirements of the chosen method (or the MCDM tool being used). Additionally, it is necessary to define the category for each criterion. Since some MCDM methods only operate with quantitative data, any qualitative criteria must first be converted into numerical values.

After formulating the decision problem and defining the set of consensus protocols, it is essential to establish appropriate criteria for evaluating the alternatives. These criteria represent the various factors by which the alternatives are assessed. In a MCDM approach, the criteria are typically classified into two groups: benefit criteria, which are preferred in higher values and need to be maximized, and cost criteria, where lower values are preferred and need to be minimized. Blockchain experts are typically the primary source for determining the most suitable set of criteria, although data from academic research and studies can also be leveraged [9]-[14].

The next step involves gathering data on the performance of the alternatives across each criterion and organizing this information into a decision matrix $X = [x_{ij}]_{m\times n}$, where each element represents the performance of alternative A_i with respect to criterion C_j. Since the criteria are often expressed in different units of measurement, the matrix X is usually heterogeneous. Additionally, some criteria may be qualitative, relying on subjective linguistic descriptions like "low," "medium," or "high." In such cases, the nine-point Saaty scale is often used to convert these qualitative evaluations into numerical values, providing more detail with its five main levels and intermediate points. After converting all criterion values into numerical form, the decision matrix can be normalized using a suitable MCDM method. It is also advisable to eliminate dominated alternatives before applying multi-criteria analysis for ranking.

Special attention must be given to determining the weights of the criteria, as they significantly influence the decision-making outcome. Furthermore, uncertainties may arise in assigning weights, which must be addressed. Techniques for assigning weights are typically classified into three main categories [15], [16].

Subjective weighting methods require decision-makers to have a deep understanding of the problem, usually relying on expert knowledge. Approaches like the analytic hierarchy process (AHP) or the Delphi method are commonly used to derive aggregate weights in these cases. Objective weighting methods, on the other hand, are based solely on data and calculate weights using mathematical approaches (e.g., entropy). These methods provide expert-independent weights, though they may not always reflect the true importance of criteria. Integrated weighting methods combine both subjective and objective approaches, merging expert-derived weights with mathematical data-driven insights. Experts in MCDM are essential for selecting the most appropriate method, with recommendations from peer-reviewed literature also providing valuable guidance.

In the final stage, a multi-criteria decision-making method is applied to compute a value that ranks the alternatives. Numerous MCDM methods exist, each with unique assumptions, characteristics, and limitations, which may yield varying results when applied to the same problem. Thus, it is often useful to assess the decision problem using multiple methods. Additionally, uncertainty analysis (such as sensitivity analysis) can be conducted to evaluate the robustness of the results [15], [16].

2.3. Reinforcement learning

RL is a category of machine learning techniques in which an agent learns to make decisions based on its interaction with the environment. The agent's goal is to maximize the total reward by choosing the optimal actions based on its experience interacting with the environment. The main elements of RL are an agent, a learning system (algorithm) that interacts with the environment and makes decisions, an environment, the surrounding system with which the agent interacts, providing the agent with information about its state, actions, a set of possible decisions that the agent can make at each step of interaction with the environment, states, a description of the current state of the environment, which the agent uses to make decisions, a reward, feedback from the environment that the agent receives for each action, stimulating it to achieve the goal, and which informs the agent how successful its action was (the reward can be positive (success) or negative (error)), a policy, a strategy by which the agent chooses its actions depending on the current state of the environment, an objective function, which shows how profitable it is to be in a certain state, based on possible future rewards, an action value function (Q-function), which reflects the value of a specific action in a specific state, taking into account the expected reward. The learning process is as follows. The agent makes decisions based on the current state of the environment and performs actions. The environment, in turn, changes under the influence of these actions and returns a new state and reward to the agent. Based on this, the agent updates its policy, aiming to choose those actions that will maximize the total reward in the long run. RL has applications in various fields, such as games (algorithms playing go or chess), robotics (optimizing robot movement), resource management (energy, computing power allocation), autonomous systems (unmanned transport systems). RL allows agents to learn complex sequences of actions without the need for full knowledge of the environment, making it a powerful tool for creating adaptive systems [1], [2].

