An Improved Cellular Goore Game-based Consensus Protocol for Blockchain

Reyhaneh Ameri*, Mohammad Reza Meybodi

Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran

r.ameri@aut.ac.ir
mmeybodi@aut.ac.ir

Abstract

A blockchain is a distributed network of nodes that records transactions in a digital ledger. To ensure trust in a network with untrusted nodes, blockchain uses consensus algorithms to record transactions in its ledger. Although various consensus protocols have been proposed and implemented for blockchain lately, they still have drawbacks. The last decade has seen a lot of attention and rapid growth in artificial intelligence and blockchain technologies. The performance of blockchain can be improved, and its problems can be solved by incorporating artificial intelligence. The concept of cognitive blockchain is related to AI functionalities into blockchain systems to enhance their utility and capabilities. Cellular Goore Game-based consensus was recently proposed as intelligence consensus in the cognitive blockchain. Although this consensus algorithm improves scalability, fault tolerance, and performance, it has problems, such as high communication complexity. In this paper, we proposed an improved Cellular Goore Game-based consensus protocol, which increases fault tolerance and decreases communication complexity. We also studied theoretically the Cellular Goore Game used as a distributed model in the proposed consensus and proved the convergence of CGG in the proposed protocol in this paper. We evaluated the performance of the proposed protocol by conducting several experiments. Empirical results show that this approach improves fault tolerance and communication complexity.

Keywords: Blockchain, consensus, Cellular Goore Game, Learning Automata, cognitive, fault tolerance;

1. Introduction

Blockchain technology, a decentralized and distributed digital ledger system, has gained significant attention in the health [1]–[3], smart contracts [4]–[6], and Internet of Things [7]–[10] domains due to its potential to disrupt traditional processes. The blockchain technology initially created for Bitcoin [11] has advanced significantly and recently has many uses [12]. Blockchain is a chain of interconnected blocks containing transactional or data records linked via cryptographic hashes and maintained by a node network [13]. Consensus mechanisms ensure that each block is verified before it is added to the chain. The transparency, security, and immutability of data are guaranteed by the decentralized nature of blockchain, which eliminates the need for intermediaries and opens up new opportunities for trust and collaboration in established and emerging industries [14].

While blockchain has the potential to create new internet infrastructure [14], its usability is limited by scalability, environmental cost, security, and privacy concerns. Blockchain's inability to serve as a general platform for a wide range of services and applications is primarily due to scalability issues [15]. Moreover,

* Corresponding Author

the majority of blockchains available today use a significant amount of energy. In pursuing transactional throughput, many new blockchain platforms overlook other crucial performance characteristics like decentralization, security, and latency.

Recently, AI-based blockchain solutions have been presented in various areas, including and performance optimization scalability [16]–[18], security improvement [19]–[25], and prediction model provision on blockchain data [26]–[30]. The optimal design of the blockchain mechanism can be facilitated through AI's efficient data analysis and system behavior prediction. In [31], a cognitive blockchain was proposed to combine intelligent cognition and blockchain to enhance performance and achieve intelligence. A cognitive blockchain is a blockchain that incorporates a cognitive process to enhance performance by making intelligent decisions based on the network's state and using the outcomes to improve future decisions. In the cognitive blockchain, network nodes have a cognitive engine set embedded, which assists them in making intelligent decisions because of the decentralized nature of the network. The cognitive engine may or may not be used by nodes. There are several cognitive and simple nodes on the cognitive blockchain. Cognitive nodes are those that have cognitive engines embedded, whereas simple nodes lack them. Different approaches, such as centralized, semi-centralized, and fully distributed approaches, are available to implement cognitive blockchain based on whether all nodes have permission to use the cognitive engine. Figure 1 depicts an example of a cognitive blockchain network.

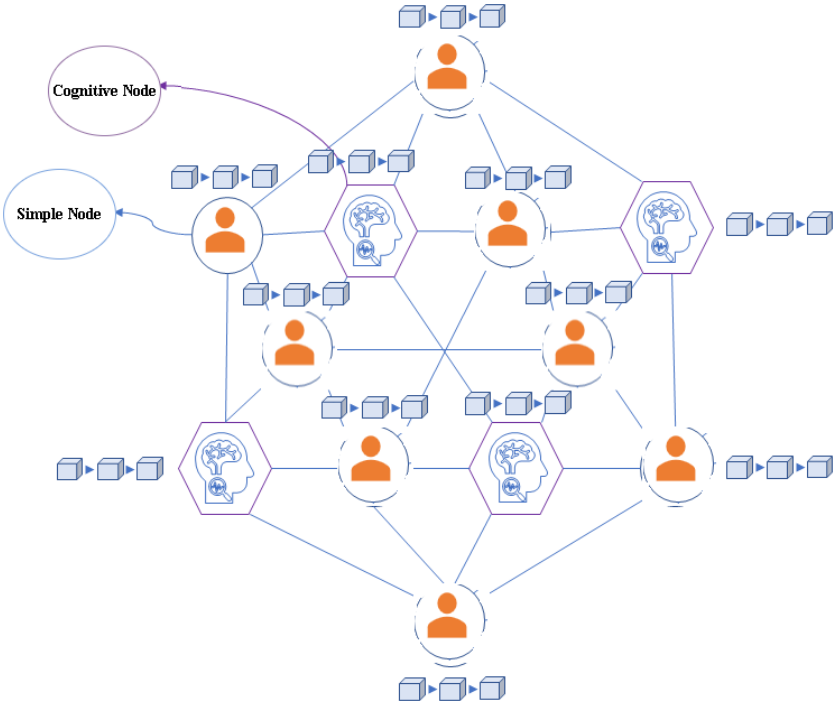


Figure 1: an example of a cognitive blockchain network.

For the stability of the blockchain, it is necessary that all nodes agree on the protocol for updating the ledger's content, and the acceptance of blocks requires a majority agreement among nodes. If faulty and malicious nodes exist in a decentralized distributed network, the only safe way to update is by using the consensus technique. [32] proposes an intelligent Cellular Goore Game-based consensus mechanism for cognitive blockchain in a semi-centralized manner. The embedded cognitive engine and proposed consensus protocol's CGG-based consensus protocol, named CGG-CP, in a partially synchronous timing model enable all consensus nodes (including owner, referees and players) in the blockchain network to reach an agreement. CGG-CP decreases the impact of failure nodes while improving the performance and scalability of consortium blockchain. For CGG convergence and consensus, nodes need to send a considerable number of exchange messages, resulting in high CGG-CP communication complexity. Also, CGG-CP is not resistant to owner failure. This protocol used CGG as a distributed model, and CGG, formed between consensus nodes, should converge to add a block. Despite this, no theoretical study has analyzed the behavior of CGG and proved its convergence. In this paper, we propose an improved consensus protocol to address the problems highlighted in CGG-CP and increase its resilience to faulty referees. We give the following contributions in this paper:

- We introduce a new consensus protocol that can be employed as intelligence consensus in the cognitive blockchain. The presented consensus protocol is an improved version of the Cellular Goore Game-based consensus [32] that reduces communication complexity and improves fault tolerance.
- To the best of our knowledge, we are the first to theoretically study the behavior of the Cellular Goore Game. Also, we prove the convergence of CGG in the proposed protocol in this paper.
- We conducted a qualitative analysis of the proposed consensus protocol. We thoroughly evaluate its performance in our blockchain implementation, considering the defined evaluation metrics like throughput, average block time, average latency, and the average number of communications, and compare it to CGG-CP. Results indicate that the proposed protocol reduces communication complexity and enhances fault tolerance.

This paper is organized as follows: Section 2 provides some background and preliminaries for this paper. The improved CGG-based consensus protocol is proposed in Section 3. In section 4, we study the behavior of the Cellular Goore Game and prove its convergence in the proposed consensus protocol. Section 5 summarizes the simulation results used to assess the proposed approach. Section 6 concludes the paper.

2. Background and preliminaries

In this section, we briefly discuss Learning Automata, the Goore Game, the Cellular Goore Game, and the Consensus Mechanism to prepare the context for the rest of the paper and provide the literature review.

2.1. Learning Automata

Learning automaton as an adaptive decision-making system [33] can improve performance by repeatedly choosing the optimal action from a list of possible options in a random environment. A learning automaton

has a limited number of actions with an uncertain probability of environmental reward. Through repeated system interactions, the goal is to choose the action with the most significant possibility of reward. Iteratively interacting with the environment using a proper learning algorithm can discover the best action [34]. LAs can be divided into two categories: fixed and variable structure.

The variable structure LAs is represented by $\langle \beta. \phi. \alpha . P. G. T \rangle$. Set β comprises input actions, while set ϕ represents internal states and set , α is for outputs. P refers to the state probability vector, G is the output mapping, and T is the learning algorithm. The learning process adjusts the probability vector. The characteristics of β could be used to classify environments as P-, Q-, and S-models. In a P-model environment, success and failure are the two components that form the output. In S-model environments, β lies within [0, 1]; however, it can take a finite number of values in Q-model environments. Assume that α_i is the selected action at step t, which is a sample from distribution P(t). Equation 1 defines the update of probability vector P(t) in a linear learning algorithm for a positive response (β=1), while equation 2 defines it for a negative response (β=0). The two parameters a and b represent the reward and penalty parameters, respectively.. By adjusting the parameter a (b), you can increase (decrease) the action probabilities. The total number of possible actions the LA may do is represented by r. If a=b, it is called the linear reward penalty (L_{RP}), and if a>>b, it is called the linear reward penalty ($L_{R\varepsilon P}$); however, if b=0, it is known as the linear reward inaction (L_{RI}) technique [35].

$$p_i(t+1) = p_i(t) + a\big(1 - p_i(t)\big) \qquad (1)$$

$$p_j(t+1) = p_j(t) - ap_j(t) \ . \forall j \neq i$$

$$p_i(t+1) = (1-b)p_i(t) \qquad (2)$$

$$p_j(t+1) = \frac{b}{r-1} + (1-b)p_j(t). \ \forall j \neq i$$

2.2. The Goore Game

This section explains the Goore Game and LA. Tsetlin [36] introduced it, while Narendra & Thathachar [37] and Thathachar & Arvind [38] provided detailed studies. The characters in The Goore Game are captivating and can be solved through distributed methods without player-to-player communication. Participants can choose between "yes" or "no" options in this cooperative game. The players' chosen voting is evaluated by the referee. The referee contains an unimodal performance evaluation function G. The referee calculates the ratio λ, which is the proportion of players that voted “yes” to all after each player chooses one of their options. Next, the referee randomly gives a dollar to one player with a chance of G(λ) and gives a dollar to each player independently with a chance of 1 - G(λ). Based on their profits and losses, players decide how to vote for the next iteration. There's a correlation between the maximum of G(λ) and the number of players who will eventually answer yes. The existence or total number of participants may go unnoticed by a player

in many cases. Areas where the GG has made critical applications are cooperative, mobile robotics [39], and Quality of Service (QoS) management in wireless sensor networks [40]. Further research was carried out in LA, and numerous families in LA solve the Goore Game successfully. Thathachar et al. [38] proposed the use of Variable Structure Learning Automata (VSLA) [41] as a solution to the Goore Game. Each player's strategies are represented by a Learning Automata (LA) that performs actions. The L_{R-I} learning algorithm [42] is used to update the action probability of each LA.

2.3. The Cellular Goore Game

Cellular Goore Game (CGG), presented in [43], is a model for simple identical elements with local interactions that can optimize one or more criteria. CGG is a network of Goore Games where a subset of nodes act as referees and participate in Goore Games with their neighboring players. Players, like in GG, independently select the best action from two options, considering the penalties and rewards given by the nearby referee. CGG players are unaware of other players and the factors that determine rewards and penalties. In CGG, the neighborhood structure is included, which distinguishes it from GG and is essential for some applications. Several referees can be contained in the proposed model to optimize multiple criteria simultaneously.

Each player in the CGG uses a learning automaton with two actions for selecting its actions, and referees reward adjacent players based on their uni-modal performance criteria. In the first step, the action probability vector of the LA residing in each player cell is used to specify the internal state of each player cell. In the second step, the referee cells determine the reinforcement signal for their nearby cells. Finally, the action probability vector of each LA is updated based on the reinforcement signal received and the internal state of

the cell. Until the desired result is gained, this process will continue. The formal definition of a CGG with learning automata is as follows:

Definition 1. A Cellular Goore Game is a structure CGG = (*N*, *P*, *LA*, *R*, *G*), where

I. P is a subset of cells or nodes that act as the role of players, and N_p is P's cardinality.

II. *R* is a subset of cells or nodes that act as the role of referees, and N_R is R's cardinality. The intersection of sets P and R could be empty or not.

III. *N = (V, E)* is a directed network that determines the structure of CGG where $V = P \cup R$ is the set of nodes which can be either players or referees, and $E = \{(P_i, R_j) \mid P_i \in P \ \& \ R_j \in R\}$ is the set of edges.

IV. $LA = \{LA_1, LA_2, \ldots, LA_{N_P}\}$ is a set of learning automata, where each of them is assigned to one player cell of CGG.

V. $G = \{G_1, G_2, \ldots G_{N_R}\}$ is a set of unimodal performance criteria for the referees, where G_i is the performance criterion for $referee_i$.

The goal of CGG is to maximize the total performance criterion that is the sum of G_i's where G_i is the performance criterion for $referee_i$. $G_i(\lambda)$ is a function of the number of $referee_i$'s neighbours who selected the first action and is optimized when the fraction is exactly λ^*_i. CGG's operation occurs in the following iterations: Each player selects an action from the available options α_1 and α_2 during round k of the CGG based on its probability vector. Let α_i represent the action chosen by LA_i embedded in i-th player. Then, all referees count the proportion, λ_i, of neighbouring players that selected α_1 and reward their players with probability $G_i(\lambda)$ and penalize them with probability 1 - $G_i(\lambda)$. After that, each player adjusts its action

probability vector using the average reward of neighbouring referees (or another predefined policy) and its chosen action α_i. Figure 2 depicts a CGG schematic.

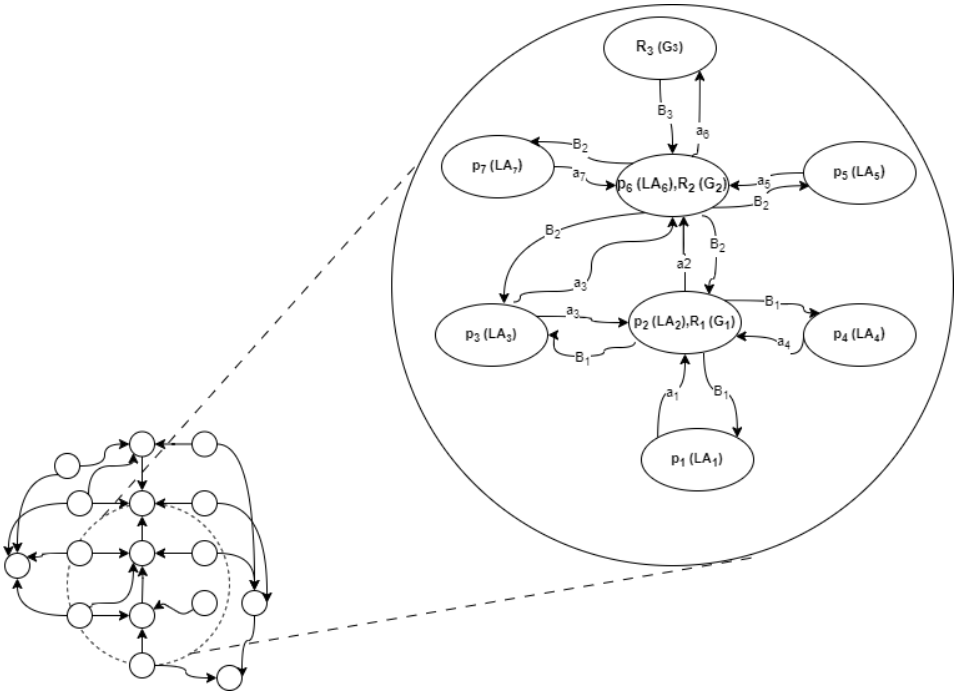


Figure 2 Cellular Goore Game.

2.4. An Overview of Consensus Mechansm

Consensus is a decision-making method where the group's best interest is more important than individual preferences. Consensus mechanisms govern the agreement process between participants and block addition to the chain. Safety, liveness, and fault-tolerance features demonstrate the usefulness and practicality of a consensus method [44], [45]. Safety ensures outputs are valid and nodes produce the same output for the same input. "Liveness" is when each participating node in the consensus mechanism gets the transaction and creates a value. The consensus mechanism should be able to recover from the failure of participating nodes to provide fault tolerance. There are two types of faults in distributed environments for fault tolerance: (1) stopping failure and (2) Byzantine failure [46]. When the hardware or software fails, the node in the consensus mechanism stops responding, resulting in stopping failure. The Byzantine failure model permits faulty processes to exhibit unrestricted behavior.

Proof-based and vote-based are the two classifications of consensus algorithms [47]. By assuming that the node in the network that gives enough proof will be rewarded with a new block added to the chain, the proof-based consensus technique works. In vote-based consensus algorithms, a voting process selects miners. Several proof-based consensus algorithms, such as proof-of-work [11] and vote-based consensus algorithms, such as PBFT [48], have been presented for use in blockchain. Blockchain technology requires consensus methods to ensure the network's security, reliability, and resilience. Each consensus algorithm

has strengths and weaknesses, and the best depends on specific needs and situations. Proof-based consensus algorithms are decentralized and perfect for public blockchains since they don't require individual trust. They are unsuitable for fast transaction processing applications due to low efficiency and probabilistic transaction confirmation. Voting-based consensus algorithms finalize transactions more quickly and efficiently with less energy usage than proof-based algorithms. Conversely, voting-based algorithms necessitate dependable validations and are less secure on open networks. When there are more network nodes in most voting-based algorithms, efficiency declines, and decentralization is restricted by a specific limit.

The BFT algorithm, introduced by Lamport et al. [49], presented an allegory for the challenges of achieving consensus in a decentralized system. Byzantine Fault Tolerance (BFT) refers to a distributed network's capability to achieve consensus despite nodes failing to react or providing inaccurate information. The purpose of BFT-based consensus algorithms, such as PBFT [48], and Tendermint [49], is to address Byzantine fault tolerance and achieve consensus despite the presence of malicious nodes. Most BFT-based consensus algorithms are vulnerable to attacks on the primary node and lack the capability to efficiently eliminate faulty nodes in the blockchain system. Lei et al. [50] proposed a Reputation-Based Byzantine Fault-Tolerance (RBFT) to address these problems by evaluating both a node's contribution and reputation during voting. Malicious behavior will result in a lower reputation and diminished influence in the consensus process for a malfunctioning node. The RBFT algorithm gives priority to nodes with a higher reputation when verifying new blocks. The Weighted Byzantine Fault Tolerance [51] enhanced system throughput and consensus delay by introducing a dynamic weighting mechanism for consensus nodes. By reducing the influence of malicious nodes, this enhances the security of the blockchain system and decreases the probability of malicious behavior. [52] introduced tPBFT, an enhanced algorithm for high-frequency trading in consortium chains. This algorithm utilizes a trust-based approach to dynamically adjust the consensus node list. Reduced network node interaction overhead is achieved by tPBFT by simplifying the pre-prepare step and verifying the hashed transaction list at the reply stage. RB-BFT [53] is an improved Byzantine fault-tolerant consensus that proposed a node reputation model to enhance PBFT's primary node reliability and reduce its communication complexity. To increase system security, the suggested approach statistically assesses node behavior in consensus to select high-trust nodes as primary nodes.

Recent years have seen the proposal of AI solutions for challenges in blockchain consensus protocols, which we will discuss below. Salimitari et al. [24] introduced a novel framework for secure and reliable consensus in IoT networks based on blockchain with the help of machine learning. The proposed framework in this paper outlines a two-stage consensus process that utilizes an outlier detection algorithm to confirm data consistency, remove suspicious data, and use PBFT as the consensus algorithm. The majority of research in this context relates to increasing the performance of BFT algorithms by using AI, and this is achieved through two methods: optimizing architecture and reducing the number of nodes involved in consensus using trusted institutions [54], [55]. Bogdi et al. [18] introduced a decision-theoretic online learning approach for selecting a subset of nodes to participate in BFT-based consensus methods. By determining reputation value for nodes condidated in consensus committee participation, the presented

model can successfully choose consensus committee nodes with a higher reputation. For evaluating the validity of nodes, criteria including response type, response type, and response time have been employed.

Liu et al. [16] presented a method that selects block producers and consensus algorithms such as Quorum [56], Zyzzyva [57], and PBFT [58] and adjusts the block size and interval to enhance the transactional throughput of blockchain-based Industrial Internet of Things using DRL methodology. The performance of blockchain systems is quantified in the paper based on scalability, decentralization, latency, and security. Through reinforcement learning, the Credit Reinforcement Byzantine Fault Tolerance Consensus (CRBFT) algorithm, as suggested in [59], assigned the credit attribute to the node and adapts it over time. The proposed technique can increase consensus network security, minimize consensus delay, and maximize energy savings by automatically detecting malicious nodes in the consensus network.

[60] utilized reinforcement learning to adapt the number of participating nodes based on network conditions and behaviour, improving scalability in PBFT. The protocol's security and liveness remain unaffected, even though the number of participating nodes has decreased. Goh et al. [61] proposed a trust-based delegated consensus blockchain framework that utilizes DRL and delegated consensus techniques to enhance security performance and blockchain throughput in IoT networks. By evaluating node reliability, the proposed schema selects reliable nodes for consensus, reducing malicious activity.

Most studies aimed at addressing blockchain consensus protocol issues concentrate on optimizing and scaling for a specific use case with customized solutions. Additionally, most solutions suggested have only aimed at enhancing scalability or performance, which might hurt other aspects of blockchain such as security and decentralization. A more centralized structure, for example, can result from the majority of reputation point-based proposed methods. Furthermore, the specifics of how to apply their method in a distributed blockchain environment have not been covered by the suggested solutions in this context.

Ameri et al. [32] proposed CGG-CP, the Cellular Goore Game-based Consensus Protocol, as a new intelligence consensus protocol to solve the mentioned issues. This consensus method promotes decentralization and scalability while reducing the impact of failed nodes in consortium blockchain systems. In addition, the cognitive blockchain framework provides all the necessary details for executing CGG-CP in a distributed environment. The four roles defined in the protocol are referee, player, owner, and ordinary role. Referee nodes are nodes that other nodes trust more. If a node has a referee listed as its trusted node, it will be considered a player role. One of the referee nodes will randomly assume the owner's role. Ordinary nodes are nodes that are unable to act as players or referees. To verify a new block's validity, created by the owner, players, and referees, create the CGG. The owner employed the result of CGG according to a predetermined policy to either approve or reject the block. High CGG-CP communication complexity occurs due to nodes sending a considerable number of exchange messages for CGG convergence and consensus. Moreover, CGG-CP is not resistant to owner failure. We will address the mentioned issues by extending the

CGG-CP and presenting an improved Cellular Goore Game consensus protocol, described in the following sections.

3. An Improved CGG based Consensus Protocol

In this section, we present an Improved Cellular Goore Game-based Consensus Protocol (ICGG-CP) as a new voting-based consensus mechanism for the consortium blockchain. ICGG-CP enhances the CGG-CP consensus algorithm introduced in [32], reducing communication complexity and improving fault tolerance, as the name implies. The proposed Consensus Protocol can be applied as intelligence consensus in the cognitive blockchain, as proposed in [31], and allows each cognitive peer to reach a consensus on the current state of the distributed ledger using a partially synchronous timing model. CGG-CP communication complexity is high because nodes have to send a significant number of exchange messages in order to converge CGG and achieve consensus. On the other side, CGG-CP is not resilient to owner failure and assumes that the owner node will not fail. Therefore, this paper proposes new consensus algorithms that improve fault tolerance and communication complexity by modifying CGG-CP as follows:

- A new feature has been added to ICGG-CP to reduce the cost of communication for player nodes. When a player's action probability converges to an action in executing the CGG, they can inform their referees through a special message and avoid sending repetitive actions if they have trust in the referee. These players are certainly rewarded by adjacent referees for updating their probability vector; however, they have this permission to not send their action until their probability vector is outside the convergence range for that action.
- CGG result summary and credit calculated for consensus nodes, includes referee and player nodes, are saved in the block header of blockchain. When a block is rejected through consensus, its transactions are deleted, but it with its header and reward-related transactions are added to the blockchain. There is a possibility that the nodes' credit will increase only as a reward, and the reward coin won't be considered, in which case a rejected block with an empty body will be added to the blockchain. The aim of this difference in ICGG-CP as opposed to CGG-CP is to prevent any changes to the credits of nodes and to preserve the result of the executed CGG for rejected blocks. Furthermore, we'll provide more details on how the referees and owner's behavior will be traceable with this difference.
- Referees in ICGG-CP are evaluated based on their credibility, which is determined by factors such as the number of players adjacent to them and their average response time. The owner node is selected randomly from referees, with probability based on their credibility. Similar to CGG-CP, in ICGG-CP, the block must be created by the owner, who then decides whether to approve or reject it based on the CGG's results, distributes the reward to other nodes, and appends the block to the blockchain. The referees in ICGG-CP have the ability to check the owner's decisions, which helps to prevent abuse and increase the consensus protocol's resistance to owner faults. Also, during the

owner selection process, the secondary owner node that acts as the owner when the owner is absent is chosen from the referees.

- ICGG-CP mandates that every player use the weighted average based on the credibility of their neighbouring referees as their predefined policy. The ICGG-CP has proposed this policy to increase the influence of nodes with higher credibility in confirming or disconfirming blocks. Because the probability of a node being Byzantinely faulty has an inverse relationship with its credibility, the fault tolerance of referee nodes has been improved in ICGG-CP in this way.
- In ICGG-CP, to identify Byzantine referees and increase fault tolerance, the ICGG-CP includes a mechanism for players to report referee-faulty behaviour. Players are able to check the behaviour of adjacent referees, such as verifying the correctness of the results that were sent to others about them.

The steps in the improved CGG-based Consensus Protocol in block validation are depicted in Figure 3. These ICGG-CP block validation steps are discussed in more detail below:

- Creating the block: Using the queue of unconfirmed transactions received from nodes, the owner creates the block.
- Broadcasting the block: The block is broadcasted by the owner to the referees, who subsequently broadcast it to the neighboring players.
- Validating the block: Referees and players verify the block. Similar to CGG-CP, the referee's performance criterion in CGG relies on its decision about block validation. Also, the players' action probability will be determined by their opinion of whether the block should be accepted or rejected.
- Executing the CGG: The CGG is executed by players and referees, like CGG-CP. Every referee plays a GG with their adjacent players. Referees declare the end of their GG to neighboring players based on predetermined principles such as convergence and maximum executive rounds. If a player trusts their referee, they can send a unique message that informs them when their action probability converges to an action in performing the CGG and avoid repetitive actions. Sending a liveness message to nearby referees at specific intervals is necessary for players whose automaton has converged on a specific action. This will help the referee distinguish between these players and players who didn't send their action due to a stop failure.
- Sending the model execution's results: It is the referee's responsibility to provide the owner, other referees, and their adjacent players with specific data structures with statistical information about their game with their neighbors, such as their selected criterion function, adjacent players' chosen actions action and convergence value. In ICGG-CP, through available data, other referees can supervise the owner's work, and players can supervise their nearby referees, which is not possible in CGG-CP.
- Determining the consensus result: Based on predefined decision rules, the owner approves or disapproves the block after getting data from referees within a specified period. If the block is rejected, the owner must remove the transactions in the body of the block.
- Calculating the reward and credits for consensus nodes: Based on a pre-defined incentive mechanism, the owner calculates the reward and punishment for the players and referees. If required, transactions are included in the block for this purpose. Then, using a predetermined algorithm that

considers factors like average response time and GG convergence speed, the owner calculates the credits of consensus nodes.

- Appending block to blockchain: The owner adds the block to the blockchain after completing it with the necessary metadata from previous steps, such as credits for consensus nodes.

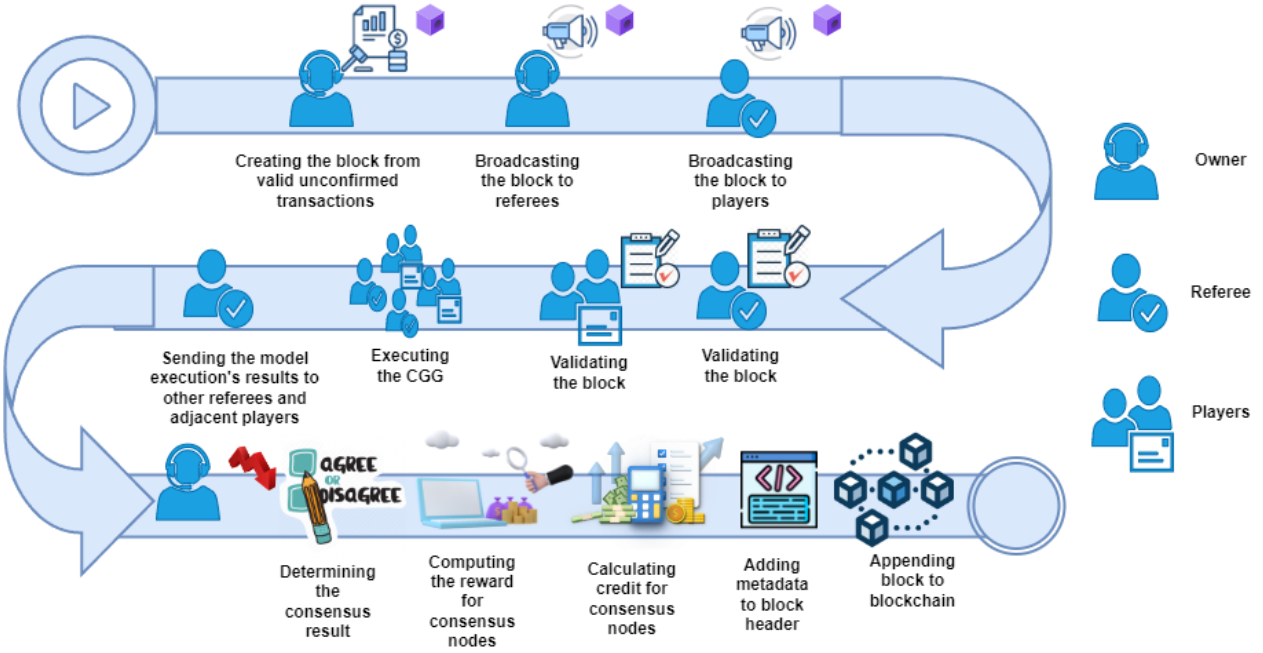


Figure 3 The steps of the improved CGG-based Consensus Protocol.

The cognitive nodes—referees and players—must perform certain operations in the ICGG-CP process. But their embedded cognitive engine handles most of them automatically, including broadcasting the block, executing the CGG, sending the model execution's results, calculating the reward and credits for consensus nodes, and appending blocks to the blockchain. It's sufficient that their connection to the network remains stable. The use of cognitive engines can minimize message manipulation for consensus tasks, thereby improving security and reducing node participation costs.

In ICGG-CP, each player calculates its reward by applying a weighted average of the rewards received from neighboring referees based on their credibility. Figure 4 illustrates how a player with several neighboring referees computes reinforcement signals at round k of executing the CGG step. In Figure 4, player i's selected action is $\alpha_i(k)$, while referee m's credit is C_m, and $\beta_i^m(k)$ is the reinforcement signal given by referee m to player i at round k. Players extract the credit of each referee from the header of the distributed block they are validating.

Voting-based consensus algorithms are limited in achieving prompt consensus in networks with a significant number of peers, but ICGG-CP, similar to CGG-CP, overcomes these issues. ICGG-CP doesn't require all consensus nodes to send their vote to every other node, in contrast to some consensus algorithms based on voting, such as pbft. Instead, the proposed algorithm involves the nodes with referee, player, and

owner roles as validating nodes in consensus, and the nodes are connected in a neighborhood structure based on the CGG network graph. To reach a consensus, referees exchange messages with neighboring players and the owner, while players also communicate with neighboring referees. Additionally, the owner must communicate with all referees to achieve consensus in accordance with the proposed protocol. The ICGG-CP ensures network scalability and decentralized consensus with high transaction speed. The proposed protocol has the potential to serve as a consensus algorithm in consortium and private blockchains. Consortium blockchains benefit from the CGG-CP consensus protocol, which utilizes organized nodes and defined roles for achieving consensus. Creating an underlying network and organizing nodes allows the suggested consensus protocol to be adapted for private blockchains. Various industries, including healthcare, insurance, and corporate governance, can benefit from the proposed consensus algorithm through intelligent and automated consensus among parties, leading to a facilitation of tasks and a reduction in human error and fraud.

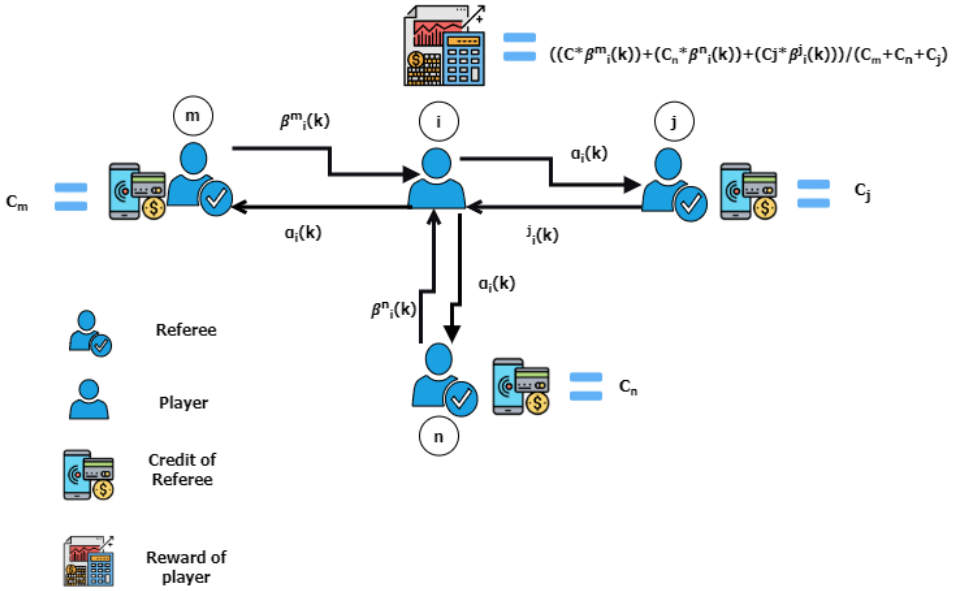


Figure 4 Example of reinforcement signal computation for a player with multiple neighbouring referees.