2.4. Combination of multi-criteria decision making and reinforcement learning

The combination of MCDM's ability to handle multiple performance criteria with RL's adaptability to changing conditions offers a more comprehensive and flexible solution for protocol selection in IoT networks. This integration is intended to provide a more dynamic and context-sensitive approach than what traditional methods can offer. Although there are examples of application of MCDM and RL separately, the specific combination for choosing an IoT consensus protocol is a relatively new direction. This new integration aims to leverage the strengths of both approaches, providing a more robust solution that can better handle the complexities and evolving nature of IoT networks. RL is widely recognized for its ability to learn optimal policies in dynamic environments through trial and error. On the other hand, MCDM provides structured frameworks for evaluating and prioritizing multiple conflicting criteria, establishing it as a valuable tool for decision-making in intricate systems. When combined, RL and MCDM can address issues such as balancing energy efficiency, latency, and resource allocation in IoT systems. In the context of IoT, there are several alternative methodologies for decision making, including heuristic optimization algorithms, fuzzy logic systems, and supervised machine learning models [26], [27]. Approaches such as genetic algorithms and particle swarm optimization are commonly used for resource optimization in IoT. Although these methods are effective in certain scenarios, they lack the adaptivity and real-time learning capabilities of RL. Fuzzy logic has been used for decision making under uncertainty, especially in energy management and task scheduling. However, it requires predefined rules and lacks the exploratory nature of RL. Methods such as support vector machines and neural networks have been applied to predictive modeling and classification tasks in IoT. These methods are data-driven, but often require large labeled datasets and do not inherently support dynamic decision making. Compared to these approaches, integrating RL and MCDM offers unique advantages by combining real-time learning with structured multi-objective evaluation. For example, RL’s ability to explore and adapt to changing IoT environments complements MCDM’s systematic prioritization of conflicting objectives. The IoT ecosystem is characterized by its heterogeneity and dynamic nature, with devices operating under different constraints and objectives. Combining RL with MCDM addresses these challenges by providing adaptability, multi-objective optimization, and scalability. RL dynamically adjusts to changes in the environment, such as changing network conditions or energy availability. MCDM provides a structured approach for balancing competing objectives such as latency, energy consumption, and throughput. The hybrid approach can scale to accommodate the growing complexity of IoT networks.

3. RESULTS AND DISCUSSION

3.1. Reinforcement learning-based consensus protocol

This section proposes a consensus protocol based on artificial intelligence as a mechanism for a cryptographic system model designed for backup storage of confidential data in secure servers, as presented in [17], [18]. The system operates through a portal for user requests, a distributed server network for computations and secure data storage, and the user (Figure 1). Users submit requests for secure storage of confidential data for a fixed term. The server network processes these requests using cryptographic protocols such as distributed key generation, Pedersen threshold secret sharing, and ElGamal encryption on elliptic curves. The system remains functional even if some servers fail, provided the failures do not exceed a certain threshold. Servers exchange messages according to a reliable communication protocol, ensuring all messages are delivered without errors. While the network requires at least three servers, this paper explores a version with a variable number of servers, also utilizing distributed ledger technologies.

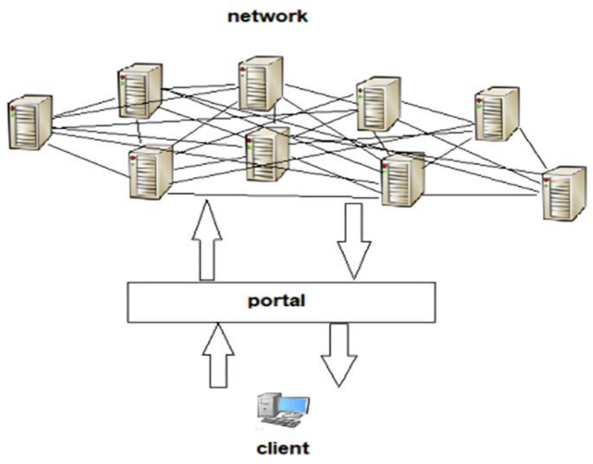


Figure 1. The cryptographic system for backup storage of confidential data in secure servers