The proposed consensus protocol is robust against various types of failures in distributed environments, including stop, Byzantine, and unavailability failures. The cognitive engine embedded in the nodes is one of the fault tolerance mechanisms, and it cannot be manipulated via a digital signature in the proposed consensus. ICGG-CP also has a fault reporting and punishment mechanism for fault tolerance that punishes faulty nodes if approved by others. If the referee detects an owner's fault, it can broadcast a predefined system message to change the owner to another referee. If more than half of the referees agree, the secondary owner will replace the owner. If a player detects a fault from the adjacent referee, they can use a specific message to inform other adjacent referees. If the request is considered valid by the adjacent referee, the referee will broadcast a predetermined message to the others. If more than half of them accept, the faulty referee's credibility will be reduced. In ICGG-CP, the fault tolerance of referee nodes has been enhanced

with using the weighted average by players. The fault tolerance level of CGG-CP relies on its decision rules and predetermined policies, such as block validation rules. The proposed method's fault tolerance has also been assessed through experimental experiments and is reported in the numerical results section. In the following, we discuss how the proposed protocol faces these types of failures that may occur in consensus.

Ordinary nodes have nothing to do with block validation; hence, their failure won't impact the consensus process. Players and referees have the potential to exhibit a Byzantine failure through a false decision on block approval in the block validation of ICGG-CP. The proposed protocol decreases the incentive for consensus nodes to engage in byzantine actions by utilizing known identities and a system of rewards and punishments. If the Byzantine-faulty referee makes an incorrect evaluation of the block, enough non-faulty players nearby can stop its GG from reaching the desired amount. Also, when a player wrongly validates the block, adjacent referees with a sufficient number of non-faulty adjacent players will experience slower convergence, leading to a decrease in rewards for this player. If a node halts during execution in the proposed schema, it can result in a stopping failure. In the ICGG-CP, failure detection mechanisms that are embedded in referees' cognitive engines improve fault tolerance in the event of stop-failures by players. If the referee fails to stop, the consensus process will remain unaffected, and neither the referee nor the nearby players will receive a reward in that round. If the referee is not available to validate a block, the option to participate with the players might be selected automatically during the criterion function determination step. In this scenario, the referee will be punished, while the players will be granted significant incentives. Also, an equal probability vector serves as the player's initial probability vector when a player is unable to validate a block.

4. Convergence results

In this section, we prove the convergence results of the CGG used in ICGG-CP proposed in this paper. For this purpose, we first outline requirement definitions, such as configuration, deterministic configuration, global behaviour, and compatible configuration of CGG, in this section. Then, we approximate the proposed algorithm with an ordinary differential equation (ODE) by applying weak convergence theorems. Next, we prove that the proposed algorithm and the resulting ODE converge to the solution of the optimization problem.

4.1. Basic Definitions and Notations

In order to analyze the CGG later in the paper, we present the problem formulation, definitions, and preliminary results in this section. The CGG involves N_P players and N_R referees in a directed network N. The definitions related to the CGG network must be provided in the following section for problem formulation.

Definition 2. The set of players adjacent to referee j in the CGG network is denoted by $NP_j = \{ P_i | \, P_i \in P, (P_i, R_j) \in E\}$, where P and E represents players and edges, respectively.

Definition 3. The set of referees adjacent to player i in the CGG network is denoted by $NR_i = \{ R_j | \, R_j \in R, (P_i, R_j) \in E\}$, where R and E represents referees and edges, respectively.

Definition 4. The set of players adjacent to player i in the CGG network is denoted by $NPP_i = \{ P_k | R_j \in R, (P_i, R_j) \in E, P_k \in P, (P_k, R_j) \in E\}$, where R, P, and E are referees, players, and edges, respectively.

Definition 5. The N^i is a subgraph of the CGG network that includes the referee nodes adjacent to player i (NR_i) and their adjacent players (NPP_i), as equation bellow:

$$N^i = (V^i, E^i). \quad (3)$$
$$V^i = NR_i \cup NPP_i$$
$$E^i = \{(P_i, R_j) \mid P_i, R_j \in V^i, (P_i, R_j) \in E\}$$

Definition 6. Referees in subgraph $N' = (\acute{V}, \acute{E})$ of the CGG network are denoted by $R_{\acute{N}} = \{ R_j | R_j \in R, R_j \in \acute{V}\}$, while players in subgraph $\acute{N}$ are represented by $P_{\acute{N}} = \{ P_i | P_i \in P, P_i \in \acute{V}\}$.

Definition 7. A connected subgraph of a graph to which no vertex can be added and which remains connected is defined as a maximally connected component [62].

Each player i (i = 1, 2,..., N_P) selects an action $\alpha_i(k)$ from two available actions α_1 and α_2, at instant k. The LA approach for solving CGG involves representing each player as LA and assigning them an action probability for action selection. At instant k, the player i has a probability of $x_i(k)$ to choose an initial action and (1-$x_i(k)$) probability to select the second action. The action vector and action probability vector are:

$$\alpha_i(k) \in \{\alpha_1, \alpha_2\}; i = 1,2 \dots, N_P. \quad (4)$$
$$\underline{x}(k) = [x_1(k), x_2(k), \dots, x_{N_P}(k)]$$
$$\underline{\alpha}(k) = [\alpha_1(k), \alpha_2(k), \dots \dots, \alpha_{N_P}(k)].$$

$n_1^j(k)$ represents the number of adjacent players with referee j that selected the initial action at k and $\lambda_j(k)$ denotes the ratio of that number to all adjacent players with referee j. Every Referee j (i = 1, 2,..., N_R) provides neighboring players with a reinforcement signal $\beta^j(k)$ based on G_j ($G_j : \lambda_j(k) \rightarrow \beta^j(k)$), and $\lambda_j(k)$. $\underline{G}(k)$ indicates the vector of the performance criteria values for the referees at iteration k. The reinforcement signal $\beta^j(k)$ is assumed to be non-negative and bounded between 0 and 1. The identity of the players is irrelevant to the environmental payoff; only the number of adjacent players choosing the first action matters. Thus, all combinations that yield the appropriate number λ_j^* will maximize G_j.

$$n_1^j(k) = \sum_{i \in NP_j} I\{\alpha_i(k) = \alpha_1\} \quad (5)$$
$$\lambda_j(k) = \frac{n_1^j(k)}{|NP_j|}$$
$$\underline{n_1}(k) = [n_1^1(k), n_1^2(k), \dots, n_1^{N_R}(k)]$$
$$\underline{G}(k) = [G_1(\lambda_1(k)), G_2(\lambda_2(k)), \dots, G_{N_R}(\lambda_{N_R}(k))]$$

The CGG's objective is to maximize $E[\sum_{j=1}^{N_R} \beta^j(k)]$ by collectively selecting the optimal number of first actions by players. A player may not be aware of other players or how many players are involved in the most

applications. Once a player chooses their action, they only need to know the payoff. Once a player selects their action, they only need to receive the payoff from adjacent referees, which they update their action probability vector based on. To calculate rewards based on adjacent referees' rewards, CGG players follow a predetermined policy. For instance, the policy function could be the mean or weighted mean of rewards attained. Choosing appropriate policy functions is crucial for the performance and convergence of CGG. To study the convergence behaviour, it is necessary to have the definitions related to the configuration of CGG that follow.

Definition 8. $\underline{x}(k) = [x_1(k), x_2(k), \dots, x_{N_P}(k)]$ denotes a configuration of CGG at step k, where $x_i(k)$ is the probability of selecting the first action in the action probability vector of the LA_i embedded in player i.

Definition 9. If the action probability vector of each learning automaton is a unit vector, the configuration $\underline{x}$ is known as deterministic. Therefore, the collections of deterministic configurations, $\mathcal{X}^*$, and probabilistic configurations, $\mathcal{X}$, in CGG are as follows:

$$\mathcal{X}^* = \{\underline{x} \mid \underline{x} = (x'_1, x'_2, \dots, x'_{N_P}), \forall i: x_i = 0 \text{ or } 1\} \quad (6)$$

$$\mathcal{X} = \{\underline{x} \mid \underline{x} = (x'_1, x'_2, \dots, x'_{N_P}), \forall i: 0 \le x_i \le 1\}$$

Definition 10. The dynamics of a CGG can be described by a mapping $\mathcal{G}: \mathcal{X} \to \mathcal{X}$, called its global behaviour. A sequence of configurations $\{\underline{x}(k)\}_{k\ge 0}$ is formed during the evolution of CGG from an initial configuration $\underline{x}(0) \in \mathcal{X}$, where $\underline{x}(k+1) = \mathcal{G}(\underline{x}(k))$.

Definition 11. In configuration $\underline{x} \in \mathcal{X}$, equation 7 defines the average reward for action r of LA_i, $d_{ir}(\underline{x})$, where $\underline{\beta_i^{NR_i}}$ is the reinforcement signal vector of adjacent referees of player i and $policyFunc^i$ is a policy function of player i. Also, equation 8 described the average reward for the LA_i , $D_i(\underline{x})$.

$$d_{ir}(\underline{x}) = \mathrm{E}[\beta_i \mid \underline{x}, \alpha_i = r] \quad (7)$$

$$\beta_i = policyFunc^i(\underline{\beta_i^{NR_i}})$$

$$D_i(\underline{x}) = \mathrm{E}[\beta_i \mid \underline{x}] = d_{i1}(\underline{x})x_i + d_{i2}(\underline{x})(1 - x_i) \quad (8)$$

Definition 12. The sum of average rewards for all LA in CGG is the total average reward for CGG at configuration $\underline{x} \in \mathcal{X}$, as defined in equation bellow.

$$D(\underline{x}) = \sum_{i \in P} D_i(\underline{x}) \quad (9)$$

Definition 13. A configuration $\underline{x} \in \mathcal{X}$ is considered compatible if equation 10 holds for all configurations $\underline{y} \in \mathcal{X}$ and all players. If the following inequalities hold strictly, then the configuration $\underline{x}$ is considered fully compatible.

$$d_{i1}(\underline{x})x_i + d_{i2}(\underline{x})(1 - x_i) \ge d_{i1}(\underline{x})y_i + d_{i2}(\underline{x})(1 - y_i) \quad (10)$$

Lemma 1: Configuration $\underline{x} \in \mathcal{X}$ is compatible if and only if $d_{ir}(\underline{x}) \leq D_i(\underline{x})$ for all i and r.

Proof: If $\underline{x} \in \mathcal{X}$ is a compatible configuration, then $d_{i1}(\underline{x})x_i + d_{i2}(\underline{x})(1 - x_i) \geq d_{i1}(\underline{x})y_i + d_{i2}(\underline{x})(1 - y_i)$ for configuration $\underline{y} \in \mathcal{X}$ that contains $y_i = 1$ for fixed i ($1 \leq i \leq N_P$), obtained from equation 8. Then we have $d_{ir}(\underline{x}) \leq D_i(\underline{x})$.

Conversely, let us assume $d_{ir}(\underline{x}) \leq D_i(\underline{x})$; $\forall$i = 1,2,.., N_P and $\forall r \in \{\alpha_1, \alpha_2\}$ but $\underline{x}$ not compatible. There is a compatible configuration $\underline{y}$ where, assuming configuration $\underline{x}$ is not compatible, $d_{i1}(\underline{x})x_i + d_{i2}(\underline{x})(1 - x_i) < d_{i1}(\underline{x})y_i + d_{i2}(\underline{x})(1 - y_i)$ holds for at least one player i. Considering the assumption that is valid for the configuration $\underline{x}$ ($d_{ir}(\underline{x}) \leq D_i(\underline{x})$), replacing $d_{i1}(\underline{x})$ and $d_{i2}(\underline{x})$ with $D_i(\underline{x})$ should meet the following equation:

$$d_{i1}(\underline{x})x_i + d_{i2}(\underline{x})(1 - x_i) < D_i(\underline{x})y_i + D_i(\underline{x})(1 - y_i) \rightarrow D_i(\underline{x}) < D_i(\underline{x}) \quad (11)$$

The contradiction above completes the proof of the lemma.

Lemma 2: Assume $\underline{x} \in \mathcal{X}$ is a compatible configuration. Therefore, for every player i that satisfy $x_i > 0$ and $r \in \{\alpha_1, \alpha_2\}$, the following equation holds true:

$$d_{ir}(\underline{x}) = D_i(\underline{x}) \quad (12)$$

Proof: According to Lemma 1, we have $d_{ir}(\underline{x}) \leq D_i(\underline{x})$. Assume there exists an action $r \in \{\alpha_1, \alpha_2\}$ by player j where $d_{jr}(\underline{x}) < D_j(\underline{x})$. The following equation can be obtained from this inequality and Equation 8.

$$D_i(\underline{x}) = \sum_{r=1}^{2} d_{ir}(\underline{x})x_{ir} = \sum_{\substack{r=1 \\ x_{ir}>0}}^{2} d_{ir}(\underline{x})x_{ir} < D_i(\mathrm{x}) \sum_{\substack{r=1 \\ x_{ir}>0}}^{2} x_{ir} \quad (13)$$

$$= D_i(\mathrm{x}), where\ x_{i2} = 1 - x_i$$

The proof of the lemma is completed by the contradiction above. Figure 5 provides an overview of basic definitions and notations described in this section that are used in the following sections.

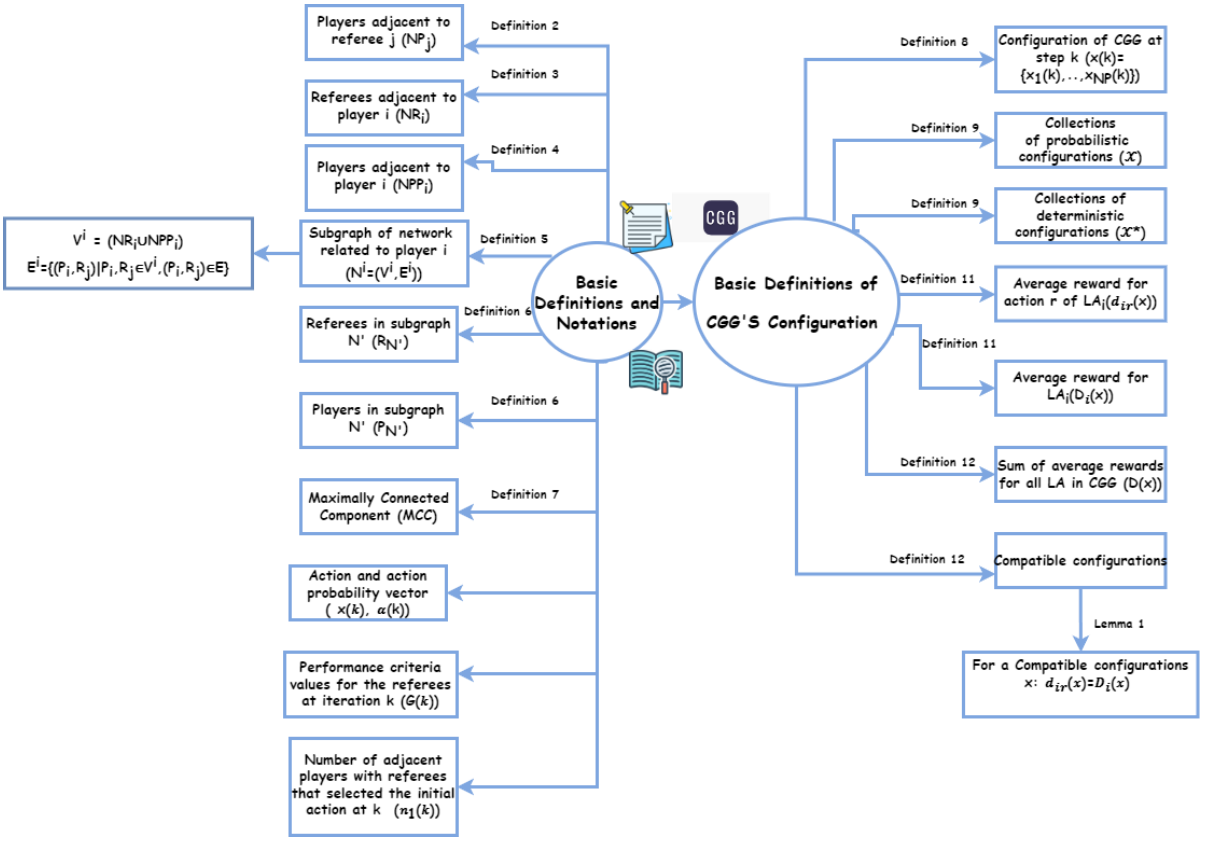


Figure 5: An overview of basic definitions and notations.

4.2. Behaviour of Cellular Goore Game

In this section, we study CGG, where the L_{R-I} learning algorithm is utilized by all learning automata. The following assumptions are used.

Assumption 1: $G_i: [0,1] \rightarrow [0,1]; \forall i = 1,2 \dots, N_P$ is continuous and unimodal.

Assumption 2: The reinforcement signal $\beta^j(k)$ is assumed to be non-negative and bounded between 0 and 1.