The main idea is to use deep RL to optimize the formation of p2p groups of servers that perform all the required calculations during the operation of a cryptographic backup system for confidential data. RL is a category of machine learning techniques, representing a set of algorithms for solving problems that are expressed in the form of Markov decision-making processes. The main components are the environment and a set of environment states, learning agents and a set of actions available to agents, transitions between states, and rewards. The environment of our RL is a distributed network of servers. The set of possible observable states of the environment are vectors $S = (s_1, s_2, \dots, s_k)$ of values of all available parameters of the network as a whole and its nodes (servers), such as the total number of servers, the total number of all working messages, the total number of active servers at the current time, the lifetime of the server (the time since it was first added to the network), the uptime of the server, the total duration of active work in the network (the amount of time during which it was active and connected to the network), the total number of all connections (which the server has with other servers in the network), the time required for the server to receive and transmit data over the network, the amount of data that the server is capable of transmitting and receiving, the capacity of the server storage, the amount of data that the server is capable of storing, and others. Based on the server parameters, a certain evaluation indicator (weight) of the server in the network is calculated. For example, if $p_1, p_2, \dots, p_n$ are the values of all available server parameters, then the server weight can be calculated as $W(p_1, p_2, \dots, p_n) = (t_1 * p_1 + t_2 * p_2 + \ \dots + t_n * p_n)$, where $t_1, t_2, \dots, t_n$ are normalizing factors ($t_i = \frac{100}{p'_i}$, p'_i is the “ideal” value of the parameter p_i), $0 \leq W \leq 100$. The goal of training is to form p2p groups of servers with optimal indicators for further calculations provided by the system protocol. The set of actions that training agents can take is the selection of servers to form a p2p group.

Since the environment state is generated deterministically, the next state is guaranteed to follow from the current state. The environment pays rewards to RL agents based on optimal computation execution times. The well-known proximal policy optimization (PPO) algorithm is assumed as the RL algorithm. PPO was chosen for its robustness and efficiency in handling complex, high-dimensional action spaces. PPO balances exploration and exploitation by limiting the magnitude of policy updates, which helps maintain stability during training. The main reasons for the choice also include sampling efficiency, scalability, and robust convergence. PPO makes efficient use of the collected data, which is critical in scenarios where modeling can be computationally expensive. It is assumed that the network parameters will be collected in real time using distributed monitoring tools. The collected data is aggregated into state vectors and then fed to the algorithm for training and inference. PPO has also demonstrated efficiency in distributed systems with large-scale interactions, making it suitable for the consensus protocol under consideration. PPO includes a clipping mechanism that limits how much the policy can be changed at each step. This makes the algorithm more stable and predictable. This minimizes the risk of performance collapse during training, which is critical in real-time systems.

A comparative discussion of alternative RL methods can provide a more holistic understanding of the trade-offs involved. For instance, deep q-networks (DQN) are computationally lighter than PPO due to their discrete action space but are less effective for problems involving high-dimensional or continuous action spaces. On the other hand, trust region policy optimization (TRPO), a precursor to PPO, ensures even stricter stability in updates but is computationally more demanding, making it less practical for real-time applications. Actor-Critic methods, such as A2C or A3C, provide alternative approaches with their distinct balance between performance and computational efficiency. However, they often lack the simplicity and scalability advantages that PPO offers.

3.2. Multi-criteria decision making-reinforcement learning-based decision framework

This section introduces the MCDM-RL-based decision framework for blockchain consensus protocol selection for IoT (Figure 2). The proposed system enables step-by-step evaluation and adaptation of consensus protocols for IoT networks.

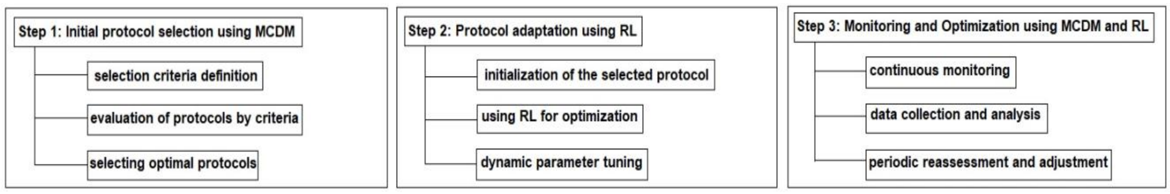

Step 1: Initial protocol selection using MCDM
selection criteria definition
evaluation of protocols by criteria
selecting optimal protocols
Step 2: Protocol adaptation using RL
initialization of the selected protocol
using RL for optimization
dynamic parameter tuning
Step 3: Monitoring and Optimization using MCDM and RL
continuous monitoring
data collection and analysis
periodic reassessment and adjustment