The evolution of process $\{\underline{x}(k)\}_{k\geq 0}$, which follows the L_{R-I} learning algorithm, is Markovian and can be explained by this difference equation [37]:

$$\underline{x}(k+1) = \underline{x}(k) + \underline{b}\,\underline{g}(\underline{x}(k), \underline{\beta}(k)), \qquad (14)$$

In equation 14, $\underline{\beta}(k)$ is comprised of components $\underline{\beta}_{iy}(k)$ (for $1 \leq i \leq N_P$ and $1 \leq y \leq |NR_i|$), dependent on $\underline{x}(k)$. Also, $\underline{g}$ is the learning algorithm, and $\underline{b}$ is a $N_P \times N_P$ diagonal matrix with $b_{ii} = b_i$, where b_i is the LA_i learning parameter. According to the type of learning algorithm used, the algorithm for updating the

probability vector of each player's actions can be written as follows in equation 15. Also, $\underline{\Delta x_i}(k)$ is defined in equation 16.

$$x_i(k+1) = \begin{cases} x_i(k) + b\beta(k)(1 - x_i(k)) \text{ if } \alpha_i(k) = \alpha_1 \\ x_i(k) - b\beta(k)x_i(k) \text{ else.} \end{cases} \quad (15)$$

$$\Delta x_i(k) = E[x_i(k+1) - x_i(k) \mid x(k)]$$
$$\underline{\Delta x} = [\Delta x_1, \Delta x_2, \dots, \Delta x_{N_P}] \quad (16)$$

Lemma 3: The Markovian process $\{\underline{x}(k)\}_{k\geq 0}$, which is defined in equation 14, is ergodic.

Proof: We can prove lemma 3 by demonstrating the following two properties in the Markovian process described by equation 14.

1. $\{\underline{x}(k)\}_{k\geq 0}$ has no absorbing states, as no $\underline{x}(k)$ satisfies $\underline{x}(k+1) = \underline{x}(k)$.
2. $\{\underline{x}(k)\}_{k\geq 0}$ is strictly distance diminishing. In Appendix A, we will prove that this Markovian process satisfies Norman's definition [63] of strictly distance diminishing processes.

Also, $\underline{x}^c$ and $\underline{y}^c$ denote $\underline{x}(k+c)$ when $\underline{x}(k) = \underline{x}$, and $\underline{x}(k) = \underline{y}$, respectively. Regardless of initial configurations $\underline{x}$ and $\underline{y}$, $\underline{x}^c \to \underline{y}^c$ as $c \to \infty$. We prove the claim in Appendix a. The ergodicity of the Markovian process $\{\underline{x}(k)\}_{k\geq 0}$ can be concluded by considering the two properties above and the results from this corollary.

Let's assume that referees $m_1, \dots, m_{|NR_i|}$ are neighbors of player i and $\underline{n_1^{NR_i}(k)}$ represents the vector of the the number of adjacent players with theses referees that selected the initial action at k. The conditional expectation of the reward signals received by player i is calculated as follows:

$$\mathrm{E}\left[\beta_i \middle| \underline{n_1}(k)\right] = \mathrm{E}\left[\beta_i \middle| n_1^{m_1}(k), .., n_1^{m_{|NR_i|}}(k)\right] \quad (17)$$
$$= PolicyFunc^i\left(G_{m_1}\left(\frac{n_1^{m_1}(k)}{|NP_{m_1}|}\right), \dots, G_{m_{|NR_i|}}\left(\frac{n_1^{m_{|NR_i|}}(k)}{|NP_{m_{|NR_i|}}|}\right)\right)$$
$$= \mathcal{G}^i\left(n_1^{m_1}(k), .., n_1^{m_{|NR_i|}}(k)\right) = \mathcal{G}^i\left(\underline{n_1^{NR_i}(k)}\right);$$

In the sequel, $\mathcal{G}^i\left(\underline{n_1^{NR_i}(k)}\right)$ is used to represent $PolicyFunc^i\left(G_{m_1}\left(\frac{n_1^{m_1}(k)}{|NP_{m_1}|}\right), \dots, G_{m_{|NR_i|}}\left(\frac{n_1^{m_{|NR_i|}}(k)}{|NP_{m_{|NR_i|}}|}\right)\right)$ for convenience. These definitions in the equation below must be considered to continue with the CGG analysis.

$$S(N) = \{1,2, \dots, N\} \quad (18)$$
$$S(N,i) = \{1,2, \dots, i-1, i+1, \dots, N\}$$
$$\mathcal{P}(N) = \{W : W \subseteq S(N)\}$$
$$\mathcal{P}(N,i) = \{W : W \subseteq S(N,i)\}$$

The equation below depicts the probability that the number of nearby players of player i's adjacent referees who chose the first action is equal to $[l_{m_1}, \dots, l_{m_{|NR_i|}}]$, given that player i has selected their first and second actions.

$$\Pr\left\{n_1^{m_1} = l_{m_1}, \dots, n_1^{m_{|NR_i|}} = l_{m_{|NR_i|}} \mid \underline{x}, \alpha_i = \alpha_1\right\} = \left(\prod_{N_u^i \in MCC(N^i)} \sum_{W \in \mathcal{P}\left(p_{N_u^i}\right): \forall j \in R_{N_u^i}, |W_j| = l_j - 1} \left(\prod_{m \in W} x_m \prod_{n \in S(p_{N_u^i}) - W} (1 - x_n)\right)\right) \tag{19}$$

$$\Pr\left\{n_1^{m_1} = l_{m_1}, \dots, n_1^{m_{|NR_i|}} = l_{m_{|NR_i|}} \mid \underline{x}, \alpha_i = \alpha_2\right\} = \left(\prod_{N_u^i \in MCC(N^i)} \sum_{W \in \mathcal{P}\left(p_{N_u^i}\right): \forall j \in R_{N_u^i}, |W_j| = l_j} \left(\prod_{m \in W} x_m \prod_{n \in S(p_{N_u^i}) - W} (1 - x_n)\right)\right)$$

Here, $MCC(N^i)$ is the maximally connected component set of N^i. Also, the event of players in i selecting the initial action is denoted by W, while the neighbouring players of the jth referee in $R_{N_u^i}$ who choose the first action are represented by W_j. Δx_i can be calculated using the $L_{R\text{-}I}$ technique, as shown in equation 20, and $f_i(x)$ is as determined in equation 21.

$$\Delta x_i = b_i x_i (1 - x_i) E\left[\beta_i \mid \alpha_i = \alpha_1, \underline{x}\right] - b_i x_i (1 - x_i) E\left[\beta_i \mid \alpha_i = \alpha_2, \underline{x}\right] = b_i x_i (1 - x_i) f_i(\underline{x}) \tag{20}$$

$$f_i(\underline{x}) = \sum_{l_{m_1}=0}^{NP_{m_1}-1} \dots \sum_{l_{m_{|NR_i|}}=0}^{NP_{m_{|NR_i|}}-1} \left(\mathcal{G}^i\left(l_{m_1} + 1, .., l_{m_{|NR_i|}} + 1\right) - \mathcal{G}^i\left(l_{m_1}, .., l_{m_{|NR_i|}}\right)\right) \left(\prod_{N_u^i \in MCC(N^i)} \sum_{W \in \mathcal{P}\left(p_{N_u^i}\right): \forall j \in R_{N_u^i}, |W_j| = l_j} \left(\prod_{m \in W} x_m \prod_{n \in S\left(p_{N_u^i}\right) - W} (1 - x_n)\right)\right) \tag{21}$$

The $L_{R\text{-}I}$ approach can be used to calculate Δx_i, as indicated in the equation below.

$$\begin{aligned} \Delta x_i &= b_i x_i (1 - x_i) E\left[\beta_i \mid \alpha_i = \alpha_1, \underline{x}\right] - b_i x_i (1 - x_i) E\left[\beta_i \mid \alpha_i = \alpha_2, \underline{x}\right] = b_i x_i (1 - x_i)\left[d_{i1}(\underline{x}) - d_{i2}(\underline{x})\right] \\ &= b_i x_i \left(d_{i1}(\underline{x}) - d_{i2}(\underline{x}) - x_i d_{i1}(\underline{x}) + x_i d_{i2}(\underline{x})\right) \\ &= b_i x_i \left(d_{i1}(\underline{x}) - \left(x_i d_{i1}(\underline{x}) + (1 - x_i) d_{i2}(\underline{x})\right)\right) = b_i x_i \left(d_{i1}(\underline{x}) - D_i(\underline{x})\right) \end{aligned} \tag{22}$$

Given that $\forall b > 0$, the Markov process of $\{\underline{x}(k)\}_{k \geq 0}$, is a function of b, and this dependence is explicit denoted by $\underline{x^b}(k)$. The equation below provides the definition of $\underline{X^b}(t)$ as the continuous time interpolation of $\underline{x^b}(k)$.

$$\underline{X^b}(t) = \underline{x^b}(k) \text{ if } t \in [kb, (k+1)b) \tag{23}$$

The learning algorithm, as expressed in equation 14, can be redefined as the equation below, with θ being a composition of the action vector, $\underline{\alpha}$ and the payoff, $\underline{\beta}$.

$$\underline{x}^b(k+1) = \underline{x}^b(k) + \underline{b}\underline{g}(\underline{x}^b(k), \underline{\theta}^b(k)), \tag{24}$$

This above algorithm satisfies the following conditions:

- $\{\underline{x}(k), \underline{\theta}(k-1)\}_{k\geq 0}$ is a Markov process.
- The automata's outputs are limited to finite sets. $\underline{\theta}(k)$ is limited to the interval [0,1]. Thus $\underline{\theta}(k)$ gets values from a compact metric space *S*.
- $\underline{g}$ is independent of b and is continuous and bounded.
- $\{\underline{\theta}(k)\}_{k\geq 0}$ is an independent identically distributed sequence for a specific configuration, $\underline{x}(k) = \overline{x}$.
- Suppose B be a Borel subset of S. One-step transition function $TF(\underline{\theta}, 1, B) = probability(\underline{\theta}(k) \in B | \underline{\theta}(k-1) = \underline{\theta}, \underline{x}(k) = \underline{x})$, is independent of b, k, and $\underline{\theta}$. In other words, if $\underline{x}(k)$ is given, $\underline{\theta}(k)$ is independent of $\underline{\theta}(k-1)$.
- $TF(\underline{\theta}, 1, .|\underline{x})$ has a unique invariant probability measure called M($\underline{x}$), which is independent of $\underline{\theta}$. The probability measures M($\underline{x}$) are trivially tight since S is compact.
- $\int \underline{g}(\underline{x}, \underline{\acute{\theta}}) TF(\underline{\theta}, 1, d\underline{\acute{\theta}}|\underline{x})$ is not dependent on $\underline{\theta}$ and is continuous for $\underline{x}$.

The following theorem results from weak convergence theory [64] provided that the above conditions are met.

Theorem 1. The sequence of interpolated processes $\{X^b(.)\}_{b>0}$, converges weakly to $z(.)$ when $b \to 0$ for $L_{R\text{-}I}$ algorithm, as specified by the following equation.

$$\frac{dz}{dt} = [h_1(z), h_2(z), \dots, h_{N_P}(z)]; z(0) = x(0) \tag{25}$$
$$h_i(z) = z_i(1 - z_i) f_i(z) = z_i\left(d_{i1}(\underline{z}) - D_i(\underline{z})\right); i = 1,2, \dots, N_P$$

4.3. Convergence of CGG in ICGG-CP

First, we propose some assumptions and remarks about ICGG-CP, which are used in our analysis in this section. Then, we determine the equilibrium points of CGG's ODE in ICGG-CP and analyse the stability of these equilibrium points in the following subsections. Finally, we present some theorems related to the convergence of CGG.

Assumption 3: Every referee has a constant credit that is assumed to be non-negative.

Assumption 4: Each player utilizes the weighted average function based on the credits of its neighboring referees as their policy function.

Assumption 5: From the selected actions of players, if $\underline{n_1}(k)$ equal to $[L_1, L_2, \dots, L_{N_R}]$, the function $\sum_{j=1}^{N_R} C_j \times G_j(k)$ will reach the maximum possible value based on the CGG network, N, the credits of referees, C, and the criteria function of the referees, G.

Remark 1: A uni-modal performance criterion of referees be defined generally as follows:

$$G_i(\lambda_t) = c + a e^{\frac{-(\lambda_i^* - \lambda_t)^2}{b}} \tag{26}$$

In round t of the CGG, λ_t represents the ratio of neighboring player nodes with a "yes" action to all adjacent players for the referee i, while c, a, and b are learning parameters. A predefined value 1 is assigned to ${\lambda_i}^*$ when the ith referee approves the block, and 0 when rejected.

Remark 2: The predetermined rules for creating the underlying ICGG-CP graph state that a player cannot have more than a certain number of adjacent referees.

Remark 3: Predetermined rules for constructing the underlying graph of ICGG-CP limit the number of referees.

Remark 4: The initialization of the players' action probability vector is determined by their opinion of the block's acceptance or rejection in ICGG-CP. If a player doesn't respond within the duration of the validating block, an action probability vector with equal probability will be used for it. Consequently, $x_i(0)$ should be in $\{\mathrm{P}_{\mathrm{approve}}, \mathrm{P}_{\mathrm{disapprove}}, 0.5\}$, given that $\mathrm{P}_{\mathrm{approve}}$ and $\mathrm{P}_{\mathrm{disapprove}}$ represent the initial of players approving and disapproving a block, respectively.

Remark 5: According to assumptions 3 and 4, the following statements must be true:

$$\begin{aligned} \forall i \in 1,2,\dots,N_P, \underline{n_1}^{NR_i} = \left[n_1^{m_1},\dots,n_1^{m_{|NR_i|}}\right] : \exists \underline{n_1}^{NR_i} = \underline{L^i} \\ = \left[L_{m_1}, L_{m_2}, \dots, L_{m_{|NR_i|}}\right] \; s.t \; \mathcal{G}^i(\underline{L^i}) \\ > \mathcal{G}^i(\underline{n_1}^{NR_i}), \forall \underline{n_1}^{NR_i} \neq L_i \end{aligned} \tag{27}$$

Remark 6: Based on the referees' sent data and a predetermined decision rule, the owner determines block validation in ICGG-CP. The threshold of the minimum percentage of referees who agree to approve or reject

the block, denoted by MA, is one of the predetermined decision rules. Referees will have the same function as the performance criterion function in CGG if they all agree to validate the block.

Remark 7: The deterministic configurations $\underline{x^*}$ that provide $\underline{n_1}(k)$ equal to $\underline{L} = [L_1, L_2, \dots, L_{N_R}]$, are compatible points.

4.3.1 Equilibrium points and stability property

Theorem 2. The deterministic configurations, $\mathcal{X}^*$, are only the equilibrium points of the ODE given in equation 25.

Proof: Setting $\frac{dz}{dt} = 0$ can determine the equilibrium points of ODE in equation 25. The solutions can be classified in the following ways ($\forall i = 1,2, \dots, N_P$):

1. $z_i \in \{0,1\}$
2. $\exists z_0 \in (0,1): z_i = z_0 \; and \; f_i(\underline{z}) = 0$
3. combinations of categories 1 and 2; $z_0 \in (0,1): z_i = z_0 \; or \; z_i \in \{0,1\}$

According to definition 9, the deterministic configuration $\underline{z} \in \mathcal{X}^*$ can be one of the equilibrium points that $\forall i = 1,2, \dots, N_P: z_i \in \{0,1\}$.

The initial probability value of players in ICGG-CP be determined using the method discussed in Remark 4, which results in $\forall i = 1,2, \dots, N_P: x_i(0) \in \{P_{\text{approve}}, P_{\text{disapprove}}, 0.5\}$. Thus, the third solution is impossible for achieving equilibrium points in this application. In the following, we prove that the second solution is impossible in ICGG-CP. The conditions $\forall i = 1,2, \dots, N_P: \exists z_0 \in (0,1), z_i = z_0$ are only satisfied when all players agree to approve or reject the block or if no player confirms the block within the time limit. First, we show that if all players approve a block ($\forall i = 1,2, \dots, N_P: z_i = P_{\text{approve}}$), $f_i(\underline{z})$ cannot be equal to zero. The other cases can be proved in the same manner. In this case, if all referees approve the block, according to Remarks 1 and 6, they will have the same function as the performance criterion function, and $E[\sum_{j=1}^{N_R} C_j \times G_j]$ can be maximized through providing $\underline{n_1}$ equal to $[|NP_1|, |NP_2|, \dots, |NP_{N_R}|]$. The configuration $\underline{z} = [P_{\text{approve}}, \dots, P_{\text{approve}}]$ is not compatible due to the assumption that $P_{\text{approve}} < 1$, because there exists a configuration $\underline{z^*} = [1,1, \dots, 1]$ that satisfies the following equation:

$$d_{i1}(\underline{z}) z_i + d_{i2}(\underline{z})(1 - z_i) < d_{i1}(\underline{z}) z_i^* + d_{i2}(\underline{z})(1 - z_i^*) \qquad (28)$$

Based on Lemma 2, we can say that there's a player j and action $r \in \{\alpha_1, \alpha_2\}$ that satisfy $d_{iy}(\underline{z}) < D_j(\underline{z})$, given that we've proven $\underline{z}$ to be non-compatible. As a result, $f_i(\underline{z})$ cannot be equal to zero, according to equation 25.

Theorem 3. Configuration $\underline{x^*}$ is an asymptotically stable equilibrium point of ODE (equation 25) over $\mathcal{X}^*$, with components that maximize the function $E[\sum_{j=1}^{N_R} C_j \times G_j]$ through providing $\underline{n_1}$ equal to $[L_1, L_2, \dots, L_{N_R}]$.

Proof: We employ the Lyapunov theorems [65] for autonomous systems to prove this theory. This proof involves transferring the origin to an equilibrium point $\underline{x}^*$, followed by the introduction of a Lyapunov function to assess the stability of this point. According to Remark 7 and Lemma 2, the deterministic configurations $\underline{x}^*$that result in $\underline{n_1}(k)$ being equal to $\underline{L} = [L_1, L_2, \dots, L_{N_R}]$ follow the following relation:

$$\forall i = 1,2,\dots,N_P \text{ and } r \in \{\alpha_1, \alpha_2\}: x_{ir}{}^* \in \{0,1\}, d_{ir}(\underline{x}^*) = D_i(\underline{x}^*) \quad (29)$$

The origin was shifted to $\underline{x}^*$ using the following transformation.

$$\hat{x}_{ir} = x_{ir} - x_{ir}{}^* \quad (30)$$

Because x_{ir} is a probability vector, the value of $\hat{x}_{ir}$ must be between -1 and 1. Consider this positive definite Lyapunov function:

$$V(\underline{\hat{x}}) = -\sum_i \sum_{y=t_i} \hat{x}_{iy} \times \ln(1 - \hat{x}_{iy}); t_i = \begin{cases} 1 \text{ if } x_i{}^* = 1 \\ 2 \text{ if } x_i{}^* = 0 \end{cases} \quad (31)$$

According to the defined Lyapunov function and $x_{it_i}{}^* = 1$, only $-1 \le \hat{x}_{iy} \le 0$ are members of the domain of $V(\underline{\hat{x}})$ ($D(V(\underline{\hat{x}}))$). Also, $\forall \underline{\hat{x}} \in D(V(\underline{\hat{x}}), V(\underline{\hat{x}}) \ge 0$ and is zero only if $\hat{x}_{iy} = 0$ for all i. The following equation expresses the time derivative of $V(.)$.

$$\dot{V}(\underline{\hat{x}}) = -\sum_i \sum_{y=t_i} \frac{d\hat{x}_{iy}}{dt} \times \mathrm{v}(\hat{x}_{iy}); \quad (32)$$

$$\mathrm{v}(\hat{x}_{iy}) = \left[\frac{\ln(1 - \hat{x}_{iy}) \times (1 - \hat{x}_{iy}) - \hat{x}_{iy}}{1 - \hat{x}_{iy}}\right]$$

To continue the proof, we need to calculate the range of values for $\frac{d\hat{x}_{iy}}{dt}$. The components of the derivative of $\hat{x}$ with respect to time can be expressed as the following equation using equation 30.

$$\frac{d\hat{x}_{ir}}{dt} = \frac{dx_{ir}}{dt} - \frac{dx_{ir}{}^*}{dt} = \frac{dx_{ir}}{dt} \quad (33)$$

The equation 34 can be reformulated by employing 24 and 29 as follows.

$$\frac{d\hat{x}_{ir}}{dt} = x_{ir}\left(d_{ir}(\underline{x}) - D_i(\underline{x})\right) = (\hat{x}_{ir} + x_{ir}{}^*)\left(d_{ir}(\underline{x}) - D_i(\underline{x})\right) \quad (34)$$

According that $x_{it_i}{}^* = 1$, equation 34 can be rewritten using equation 8 as follows:

$$\frac{d\hat{x}_{ir}}{dt} = \begin{cases} (\hat{x}_{i1}+1)(-\hat{x}_{i1})\left(d_{i1}(\underline{x}) - d_{i2}(\underline{x})\right) \ if\ r = \alpha_1 \\ (\hat{x}_{i2}+1)(-\hat{x}_{i2})\left(d_{i2}(\underline{x}) - d_{i1}(\underline{x})\right) \ if\ r = \alpha_2 \end{cases} \quad (35)$$

Since $-1 \le \hat{x}_{iy} \le 0$, if $t_i = 1 \rightarrow d_{i1}(\underline{x}) > d_{i2}(\underline{x}) \rightarrow \frac{d\hat{x}_{i1}}{dt} > 0$, and if $t_i = 2 \rightarrow d_{i2}(\underline{x}) > d_{i1}(\underline{x}) \rightarrow \frac{d\hat{x}_{i2}}{dt} > 0$, thus implying $\frac{d\hat{x}_{ir}}{dt} \ge 0$. The following cases can occur for each term of this derivative, depending on the value of $\hat{x}_{iy}$:

1. $-1 \le \hat{x}_{iy} < 0$: $\mathrm{v}(\hat{x}_{iy}) > 0$ and $\frac{d\hat{x}_{iy}}{dt} > 0 \rightarrow \frac{d\hat{x}_{iy}}{dt} \times \mathrm{v}(\hat{x}_{iy}) > 0$
2. $\hat{x}_{iy} = 0 : \mathrm{v}(\hat{x}_{iy}) = 0 \rightarrow \frac{d\hat{x}_{iy}}{dt} \times \mathrm{v}(\hat{x}_{iy}) = 0$

Therefore, the inequality $\dot{V}(\underline{\hat{x}}) \le 0$ holds for all configurations $\underline{\hat{x}}$, and is zero only when $\underline{\hat{x}}$ equals zero for all i and r. Thus, by applying Lyapunov theorems to autonomous systems, we can prove the asymptotic stability of equilibrium point $\underline{x}^*$ for ODE 25.

Lemma 4. Configuration $\underline{x}^*$, which is an asymptotically stable equilibrium point of ODE (equation 25) over $\mathcal{X}^*$, is compatible.

Proof: Imagine about a solution $\underline{x}^*$, where $\mathrm{x}^*{}_i \in \{0,1\}$, is an asymptotically stable equilibrium point of ODE (equation 25). From the expression for h_i in equation 25, we can obtain equation 36.

$$\begin{cases} \frac{\partial h_i}{\partial x_j} = x_i(1-x_i)\frac{\partial f_i}{\partial x_j}; \forall j \ne i \\ \frac{\partial h_i}{\partial x_i} = (1-2x_i)f_i \end{cases} \quad (36)$$

Equation 37 gives the matrix $\left[\frac{\partial h_i}{\partial x_j}\right]$ of this solution $\underline{x}^*$. It is easy to verify from equation 20 that equation 38 is correct for this soloution.

$$\begin{cases} \frac{\partial h_i}{\partial x_j}(\underline{x}^*) = 0; \forall j \ne i \\ \frac{\partial h_i}{\partial x_i}(\underline{x}^*) = \begin{cases} f_i(\underline{x}^*) \ if \ \ x_i{}^* = 0 \\ -f_i(\underline{x}^*) \ if \ x_i{}^* = 1 \end{cases} \end{cases} \quad (37)$$

$$\frac{\partial h_i}{\partial x_i}(\underline{x}^*) = \begin{cases} -\left(d_{i1}(\underline{x}) - d_{i2}(\underline{x})\right) \ if \ x_i{}^* = 1 \\ d_{i1}(\underline{x}) - d_{i2}(\underline{x}) \qquad if \ x_i{}^* = 0 \end{cases} \quad (38)$$

According to $\underline{x}^*$ is an asymptotically stable equilibrium point of ODE (equation 25), the eigenvalues of the matrix $\left[\frac{\partial h_i}{\partial x_j}\right] : i \ge 1, j \le N_P$ must be negative. Therefore, the following equation holds for $\underline{x}^*$. As a

consequence of this equation, $d_{ir}(\underline{x}) \leq D_i(\underline{x})$ for all i and r, and $\underline{x}^*$ must be compatible based on Lemma 1.

$$\frac{\partial h_i}{\partial x_i}(\underline{x}^*) < 0 \xrightarrow{if\ x_i^*=1} -\left(d_{i1}(\underline{x}) - d_{i2}(\underline{x})\right) < 0 \rightarrow d_{i2}(\underline{x}) < d_{i1}(\underline{x}) \xrightarrow{using\ Equation\ 6} D_i(\underline{x}) = d_{i1}(\underline{x}) \rightarrow D_i(\underline{x}) > d_{i2}(\underline{x}) \quad (39)$$

$$\frac{\partial h_i}{\partial x_i}(\underline{x}^*) < 0 \xrightarrow{if\ x_i^*=0} d_{i1}(\underline{x}) - d_{i2}(\underline{x}) < 0 \rightarrow d_{i1}(\underline{x}) < d_{i2}(\underline{x}) \xrightarrow{using\ Equation\ 6} D_i(\underline{x}) = d_{i2}(\underline{x}) \rightarrow D_i(\underline{x}) > d_{i1}(\underline{x})$$

Lemma 5. The equilibrium points over $\mathcal{X}$ that are not compatible in the system are unstable.

Proof: Assume that the not compatible deterministic configuration $\underline{x}^*$ is an ODE equilibrium point that is asymptotically stable. So, according to Lemma 4, $\underline{x}^*$ must be compatible. The proof of the lemma is completed by contradicting the assumption of non-compatibility $\underline{x}^*$.

4.3.3 Convergence results

Theorem 4. The CGG in ICGG-CP always converges to a stable and compatible configuration with the initial configuration $\underline{x} \in \mathcal{X} - \mathcal{X}^*$ when the value of b is sufficiently small.

Proof: According to Remark 4, the initial probability value of player i in ICGG-CP is $x_i(0) \in \{\mathrm{P_{approve}}, \mathrm{P_{disapprove}}, 0.5\}$ for $i = 1,2,\dots,N_P$. Thus, $\underline{x}(0)$ belongs to $\mathcal{X} - \mathcal{X}^*$. Equation 14 describes the evolution of a CGG with $\mathrm{L_{R\text{-}I}}$ learning algorithm. $\{\underline{x}(k)\}_{k\geq 0}$ is a discrete-time Markov process, as implied by this equation. According to Lemma 3, this Markovian process is ergodic, and CGG can reach a maximum of z(.) , where $\frac{dz}{dt} = 0$, as described in Theorem 1. Only for all deterministic configurations, Theorem 2 shows that the derivative of z(.) is zero. Therefore, the solution to ODE (25) converges to a set that only contains equilibrium points of the ODE (25) for any initial configuration in $\underline{x} \in \mathcal{X} - \mathcal{X}^*$. As all equilibrium

configurations that aren't compatible are unstable, the theorem is proven. Figure 6 shows an overview of the steps involved in studying the behavior of CGG and its convergence in ICGG-CP.

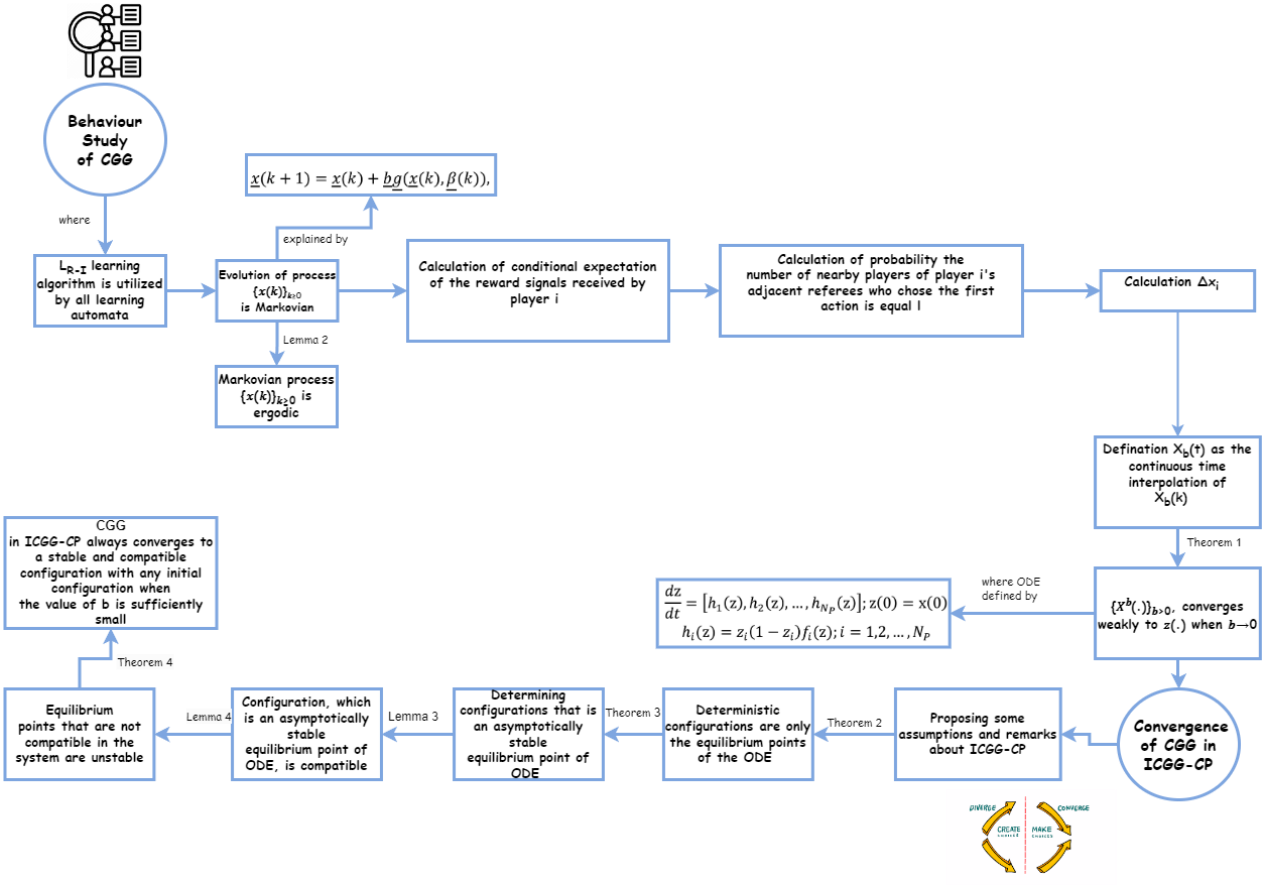


Figure 6: An overview of steps involved in studying the behaviour of CGG and its convergence in ICGG-CP.

5. Numerical results

In this section, we provide several experiments to assess the proposed consensus protocol. We use Python to simulate a consortium blockchain as a decentralized peer-to-peer network, and we utilize multiple ports to simulate the blockchain's multi-node environment. Table 1 shows the simulation parameters. We perform the experiments on a PC with a single Intel(R) Core i7-8550U 1.80 GHz CPU and 12 GB of RAM. We repeated all experiments described in this section 20 times to eliminate statistical bias and are reporting their average. Our evaluation of the performance of ICGG-CP is based on several metrics, including average entropy, average performance function, success percentage, throughput, average latency, average block time, average number of communications, and average reward, as described in detail in [32]. Weighted average G_{avg} (G_{w_avg}) and weighted total average G_{avg} ($\overline{G_{w\_avg}}$) are another performance evaluation metric of

ICGG-CP, which shows the average of the weighted scaled performance function during the execution of the CGG model in episode t and all episodes, respectively, according to the equation below.

$$\forall \lambda_i^*, \lambda_i \in [0,1] \quad (40)$$

$$G_{w\_avg}(\mathrm{t}) = \frac{\sum_{i=1}^{N_R} C_i \times G_i(\lambda_t^i)}{N_R \times \sum_{i=1}^{N_R} C_i \times Max\left(\left[G_1(\lambda_1^*), \dots, G_{N_R}(\lambda_{N_R}^*)\right]\right)};$$

$$\overline{G_{w\_avg}} = \frac{1}{T} \times \sum_{t=1}^{T} G_{w\_avg}(t)$$

The equations above use T to represent total episodes of block production, C_R to represent the credit of referee R, $G_R(t)$ to represent the scaled performance function value of referee R, and N_R to represent the number of total referees, N^i to represent the total number of neighboring players in referee i. The fraction of α_1 votes that referee i obtained from its neighboring players in episode t of block production is denoted by λ_i, whereas λ_i^* denotes the optimal fraction of α_1 votes in this cell that maximizes G_i.

The simulation estimates the referees' credibility by considering the number of nearby players and updates it based on behavior type and average response time for producing subsequent blocks. In the following section, we will detail a series of experiments.