Figure 2. The MCDM-RL-based decision framework to consensus protocol selection for IoT

In the first step, MCDM methods are used to initially select one or more protocols that are best suited for a particular IoT network based on a set of criteria. Thus, in the first step, MCDM methods help candidates, which can then be trained and adapted using RL methods in the second step. The selected protocols can be further optimized using RL to improve efficiency in changing conditions. RL can dynamically adapt the selected protocol to changing environmental conditions, optimizing its operation in real time. Using RL methods allows network nodes to dynamically adjust protocol parameters to current operating conditions. For example, if the network encounters a large number of IoT devices, RL can adapt protocol parameters (e.g., block generation frequency or the number of participants required for consensus) to minimize delays and energy losses.

Finally, the system allows monitoring and quality control of the currently selected and running protocols and, if necessary, optimizing and retraining them. MCDM and RL can work together to periodically re-evaluate and retrain the protocol, adapting to changing network requirements and loads. The system can periodically re-evaluate the protocol using MCDM to track its performance. If the protocol is no longer optimal, RL can learn new consensus strategies or the system can switch to another protocol.

In highly dynamic IoT networks with highly variable operating conditions (e.g., smart cities, industrial IoT networks), dynamic adaptation of consensus protocols is required to minimize latency and energy consumption. Using MCDM methods, several suitable consensus protocols can be selected at the system design stage based on the characteristics of devices and the network. Then, consensus protocols based on RL will allow the network to dynamically adapt to loads and ensure stable operation even under changing conditions (e.g., an increase in the number of devices, communication channel congestion). Thus, the combination of MCDM and RL allows us to create a system where the initial choice of a consensus protocol is based on objective criteria, and further adaptation is carried out dynamically to improve the efficiency of the IoT network.

The proposed approach offers several advantages. It offers adaptability by integrating MCDM and RL, enabling the system to adjust consensus protocols based on evolving conditions and the characteristics of the IoT network. The proposed system is capable of providing energy efficiency. RL reduces energy consumption by optimizing the consensus process. Additionally, this system is capable of scalability. The approach is easily adapted to scalable systems where the load and number of devices are constantly changing. The proposed system is capable of ensuring security and reliability. By adapting the consensus parameters in real time, the system is more resilient to external attacks and failures.

MCDM allows for explicit consideration of various performance metrics, while RL can adapt to changes over time, providing a dynamic and adaptive decision-making framework. Combining these approaches can address both the static evaluation of criteria and the need for adaptability in dynamic environments, which is critical for IoT networks that are subject to frequent changes. The integration of MCDM and RL is supported by empirical studies in other domains where combining different decision-making paradigms has proven effective. For example, in domains such as autonomous systems, financial decision-making, and complex resource management, such integrations have shown improvements in performance and adaptability. In practice, such a framework can be tested and validated through simulations and real-world experiments consistent with scientific methodology. By applying this integrated approach to IoT networks, researchers can evaluate its effectiveness and refine the model based on empirical data.

The MCDM-RL-based framework represents an innovative approach to solving complex protocol selection problems in IoT networks. The combination of these methodologies leverages their individual strengths and addresses their limitations, making it a promising direction for further research and application.

3.3. On practical implementation of the multi-criteria decision making-reinforcement learning-based decision framework

A fundamental step in the practical implementation of the proposed framework is the effective collection of data for evaluation, training, and validation of the system. The data collection process begins with the identification of relevant sources of real-time data characterizing the key performance indicators required for IoT networks. The choice of modeling tools allows to supplement real data and conduct experiments and testing. Various existing industry reports and studies, as well as open-source repositories of consensus protocols, provide data on performance indicators. The toolkit for data collection and analysis includes performance monitoring tools (e.g., Grafana and Prometheus), blockchain analytics tools (e.g., BlockSci and Etherscan), IoT device emulators (e.g., Cooja and NS-3), data logging frameworks (e.g., Apache Kafka), and analytics tools (e.g., Pandas and NumPy).

The next important steps are the selection of evaluation criteria. The evaluation criteria should cover the main properties of blockchain consensus protocols that affect their suitability for IoT networks. Using various methods, certain weights are assigned to the selected criteria to reflect their importance. Next, weight normalization and sensitivity analysis are performed to assess the reliability of the weights. In addition, mechanisms for periodic re-evaluation of the weights are needed, given the dynamic nature of IoT environments. Next, after the criteria and alternatives have been defined, an MCDM method is selected. Suitable candidates include methods such as AHP, TOPSIS, and SAW. These methods systematically rank alternatives by quantifying trade-offs between criteria.

RL complements the MCDM process by dynamically optimizing and adapting selected protocols in IoT networks. It does this by defining a state space of the environment that the RL agent will observe (e.g., network conditions, protocol performance, environmental externalities). A set of actions that the RL agent can perform to optimize the protocol (adjust consensus parameters, switch to another protocol) is specified. A reward function is designed to incentivize the desired outcomes (e.g., minimize latency and energy consumption, maximize throughput and security). The RL agent is trained using an appropriate algorithm (e.g., DQN, policy gradient methods, and PPO). The RL agent is trained in a simulated IoT environment to explore different states and actions. The resulting model is validated using real data or test scenarios to ensure generalizability.

The cornerstone of the developed framework is the integration process, which ensures a seamless data flow between the MCDM and RL components. Initially, MCDM methods identify the most suitable consensus protocols by evaluating their performance against certain criteria. The outputs of the MCDM stage, such as ranked protocols or weighted scores, serve as input to the RL stage. These results help define the action space and set the initial conditions for the RL agent. The RL agent, trained in a simulated or real IoT environment, uses the MCDM ratings to focus on optimizing the parameters of the most promising protocols. Feedback from the RL agent, such as the observed performance of the protocol under different conditions, can in turn be fed back into the MCDM process for re-evaluation. This bidirectional flow ensures that decisions remain adaptive and informed by both structured evaluation and empirical learning.

To evaluate the performance of the framework being developed, reliable evaluation metrics are needed to reflect the system’s performance, efficiency, adaptability, accuracy, and scalability. For verification, a dedicated simulation environment is needed to evaluate the framework in various IoT network scenarios, as well as deploy it in a small network with real devices for real-world testing. Statistical methods are used to analyze the results and validate the effectiveness. By following this detailed approach, the MCDM-RL framework can be thoroughly evaluated, ensuring that it meets the diverse requirements of IoT networks while offering robust and adaptive decision making.

Practical implementation of the MCDM-RL framework faces challenges like computational complexity, the need for diverse and reliable data, and seamless integration of its components. Scalability in large IoT networks and balancing MCDM criteria with RL's exploration-exploitation trade-offs further complicate deployment.

Traditional static or heuristic methods lack adaptability and fail to balance priorities in dynamic IoT environments. The MCDM-RL framework addresses these issues by combining multi-criteria evaluation with real-time optimization. MCDM ensures comprehensive assessment, while RL adapts dynamically, learning from interaction with the environment. This approach offers a scalable, flexible, and robust solution for IoT blockchain consensus selection, surpassing traditional methods.

4. CONCLUSION

In this paper, we introduced a conceptual framework that combines MCDM and RL to select blockchain consensus protocols for IoT networks. We proposed this approach because IoT systems encounter distinct difficulties, including constrained resources, changing network conditions, and scalability demands. At this stage, our framework remains theoretical-practical implementation and validation through simulations are yet to be conducted.

The decision to present this as a concept rather than a fully modeled system in our paper can be explained by several reasons. First, IoT environments vary significantly in terms of device capabilities, communication protocols, and application-specific requirements, making it difficult to develop a universal model. Crafting a comprehensive modeling framework that accurately reflects the varied and dynamic nature of IoT networks is both complex and essential. Second, integrating MCDM and RL requires further research to balance decision accuracy and computational efficiency. The interplay between these components introduces complexities that must be systematically addressed prior to deployment. Third, there is currently no standardized benchmark or dataset for evaluating blockchain consensus mechanisms in IoT, which complicates meaningful validation. Establishing such standards is crucial for effectively validating the framework's efficacy.