Table 1. The simulation parameters

Simulation parameter	Description	Value
N	Total number of nodes	150
R	Number of referee nodes	5
P	Number of player nodes	100
x	Average size for transaction	1000B
T^I	Size of blocks	2MB
f_p	Percentage of player faulty nodes	30%
f_r	Percentage of referee faulty nodes	20%
K	Maximum number of executive rounds permitted for each referee	500
T_{min}	Threshold for entropy	0.01
$P_{approve}$	Initial probability for players to approve a block	0.8
$P_{disapprove}$	Initial probability for players to disapprove a block	0.2
$G(\lambda)$	Unimodal performance criterion of every referee	$0.9\, e^{-\lfloor \frac{(\lambda^* - \lambda_t)^2}{0.0625} \rfloor}$
a	Reward learning parameter for Learning automaton of every player	0.15
-	Rate of invalid blocks	0.1

5.1. Experiment 1

In this experiment, we aim to determine how the proposed consensus algorithm performs compared to CGG-CP [32], PBFT, and RB-BFT [53]. Figures 7-10 and Table 2 present the experiment's results. From this experiment's results, we can draw the following conclusions:

- Figure 7 illustrates the comparison of fault tolerance between CGG-CP and ICGG-CP based on failure type and role. The results indicate that ICGG-CP has significantly enhanced referees' fault tolerance, particularly Byzantine fault. Players' credibility-based weighted average policy and a reporting mechanism for referee faulty behavior enhances referee fault tolerance in ICGG-CP compared to CGG-CP. ICGG-CP's player fault tolerance is equivalent to CGG-CP's fault tolerance, as depicted in Figure 7. No attempts have been made to increase the fault tolerance of players in the improved CGG-CP protocol.
- It is evident from Figure 7's results that the fault tolerance rate of CGG-CP varies based on the fault type and participants' roles in consensus. The lower fault tolerance rate of referees in ICGG-CP is justified by their higher reliability and responsibilities in the consensus process compared to players. The fault tolerance rate of PBFT is 33%, while the fault tolerance rate of RB-BFT is 50%. Based on the results, it can be concluded that the proposed algorithm outperforms PBFT in terms of fault tolerance rate. Due to the higher number of players than referees in ICGG-CP, this protocol has more fault tolerance than RB-BFT.
- Figure 8 demonstrates that ICGG-CP has a higher throughput compared to PBFT and RB-BFT. Our proposed strategy of intelligently achieving consensus using the CGG model and grouping nodes has reduced the time it takes to reach consensus.
- Figure 8 indicates that when the number of nodes increases, PBFT and RB-BFT algorithms show a significant decrease in throughput, whereas the CGG-CP algorithm demonstrates a slow linear decline.
- Based on the results of this experiment, the CGG-CP outperforms the ICGG-CP slightly in terms of average latency and throughput. ICGG-CP's addition of fault tolerance mechanisms caused a slight decrease in throughput.
- Comparative analysis of communication complexity among ICGG-CP, CGG-CP, PBFT, and RB-BFT is depicted in Figure 9. The average number of communications is higher in ICGG-CP compared to RB-PBFT. The reason behind this is the large number of exchange messages sent by nodes to converge the CGG and reach a consensus. When the node count exceeds 73, ICGG-CP has lower communication complexity than PBFT, as depicted in Figure 9. Unlike PBFT, the proposed consensus method doesn't need all consensus nodes to send their vote to every node, justifying this result.
- Figure 10 gives the results of a comparison of the communication complexity of ICGG-CP and CGG-CP. Players in ICGG-CP have lower communication complexity compared to players in CGG-CP. Players in ICGG-CP have lower communication complexity compared to players in CGG-CP. This is because if a player's action probability converges to an action while executing the CGG of

ICGG-CP, they can notify their referees via a specific message and prevent sending repetitive actions. However, ICGG-CP has slightly increased the communication complexity of referees. This protocol requires referees to exchange more messages to increase their fault tolerance and tolerate the owner's fault, such as sending the results of their GG to other referees and the owner. It's more crucial to decrease the complexity of communication among players because they usually outnumber referees.

- Table 2 provides the results of a comparison of ICGG-CP with CGG-CP based on several metrics, such as throughput, average latency, and average block time, after generating 1000 blocks on the blockchain. According to Table 2, ICGG-CP has a higher weighted total average G_{avg} value than CGG-CP, but a lower total average G_{avg} value. Given that the credit-based weighted average

mechanism is defined for ICGG-CP players, the value of $\overline{G_{w\_avg}}$ in ICGG-CP is greater than that of CGG-CP, which employs the average policy for its players.

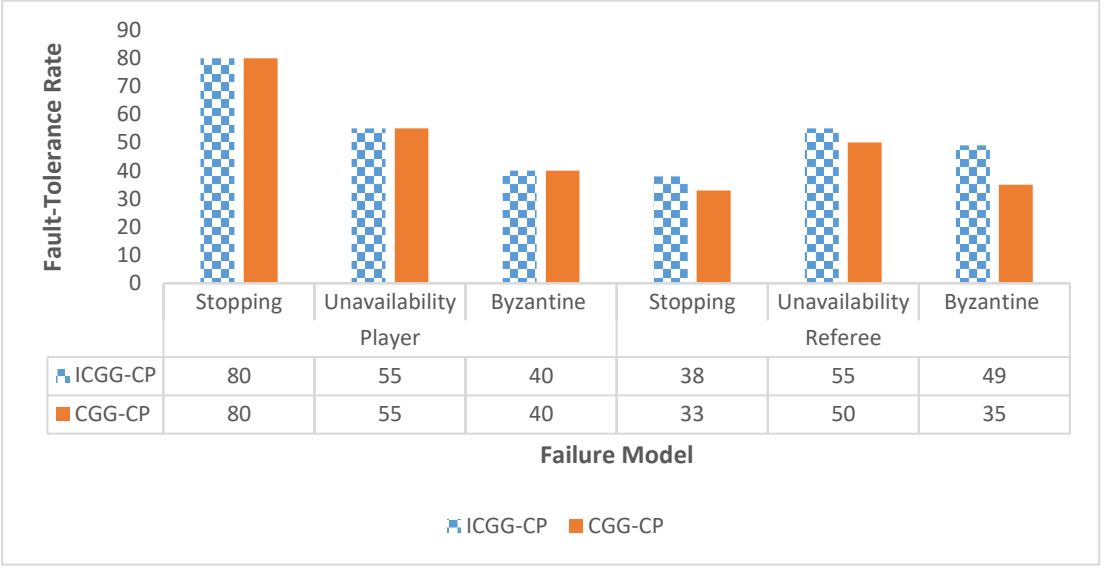


Figure 7 A comparison of the fault tolerance of ICGG-CP and CGG-CP

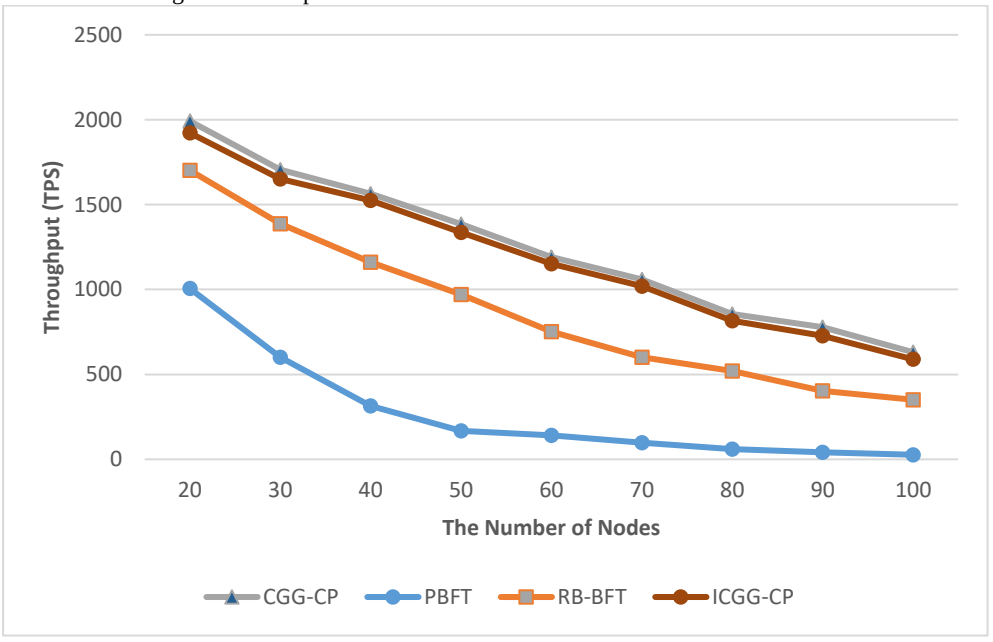


Figure 8 A comparison of throughput of ICGG-CP with CGG-CP, PBFT, and RB-BFT

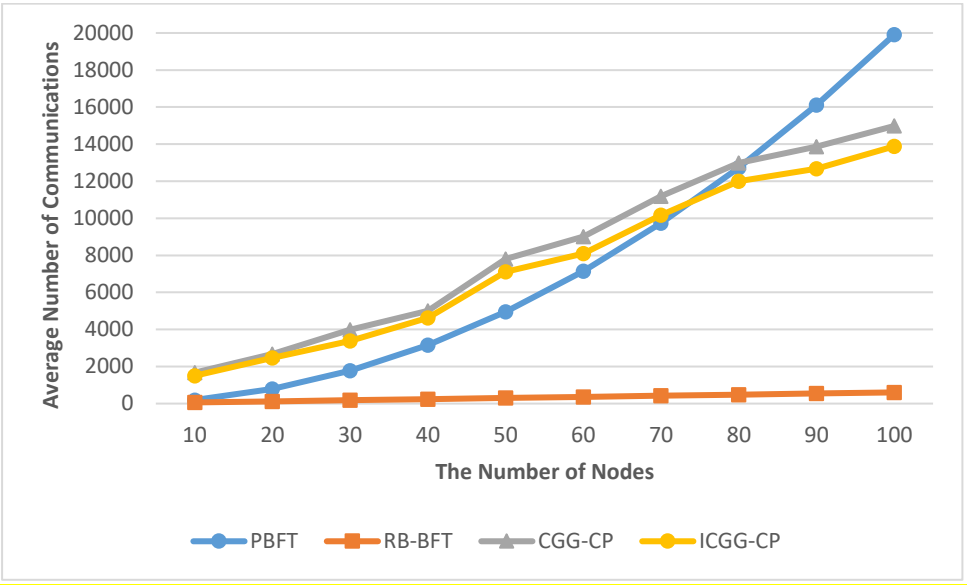


Figure 9 A comparison of communication complexity of ICGG-CP with CGG-CP, PBFT, and RB-BFT

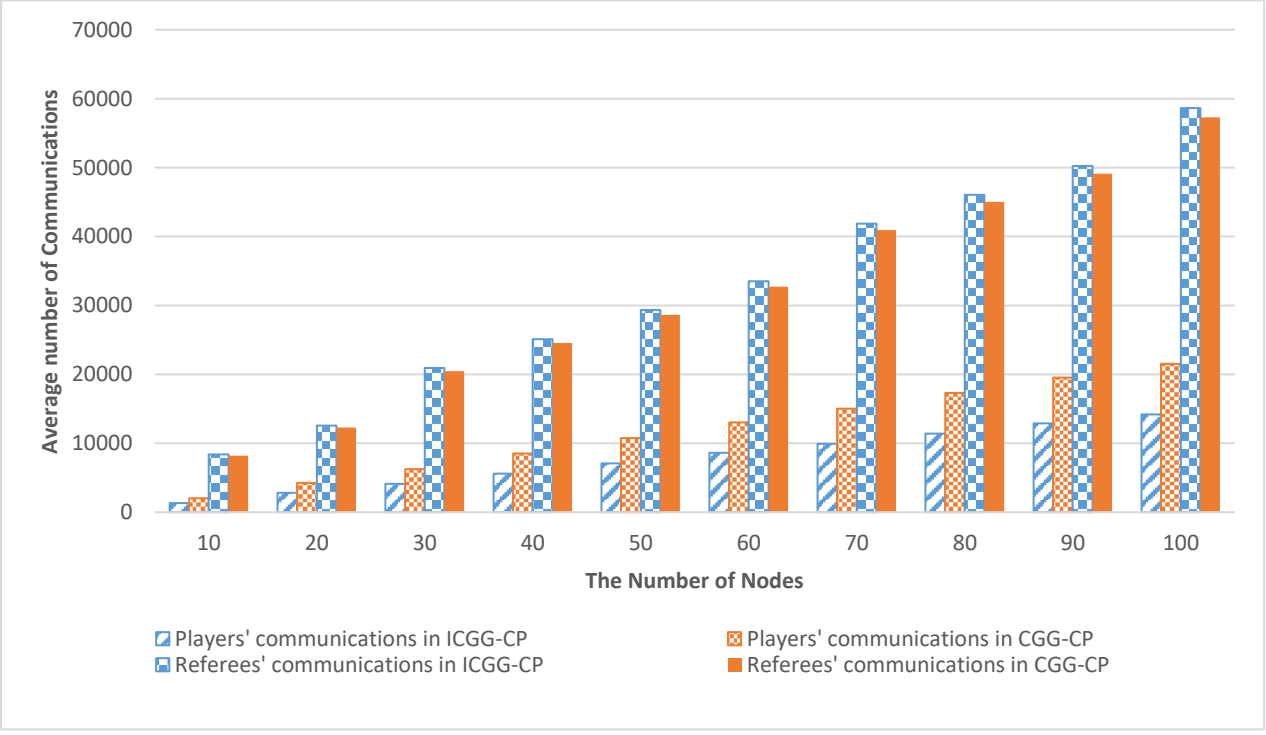


Figure 10 A comparison of the communication complexity of ICGG-CP and CGG-CP

Table 2. Comparison table of ICGG-CP with CGG-CP in Experiment 1.

Evaluation Parameter	ICGG-CP	CGG-CP
Success percentage	100%	100%
Throughput (TPS)	579.24	592.31
Average latency (seconds)	8.99	8.72
Total average G_{avg}	0.933	0.972
Weighted total average G_{avg}	0.991	0.891
Total average entropy	0.01	0.01
Average number of player communications	165.25	250.41
Average number of referee communications	4187.65	4091.97
Average player reward	3.37	3.12
Average referee reward	482.14	451.28
Average block time (seconds)	3.41	3.32

5.2. Experiment 2

In this experiment, we use evaluation metrics, such as throughput, entropy, and the weighted average performance function, to assess our proposed consensus's efficiency and convergence performance in 1000 block-creation episodes. Figure 11 and 12 shows the weighted average G_{avg} and the total average entropy versus block production episode, respectively. Across various episodes, the total average entropy and average G_{W_avg} are practically the same, at approximately 0.01 and 0.94. As a result, the convergence of G_{w_avg} to a desirable value has been achieved in all episodes of block creation. Figure 13 depicts the throughput in different episodes is roughly between 500 and 550. The throughput was reduced in some episodes where a block was rejected and a transaction was not added to the new block. Figure 14 shows credit and the average G_{avg} of referees in the proposed consensus in 100 episodes of block creation. The credentials of referees differ, and their behavior during block generation episodes causes their credits to change, as indicated in Figure 14-a. Figure 14-b demonstrates that referees with higher credibility achieved

a higher average G_{avg} during the production of 100 blocks. The result is justified because the players' reward policy in each CGG round in ICGG-CP is based on their referees' credit weighted average.

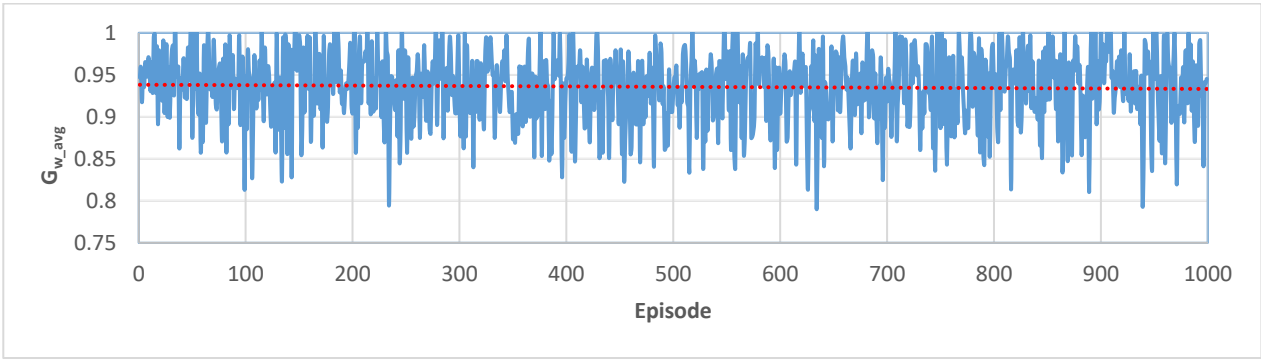


Figure 11: G_{w_avg} versus episode in experiment 2.

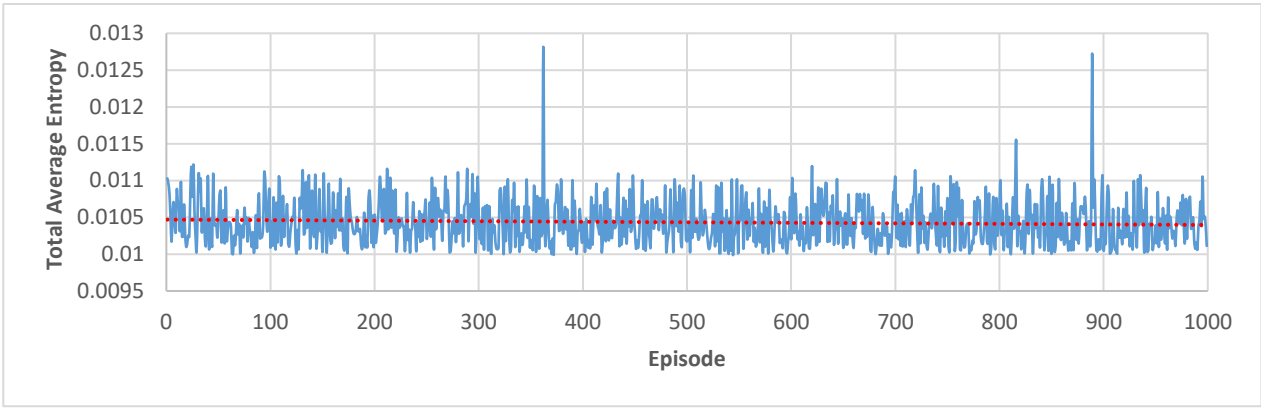


Figure 12: The total average entropy versus episode in experiment 2.