Our work lays the foundation for future research, and the next step is to move from concept to practical implementation. To achieve this, we plan to: i) define key evaluation criteria for MCDM, including scalability, latency, energy efficiency, and security; ii) develop simulation environments that closely mimic real-world IoT networks, accounting for factors like device density fluctuations, resource constraints, and diverse communication protocols; iii) explore advanced RL techniques, such as multi-agent RL and federated learning, to enable decentralized decision-making and reduce computational overhead; iv) implement mechanisms for continuous RL model updates, ensuring the system adapts to changing network conditions in real time; v) prototype the framework and test it in controlled IoT environments, such as smart homes or industrial IoT applications, to assess real-world performance; vi) evaluate the system’s ability to switch between different consensus protocols dynamically, depending on network conditions such as traffic surges or resource limitations; vii) examine the framework’s scalability in large-scale IoT networks with thousands or even millions of devices, optimizing computational efficiency where necessary; viii) integrate security measures into the RL optimization process, ensuring resilience against potential cyber threats; and ix) establish key success benchmarks, such as reduced consensus time, energy savings, and adaptability, and compare our approach with existing solutions. By following this step-by-step approach, we aim to transition our conceptual framework into a practical, adaptive, and efficient decision-making system for IoT networks.

FUNDING INFORMATION

This research has been funded by the SC of the MSHE of the RK (Grant No. AP23487259).

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Nurlan Tashatov	✓	✓				✓			✓	✓			✓	
Ruslan Ospanov	✓	✓			✓	✓			✓	✓	✓			
Yerzhan Seitkulov		✓			✓					✓			✓	
Dina Satybaldina		✓				✓				✓		✓		
Banu Yergaliyeva		✓			✓					✓	✓			

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] G. Dulac-Arnold *et al.*, "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis," *Machine Learning*, vol. 110, no. 9, pp. 2419–2468, Sep. 2021, doi: 10.1007/s10994-021-05961-4.

[2] T. Zhang and H. Mo, "Reinforcement learning for robot research: A comprehensive review and open issues," *International Journal of Advanced Robotic Systems*, vol. 18, no. 3, pp. 1–22, May 2021, doi: 10.1177/17298814211007305.

[3] H. Zhang, X. Zhang, Z. Guo, H. Wang, D. Cui, and Q. Wen, "Secure and Efficiently Searchable IoT Communication Data Management Model: Using Blockchain as a New Tool," *IEEE Internet of Things Journal*, vol. 10, no. 14, pp. 11985-11999, Jul. 2023, doi: 10.1109/JIOT.2021.3121482.

[4] E. A. Shammar, A. T. Zahary, and A. A. Al-Shargabi, "A Survey of IoT and Blockchain Integration: Security Perspective," *IEEE Access*, vol. 9, pp. 156114-156150, 2021, doi: 10.1109/ACCESS.2021.3129697.

[5] A. A. Khan, A. A. Laghari, Z. A. Shaikh, Z. Dacko-Pikiewicz, and S. Kot, "Internet of Things (IoT) Security With Blockchain Technology: A State-of-the-Art Review," *IEEE Access*, vol. 10, pp. 122679-122695, 2022, doi: 10.1109/ACCESS.2022.3223370.

[6] B. K. Mohanta, D. Jena, S. Ramasubbareddy, M. Daneshmand, and A. H. Gandomi, "Addressing Security and Privacy Issues of IoT Using Blockchain Technology," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 881-888, Jan. 2021, doi: 10.1109/JIOT.2020.3008906.

[7] L. D. Xu, Y. Lu, and L. Li, "Embedding Blockchain Technology Into IoT for Security: A Survey," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10452-10473, July 2021, doi: 10.1109/JIOT.2021.3060508.

[8] H. -N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8076-8094, Oct. 2019, doi: 10.1109/JIOT.2019.2920987.

[9] B. Lashkari and P. Musilek, "A Comprehensive Review of Blockchain Consensus Mechanisms," *IEEE Access*, vol. 9, pp. 43620–43652, 2021, doi: 10.1109/ACCESS.2021.3065880.