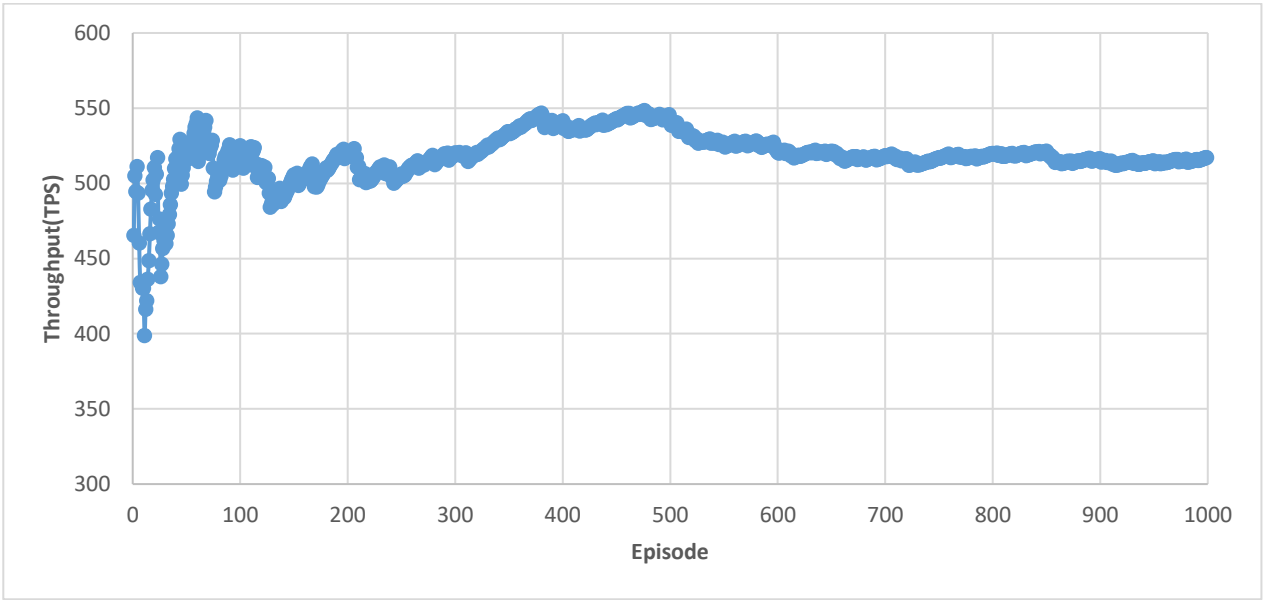
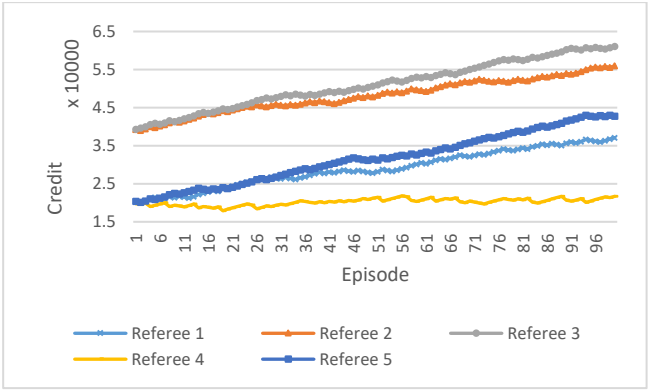
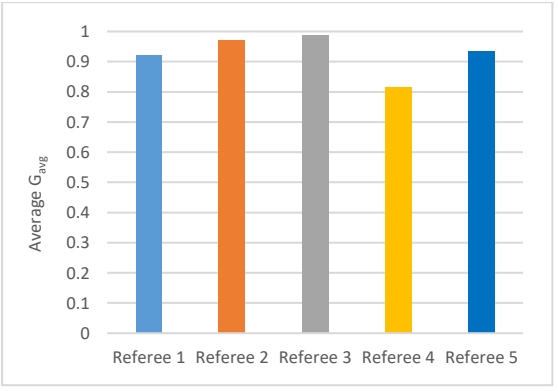


Figure 13: The throughput of ICGG-CP versus episode in experiment 2.



a) The credit of referees.



b) The average G_{avg} of referees.

Figure 14: The credit and the average G_{avg} of referees versus episode in experiment 2.

5.3. Experiment 3

In this experiment, we evaluate the effect of faulty nodes on the performance of ICGG-CP. For this objective, we analyze the referees' Byzantine failure and owner's failure in CGG-CP and summarize the results in Figures 15 to 16 and 17, respectively. One of the five available referees had a Byzantine failure during episodes 3, 7, 13, 18, 25, 51, 57, 62, 67, 82, 89, and 93 of block production in the faulty referee experiment. Figures 15 and 16 represent the credit and the G_{avg} of a faulty and non-faulty referee versus episode, respectively. Figure 15 implies that the referee's failure will not affect the consensus process since its G_{avg}

value decreased drastically during the error episodes. Furthermore, Figure 15 illustrates a decrease in the credibility of the faulty referee after it commit a mistake. Additionally, it can be deduced from Figure 16 that the G_{avg} value of the non-failure referee slightly decreased during the episodes of the fault. Figure 17 represents the throughput of ICGG-CP in the referees ' Byzantine failure scenario. Figure 17 indicates a slight decrease in throughput during episodes with the fault. This is due to the faulty referee taking longer during its GG process with its players, causing a delay in CGG execution step in ICGG-CP.

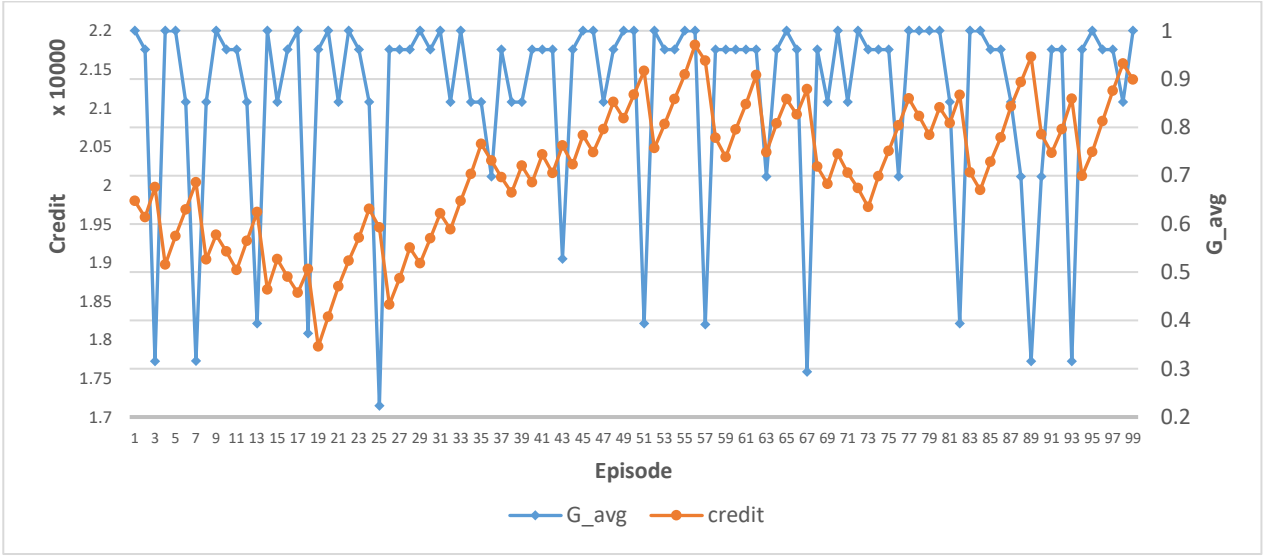


Figure 15: The credit and the G_{avg} of a faulty referee versus episode in experiment 3.

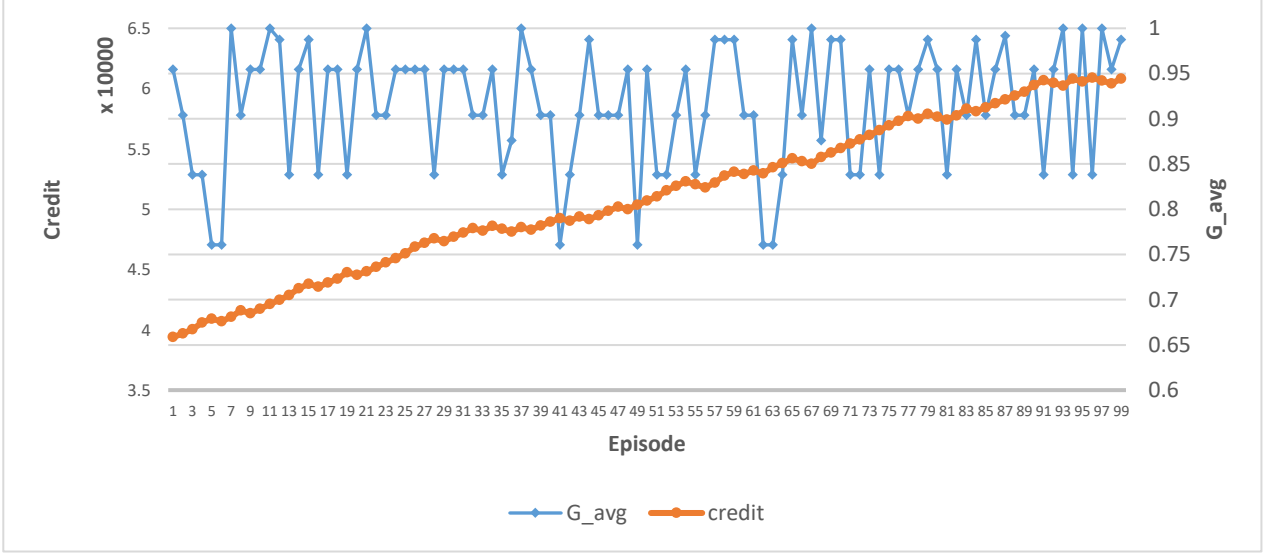


Figure 16: The credit and the G_{avg} of a non-faulty referee versus episode in experiment 3.

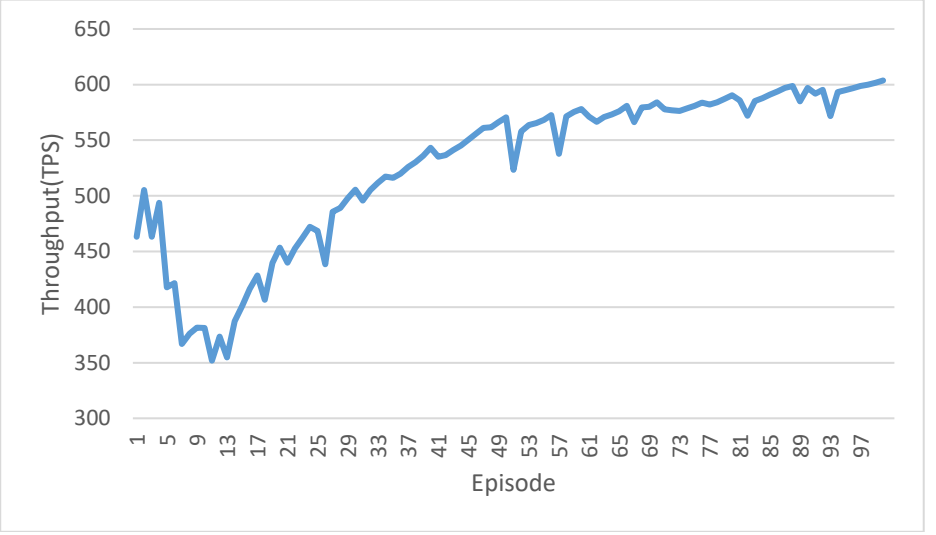


Figure 17: The throughput of ICGG-CP in the referees ' Byzantine failure in experiment 3.

The owner, one of the referees, faults episodes 3, 22, 44, and 52 of block production in the faulty owner experiment scenario. Figure 18 depicts the throughput and block time of ICGG-CP in this failure model. Figure 18 indicates that during fault episodes, there was a decrease in throughput and an increase in block production time. The reason can be explained in the way that the procedure considered to declare the owner's fault increases the block production time, but instead, it has made this method resistant to the owner's fault.

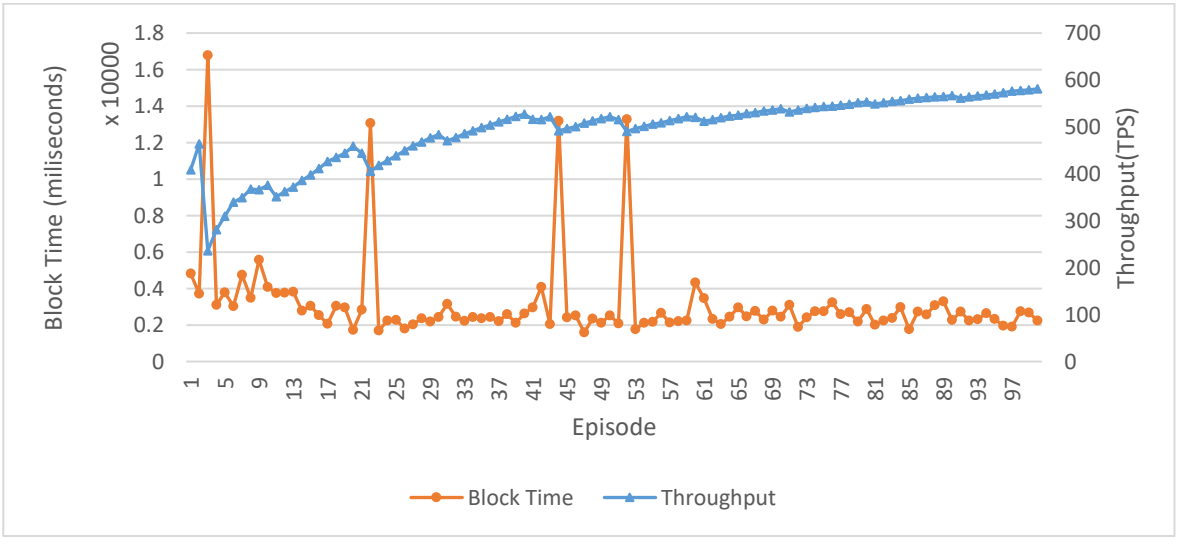


Figure 18: The throughput and block time of ICGG-CP in the owner's failure in experiment 3.

6. Conclusions

This article presents consensus algorithms by extending CGG-CP to improve fault tolerance and communication complexity. The proposed protocol, like CGG-CP, has three roles - referee, player, and owner - that act as consensus nodes. However, other referees also perform some of the owner's duties simultaneously to monitor the owner's tasks. Additionally, a secondary owner has been selected from the referees to take on the owner's responsibilities in the event of the owner's failure. Therefore, the proposed protocol is resistant to owner failure, unlike CGG-CP. Also, ICGG-CP increases referees' fault tolerance by proposing a mechanism for players to report referee-faulty behavior. Using a weighted average to calculate player rewards based on their adjacent referees, the ICGG-CP aims to increase the impact of nodes with higher credibility in validating or invalidating blocks, which ultimately enhances the referee's fault tolerance. Furthermore, ICGG-CP introduces a new feature that reduces player node communication costs. If players trust their referee, they can send a unique message informing them when their action probability converges to action in performing the CGG and avoid repetitive actions. This paper studies comprehensively and theoretically the Cellular Goore Game behavior and proves the convergence of CGG in the proposed protocol. Finally, several experiments were performed to assess the performance of the proposed protocol. According to empirical results, this approach improves fault tolerance and communication complexity.

In the future, we intend to design a new approach to accelerate the convergence of the CGG, leading to a decrease in communication complexity and an increase throughput the proposed consensus algorithm. Also, we plan to apply other AI techniques, like outlier analysis and clustering, as knowledge discovery methods in the cognitive engines of the presented consensus algorithm.

Appendix A

Proof of Lemma 3: Let $ES(k) = \{\underline{\alpha} | \underline{\alpha} = (\alpha_1(k), \alpha_2(k), \dots\dots, \alpha_{N_P}(k))^T\}$ be the event set that evolves state $\underline{x}(k)$. Therefore, we can express the following equation, where the definition of $f_{ES(k)}$ is in accordance with equation 14.

$$\underline{x}(k+1) = f_{ES(k)}(\underline{x}(k)) \tag{41}$$

Consider $Probability[ES(k) = e | \underline{x}(k) = \underline{x}] = \rho_e(\underline{x})$, wherein $\rho_e(\underline{x})$ is a real-valued function on $ES \times \mathcal{X}$. Let us define $d(\underline{x}, \underline{y}), m(\rho_e)$ and $\mu(f_e)$ using the equations below.

$$d\left(\underline{x}, \underline{y}\right) = \sum_i |x_i - y_i| \tag{42}$$

$$m(\rho_e) = sup_{\underline{x} \neq \acute{\underline{x}}} \frac{|\rho_e(\underline{x}) - \rho_e(\acute{\underline{x}})|}{d(\underline{x}, \acute{\underline{x}})} \tag{43}$$

$$\mu(f_e) = sup_{\underline{x} \neq \acute{\underline{x}}} \frac{d(f_e(\underline{x}) - f_e(\acute{\underline{x}}))}{d(\underline{x}, \acute{\underline{x}})} \tag{44}$$

The following statements are true:

1. The set ES is finite.
2. $(\mathcal{X}, d)$ is a compact metric space.
3. For every $e \in ES$, $m(\rho_e) < \infty$.
4. For every $e \in ES$, $\mu(f_e) < 1$. To prove this statement, suppose $\underline{x}$ and $\underline{y}$ as two states of the process $\{\underline{x}(k)\}_{k\geq 0}$. The equation below shows that $\mu(f_e) < 1$ since $0 < b_i < 1, 0 < \beta(k) < 1, \forall i$ from equations 15 and 44.

$$\mu(f_e) = sup_{\underline{x}\neq\underline{\acute{x}}} \frac{d\left(f_e(\underline{x}) - f_e\left(\underline{y}\right)\right)}{d\left(\underline{x}, \underline{y}\right)} = \frac{\sum_i |f_{e_i}(x_i) - f_{e_i}(y_i)|}{\sum_i |x_i - y_i|} \quad (45)$$
$$= \frac{\sum_i (1 - b_i\beta(k))|x_i - y_i|}{\sum_i |x_i - y_i|}$$

As a result, the Markovian process provided by equation 14 is strictly a distance diminishing process, as defined by Norman [63] in his definition.

Proof of Corollary in lemma 3: The equation that results from Lemma 3 is as follows:

$$d\left(\underline{x}^c, \underline{y}^c\right) = \sum_i (1 - b_i\beta(k))^c \times |x_i - y_i| \quad (46)$$

As $c \to \infty$, the right hand side of the preceding equation tends to zero. As a result, $\underline{x}^c \to \underline{y}^c$ as regardless of the initial configurations $\underline{x}$ and $\underline{y}$.

References

[1] E. S. Babu, B. V. R. N. Yadav, A. K. Nikhath, S. R. Nayak, and W. Alnumay, "MediBlocks: secure exchanging of electronic health records (EHRs) using trust-based blockchain network with privacy concerns," *Cluster Comput.*, vol. 26, no. 4, pp. 2217–2244, 2023.

[2] S. Rouhani, L. Butterworth, A. D. Simmons, D. G. Humphery, and R. Deters, "MediChain TM: a secure decentralized medical data asset management system," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2018, pp. 1533–1538.

[3] V. Merlo, G. Pio, F. Giusto, and M. Bilancia, "On the exploitation of the blockchain technology in the healthcare sector: A systematic review," *Expert Syst. Appl.*, p. 118897, 2022.

[4] L. W. Cong and Z. He, "Blockchain disruption and smart contracts," *Rev. Financ. Stud.*, vol. 32, no. 5, pp. 1754–1797, 2019.

[5] S. N. Khan, F. Loukil, C. Ghedira-Guegan, E. Benkhelifa, and A. Bani-Hani, "Blockchain smart contracts: Applications, challenges, and future trends," *Peer-to-peer Netw. Appl.*, vol. 14, pp. 2901–2925, 2021.

[6] A. Vacca, A. Di Sorbo, C. A. Visaggio, and G. Canfora, "A systematic literature review of blockchain and smart contract development: Techniques, tools, and open challenges," *J. Syst. Softw.*, vol. 174, p. 110891, 2021.

[7] P. Sharma, W. Wilfred Godfrey, and A. Trivedi, "When blockchain meets IoT: a comparison of the performance of communication protocols in a decentralized identity solution for IoT using blockchain," *Cluster Comput.*, pp. 1–16, 2022.

[8] O. Alfandi, S. Khanji, L. Ahmad, and A. Khattak, “A survey on boosting IoT security and privacy through blockchain: Exploration, requirements, and open issues,” *Cluster Comput.*, vol. 24, pp. 37–55, 2021.
[9] H. Li, L. Pei, D. Liao, X. Wang, D. Xu, and J. Sun, “BDDT: use blockchain to facilitate IoT data transactions,” *Cluster Comput.*, vol. 24, pp. 459–473, 2021.
[10] J. Miao, Z. Wang, Z. Wu, X. Ning, and P. Tiwari, “A blockchain-enabled privacy-preserving authentication management protocol for internet of medical things,” *Expert Syst. Appl.*, p. 121329, 2023.
[11] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” *Decentralized Bus. Rev.*, p. 21260, 2008.
[12] J. Garay, A. Kiayias, and N. Leonardos, “The bitcoin backbone protocol: Analysis and applications,” in *Annual international conference on the theory and applications of cryptographic techniques*, 2015, pp. 281–310.
[13] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, “Blockchain challenges and opportunities: A survey,” *Int. J. Web Grid Serv.*, vol. 14, no. 4, pp. 352–375, 2018.
[14] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, “An overview of blockchain technology: Architecture, consensus, and future trends,” in *2017 IEEE international congress on big data (BigData congress)*, 2017, pp. 557–564.
[15] F. R. Yu, J. Liu, Y. He, P. Si, and Y. Zhang, “Virtualization for distributed ledger technology (vDLT),” *IEEE Access*, vol. 6, pp. 25019–25028, 2018.
[16] M. Liu, F. R. Yu, Y. Teng, V. C. M. Leung, and M. Song, “Performance optimization for blockchain-enabled industrial internet of things (iiot) systems: A deep reinforcement learning approach,” *IEEE Trans. Ind. Informatics*, vol. 15, no. 6, pp. 3559–3570, 2019, doi: 10.1109/TII.2019.2897805.
[17] M. Liu, Y. Teng, F. R. Yu, V. C. M. Leung, and M. Song, “Deep Reinforcement Learning Based Performance Optimization in Blockchain-Enabled Internet of Vehicle,” in *IEEE International Conference on Communications*, 2019, vol. 2019-May, pp. 1–6. doi: 10.1109/ICC.2019.8761206.
[18] A. Bugday, A. Ozsoy, S. M. Öztaner, and H. Sever, “Creating consensus group using online learning based reputation in blockchain networks,” *Pervasive Mob. Comput.*, vol. 59, p. 101056, 2019, doi: 10.1016/j.pmcj.2019.101056.
[19] T. Pham and S. Lee, “Anomaly Detection in Bitcoin Network Using Unsupervised Learning Methods,” *arXiv Prepr. arXiv1611.03941*, 2016, [Online]. Available: http://arxiv.org/abs/1611.03941
[20] P. M. Monamo, V. Marivate, and B. Twala, “A multifaceted approach to Bitcoin fraud detection: Global and local outliers,” in *Proceedings - 2016 15th IEEE International Conference on Machine Learning and Applications, ICMLA 2016*, 2017, pp. 188–194. doi: 10.1109/ICMLA.2016.19.
[21] P. Monamo, V. Marivate, and B. Twala, “Unsupervised learning for robust Bitcoin fraud detection,” in *2016 Information Security for South Africa - Proceedings of the 2016 ISSA Conference*, 2016, pp. 129–134. doi: 10.1109/ISSA.2016.7802939.
[22] S. Sayadi, S. Ben Rejeb, and Z. Choukair, “Anomaly detection model over blockchain electronic transactions,” in *2019 15th International Wireless Communications and Mobile Computing Conference, IWCMC 2019*, 2019, pp. 895–900. doi: 10.1109/IWCMC.2019.8766765.
[23] S. Morishima, “Scalable anomaly detection method for blockchain transactions using GPU,” in *Proceedings - 2019 20th International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT 2019*, 2019, pp. 160–165. doi:

10.1109/PDCAT46702.2019.00039.
[24] M. Salimitari, M. Joneidi, and M. Chatterjee, “AI-enabled blockchain: An outlier-aware consensus protocol for blockchain-based iot networks,” in *2019 IEEE Global Communications Conference, GLOBECOM 2019 - Proceedings*, 2019, pp. 1–6. doi: 10.1109/GLOBECOM38437.2019.9013824.
[25] W. Wang *et al.*, “BSIF: Blockchain-based secure, interactive, and fair mobile crowdsensing,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 12, pp. 3452–3469, 2022.
[26] H. Jang and J. Lee, “An Empirical Study on Modeling and Prediction of Bitcoin Prices with Bayesian Neural Networks Based on Blockchain Information,” *IEEE Access*, vol. 6, pp. 5427–5437, 2017, doi: 10.1109/ACCESS.2017.2779181.
[27] A. Demir, B. N. Akilotu, Z. Kadiroglu, and A. Sengur, “Bitcoin Price Prediction Using Machine Learning Methods,” in *1st International Informatics and Software Engineering Conference: Innovative Technologies for Digital Transformation, IISEC 2019 - Proceedings*, 2019, pp. 144–147. doi: 10.1109/UBMYK48245.2019.8965445.
[28] S. McNally, J. Roche, and S. Caton, “Predicting the Price of Bitcoin Using Machine Learning,” in *Proceedings - 26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, PDP 2018*, 2018, pp. 339–343. doi: 10.1109/PDP2018.2018.00060.
[29] L. Li, A. Arab, J. Liu, J. Liu, and Z. Han, “Bitcoin options pricing using LSTM-Based prediction model and blockchain statistics,” in *Proceedings - 2019 2nd IEEE International Conference on Blockchain, Blockchain 2019*, 2019, pp. 67–74. doi: 10.1109/Blockchain.2019.00018.
[30] M. Saad, J. Choi, D. Nyang, J. Kim, and A. Mohaisen, “Toward characterizing blockchain-based cryptocurrencies for highly accurate predictions,” *IEEE Syst. J.*, vol. 14, no. 1, pp. 321–332, 2020, doi: 10.1109/JSYST.2019.2927707.
[31] R. Ameri and M. R. Meybodi, “Cognitive Blockchain and Its Application to Performance Optimization in Blockchain Systems,” 2022.
[32] R. Ameri and M. R. Meybodi, “The cellular goore game-based consensus protocol: a cognitive model for blockchain consensus,” *Cluster Comput.*, pp. 1–26, 2023.
[33] M. A. L. Thathachar and P. S. Sastry, *Networks of learning automata : Techniques for Online Stochastic Optimization*. Springer Science & Business Media, 2004.
[34] J. Akbari Torkestani, “An adaptive learning to rank algorithm: Learning automata approach,” *Decis. Support Syst.*, vol. 54, no. 1, pp. 574–583, 2012, doi: 10.1016/j.dss.2012.08.005.
[35] B. H. Lee and K. Y. Lee, “Application of S-model learning automata for multi-objective optimal operation of power systems,” *IEE Proceedings-Generation, Transm. Distrib.*, vol. 152, no. 2, pp. 295–300, 2005.
[36] M. L. Tsetlin, *Automaton Theory and Modeling of Biological Systems*, vol. 102. Academic Press New York, 1973. [Online]. Available: http://orton.catie.ac.cr/cgi-bin/wxis.exe/?IsisScript=UACHBC.xis&method=post&formato=2&cantidad=1&expresion=mfn=059666
[37] K. Narendra and M. Thathachar, “Learning automata: an introduction,” *IEEE Trans. Syst. Man. Cybern. B. Cybern.*, vol. 32, no. 6, 2012.
[38] M. A. L. Thathachar and M. T. Arvind, “Solution of Goore game using modules of stochastic learning automata,” *J. Indian Inst. Sci.*, vol. 77, no. 1, pp. 47–61, 1997.
[39] Y. U. Cao, A. B. Kahng, and A. S. Fukunaga, “Cooperative mobile robotics: Antecedents and directions,” in *Robot colonies*, Springer, 1997, pp. 7–27.
[40] D. Chen and P. K. Varshney, “QoS Support in Wireless Sensor Networks: A Survey.,” in *International conference on wireless networks*, 2004, vol. 233, pp. 1–7.

[41] A. Rezvanian, A. M. Saghiri, S. M. Vahidipour, M. Esnaashari, and M. R. Meybodi, “Recent advances in learning automata,” *Stud. Comput. Intell.*, vol. 754, pp. 1–458, 2018, doi: 10.1007/978-3-319-72428-7.
[42] M. F. Norman, “On the linear model with two absorbing barriers,” *J. Math. Psychol.*, vol. 5, no. 2, pp. 225–241, 1968, doi: 10.1016/0022-2496(68)90073-4.
[43] R. Ameri, M. R. Meybodi, and M. M. Daliri Khomami, “Cellular Goore Game and its application to quality-of-service control in wireless sensor networks,” *J. Supercomput.*, pp. 1–48, 2022.
[44] Dr Arati Baliga, “Understanding Blockchain Consensus Models,” *Persistent*, vol. 4, pp. 1–14, 2017.
[45] J.-M. Seigneur, “Distributed Ledger Technologies (Blockchain) Ecosystem and Decentralization,” in *ITU Asia-Pacific Centre of Excellence Bangkok 2018 DLT Training*, 2018.
[46] N. A. Lynch, *Distributed algorithms*. Elsevier, 1996.
[47] G. T. Nguyen and K. Kim, “A survey about consensus algorithms used in Blockchain,” *J. Inf. Process. Syst.*, vol. 14, no. 1, pp. 101–128, 2018, doi: 10.3745/JIPS.01.0024.
[48] M. Castro and B. Liskov, “Practical byzantine fault tolerance,” in *OSDI*, 1999, vol. 99, no. 1999, pp. 173–186.
[49] J. Kwon, “TenderMint : Consensus without Mining,” *the-Blockchain.Com*, vol. 6, pp. 1–10, 2014, [Online]. Available: tendermint.com/docs/tendermint.pdf
[50] K. Lei, Q. Zhang, L. Xu, and Z. Qi, “Reputation-based byzantine fault-tolerance for consortium blockchain,” in *2018 IEEE 24th international conference on parallel and distributed systems (ICPADS)*, 2018, pp. 604–611.
[51] H. Qin, Y. Cheng, X. Ma, F. Li, and J. Abawajy, “Weighted byzantine fault tolerance consensus algorithm for enhancing consortium blockchain efficiency and security,” *J. King Saud Univ. Inf. Sci.*, vol. 34, no. 10, pp. 8370–8379, 2022.
[52] S. Tang, Z. Wang, J. Jiang, S. Ge, and G. Tan, “Improved PBFT algorithm for high-frequency trading scenarios of alliance blockchain,” *Sci. Rep.*, vol. 12, no. 1, pp. 1–12, 2022.
[53] F. He, W. Feng, Y. Zhang, and J. Liu, “An Improved Byzantine Fault-Tolerant Algorithm Based on Reputation Model,” *Electronics*, vol. 12, no. 9, p. 2049, 2023.
[54] X. Liu, Y. Liu, X. Li, H. Cao, and Y. Wang, “FP-BFT: A fast pipeline Byzantine consensus algorithm,” *IET Blockchain*, 2023.
[55] G. Danezis, L. Kokoris-Kogias, A. Sonnino, and A. Spiegelman, “Narwhal and tusk: a dag-based mempool and efficient bft consensus,” in *Proceedings of the Seventeenth European Conference on Computer Systems*, 2022, pp. 34–50.
[56] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance and Proactive Recovery,” *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002, doi: 10.1145/571637.571640.
[57] R. Kotla, L. Alvisi, M. Dahlin, A. Clement, and E. Wong, “Zyzzyva: Speculative Byzantine fault tolerance,” in *ACM Transactions on Computer Systems*, 2009, vol. 27, no. 4, pp. 45–58. doi: 10.1145/1658357.1658358.
[58] R. Guerraoui, N. Knežević, V. Quéma, and M. Vukolić, “The next 700 BFT protocols,” in *Proceedings of the 5th European conference on Computer systems*, 2010, pp. 363–376.
[59] P. Chen, D. Han, T.-H. Weng, K.-C. Li, and A. Castiglione, “A novel Byzantine fault tolerance consensus for Green IoT with intelligence based on reinforcement,” *J. Inf. Secur. Appl.*, vol. 59, p. 102821, 2021.
[60] K. Riahi, A. Abouaissa, and L. Idoumghar, “A Reinforcement Learning-Based Node Selection for PBFT Consensus,” in *2022 Ninth International Conference on Software Defined Systems (SDS)*,

2022, pp. 1–3.
[61] Y. Goh, J. Yun, D. Jung, and J.-M. Chung, “Secure Trust-Based Delegated Consensus for Blockchain Frameworks Using Deep Reinforcement Learning,” *IEEE Access*, vol. 10, pp. 118498–118511, 2022.
[62] Alen Lovrencic, “maximally connected component,” *Dictionary of Algorithms and Data Structures*, 2022. https://xlinux.nist.gov/dads/HTML/maximallyConnectedComponent.html (accessed Aug. 05, 2023).
[63] M. F. Norman, “Some convergence theorems for stochastic learning models with distance diminishing operators,” *J. Math. Psychol.*, vol. 5, no. 1, pp. 61–101, 1968.
[64] H. J. Kushner, *Approximation and weak convergence methods for random processes, with applications to stochastic systems theory*, vol. 6. MIT press, 1984.
[65] A. M. Lyapunov, “The general problem of the stability of motion,” *Int. J. Control*, vol. 55, no. 3, pp. 531–534, 1992.