[10] D. P. Oyinloye, J. S. Teh, N. Jamil, and M. Alawida, "Blockchain consensus: An overview of alternative protocols," *Symmetry*, vol. 13, no. 8, pp. 1–35, Jul. 2021, doi: 10.3390/sym13081363.

[11] M. Salimitari, M. Chatterjee, and Y. P. Fallah, "A survey on consensus methods in blockchain for resource-constrained IoT networks," *Internet of Things*, vol. 11, 100212, 2020, doi: 10.1016/j.iot.2020.100212.

[12] J. Huang, L. Kong, G. Chen, M. -Y. Wu, X. Liu, and P. Zeng, "Towards Secure Industrial IoT: Blockchain System With Credit-Based Consensus Mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680-3689, Jun. 2019, doi: 10.1109/TII.2019.2903342.

[13] M. Khan, F. den Hartog, and J. Hu, "A Survey and Ontology of Blockchain Consensus Algorithms for Resource-Constrained IoT Systems," *Sensors*, vol. 22, no. 21, pp. 1–32, Oct. 2022, doi: 10.3390/s22218188.

[14] Y. Wen, F. Lu, Y. Liu, P. Cong, and X. Huang, "Blockchain Consensus Mechanisms and Their Applications in IoT: A Literature Survey," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12454 LNCS, 2020, pp. 564–579, doi: 10.1007/978-3-030-60248-2_38.

[15] S. K. Sahoo and S. S. Goswami, "A Comprehensive Review of Multiple Criteria Decision-Making (MCDM) Methods: Advancements, Applications, and Future Directions," *Decision Making Advances*, vol. 1, no. 1, pp. 25–48, Dec. 2023, doi: 10.31181/dma1120237.

[16] E. Filatovas, M. Marcozzi, L. Mostarda, and R. Paulavičius, "A MCDM-based framework for blockchain consensus protocol selection," *Expert Systems with Applications*, vol. 204, Oct. 2022, doi: 10.1016/j.eswa.2022.117609.

[17] B. Yergaliyeva, Y. Seitkulov, D. Satybaldina, and R. Ospanov, "On some methods of storing data in the cloud for a given time," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 20, no. 2, pp. 366–372, Apr. 2022, doi: 10.12928/TELKOMNIKA.v20i2.21887.

[18] O. Tasmagambetov, Y. Seitkulov, R. Ospanov, and B. Yergaliyeva, "Fault-tolerant backup storage system for confidential data in distributed servers," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 21, no. 5, pp. 1030–1038, Oct. 2023, doi: 10.12928/TELKOMNIKA.v21i5.25305.

[19] L. Ismail and H. Materwala, "A review of blockchain architecture and consensus protocols: Use cases, challenges, and solutions," *Symmetry*, vol. 11, no. 10, pp. 1–47, Sep. 2019, doi: 10.3390/sym11101198.

[20] S. J. Alsunaidi and F. A. Alhaidari, "A survey of consensus algorithms for blockchain technology," in *In 2019 International Conference on Computer and Information Sciences (ICCIS)*, IEEE, Apr. 2019, pp. 1–6, doi: 10.1109/ICCISci.2019.8716424.

[21] M. Belotti, N. Božić, G. Pujolle, and S. Secci, "A Vademecum on Blockchain Technologies: When, Which, and How," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 4, pp. 3796–3838, 2019, doi: 10.1109/COMST.2019.2928178.

[22] X. Fu, H. Wang, and P. Shi, "A survey of Blockchain consensus algorithms: mechanism, design and applications," *Science China Information Sciences*, vol. 64, no. 2, p. 121101, Feb. 2021, doi: 10.1007/s11432-019-2790-1.

[23] Q. Bao, B. Li, T. Hu, and X. Sun, "A survey of blockchain consensus safety and security: State-of-the-art, challenges, and future work," *Journal of Systems and Software*, vol. 196, Feb. 2023, doi: 10.1016/j.jss.2022.111555.

[24] A. Singh, G. Kumar, R. Saha, M. Conti, M. Alazab, and R. Thomas, "A survey and taxonomy of consensus protocols for blockchains," *Journal of Systems Architecture*, vol. 127, Jun. 2022, doi: 10.1016/j.sysarc.2022.102503.

[25] A. K. Yadav, K. Singh, A. H. Amin, L. Almutairi, T. R. Alsenani, and A. Ahmadian, "A comparative study on consensus mechanism with security threats and future scopes: Blockchain," *Computer Communications*, vol. 201, pp. 102–115, Mar. 2023, doi: 10.1016/j.comcom.2023.01.018.

[26] G. Ilieva and T. Yankova, "IoT System Selection as a Fuzzy Multi-Criteria Problem," *Sensors*, vol. 22, no. 11, 4110, 2022, doi: 10.3390/s22114110.

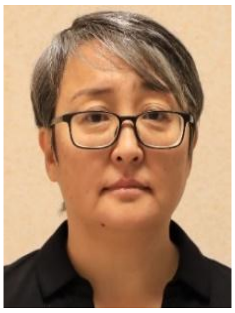
[27] Y. Yamashita, A. Taya, and T. Wada, "Decision-tree-based distributed learning for IoT devices," *Journal of Reliable Intelligent Environments*, vol. 11, no. 2, p. 9, 2025, doi: 10.1007/s40860-025-00249-z.

BIOGRAPHIES OF AUTHORS

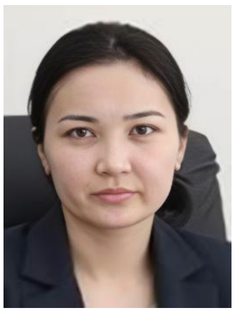
Nurlan Tashatov graduated from the Kirov KSU in Almaty in 1978 with a degree in mathematics. He received his Ph.D. in applied mathematics in 2002. He currently conducts research at the Research IISC of the Gumilyov ENU, and also works as an associate professor at the Department of Information Security, conducts seminars and lectures on coding theory and cryptography. His research interests include the theory of noise-resistant coding for information security. He can be contacted at email: tash.nur@mail.ru.

Ruslan Ospanov is currently working on his Ph.D. thesis in Information Security at Research IISC of the L.N. Gumilyov ENU. His research interests include coding theory, parallel computing, secure outsourcing, cryptographic protocols, blockchain, big data, and the internet of things. He is the author of a number of utility model patents and copyright certificates. He is also the author of the author's online course on cryptography, intended for schoolchildren, students. He is also a co-author of the post-quantum encryption algorithm based on hashing. He researched and developed the hash function, which is new, and also developed new methods for generating substitution tables used in classical cryptography. He can be contacted at email: ospanovrm@gmail.com.

Yerzhan Seitkulov is a Ph.D., professor at the Department of Information Security and a chief research fellow at the Research IISC at L.N. Gumilyov ENU. His research interests include cellular network security, modern cryptographic protocols, post-quantization cryptography, secure outsourcing, speech technologies, voice information protection, and supercomputer technologies. He is also a co-author of a number of utility model and invention patents, as well as copyright certificates for software. Over the past 10 years, he has conducted a number of research projects within the framework of grant projects and programs through the industry ministries of the MSHE of the RK, and others. He can be contacted at email: yerzhan.seitkulov@gmail.com.



Dina Satybaldina graduated from the University with a degree in Physics and Computer Science in the Republic of Kazakhstan. She received her Ph.D. in Physics and Mathematics in 1993 from the Buketov KSU Karaganda, Karaganda, Republic of Kazakhstan. She received her Ph.D. degree in Computer Science from the Al-Farabi KNU, in 2011. Currently, she is the Director of the Research IISC, and also teaches as a professor at the ENU. She is the author of several textbooks, she has published more than 120 articles, more than 15 patents for inventions, as well as author's certificates. Her research interests include cryptography, cellular security, error-correcting coding, supercomputing, secure outsourcing, machine learning, artificial intelligence, and malware analysis. She can be contacted at email: dinasaty@gmail.com.



Banu Yergaliyeva entered the Department of Mathematics and Computer Science at the L.N. Gumilyov ENU in 2000. In addition, she received a second higher education in economics, as well as a master's degree in finance. She has prepared a number of scientific articles, received copyright certificates, and is also a co-author of a patent for a utility model in the field of cellular network security. She is an expert in the field of cellular network security, post-quantum cryptography, distributed computing, cloud computing. Currently, she has completed her Ph.D. studies in the specialty "Information Security Systems". She can be contacted at email: banu.yergaliyeva@gmail.com.