Received: 23 March 2024 | *Revised:* 29 April 2024 | *Accepted:* 1 June 2024
DOI: 10.37917/ijeee.22.1.19 *Vol. 22 | Issue 1 | June 2026*

Iraqi Journal for Electrical and Electronic Engineering
Original Article

Open Access
IJEEE University of Basrah College of Engineering

TI-PBFT: Improved Practical Byzantian Fault Tolerance Consensus Mechanism Based on ID-Trust Algorithm

Sajad Hussien Faleh*, Dheyaa Jasim Kadhim
Department of Electrical Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq

Correspondance
*Sajad Hussien Faleh
Department of Electrical Engineering, College of Engineering,
University of Baghdad, Baghdad, Iraq
Email: sajad.faleh2102m@coeng.uobaghdad.edu.iq

Abstract
Blockchain innovation is gaining attention in fields like monetary exchange, edge computing, medical care, and data-security. Consortium chains, using lightweight consensus algorithms like PBFT, offer alternatives to proof-based mechanisms while maintaining decentralization, security, and scalability. However, it also has some limitations and challenges that need to be addressed to improve its performance and scalability. PBFT is a classical algorithm with high complexity due to three-stage broadcasting and arbitrary selection of master nodes. Its communication efficiency is low, and scalability issues arise when nodes are large, causing significant delays and performance degradation in unstable networks. Furthermore, the requirement for every node to bundle, check, and broadcast the exchange list in the pre-prepared, prepared and commit stages diminishes the efficiency of consensus and performance between nodes and comes down on network correspondence. The research proposes a new methodology for the consensus algorithm, focusing on high-trust nodes to protect the network from malicious actors and reducing computational overhead and latency by eliminating Byzantian nodes and grouping the remaining nodes into groups, each of which has a main node selected based on a higher trust score. According to the results, the suggested methodology leads to significant improvements in communication complexity and Byzantine fault tolerance compared to standard PBFT networks and previous works. This indicates a substantial enhancement in network efficiency and scalability, offering promising prospects for blockchain applications in various fields.

Keywords
Consensus Algorithm, PBFT, Blockchain, ID-Trust, Communication Complexity, Byzantian Fault Tolerance.

I. Introduction

In 2008 [1], Satoshi Nakamoto was the first to propose blockchain technology in published literature. Many researchers are interested in blockchain technology because of how quickly science and technology are developing. Blockchain applications are emerging in various industries, for example, financial, e-government, edge computing, health, and data security [2–5]. As of now, blockchains can be sorted into three kinds in view of their degree of openness: the first is a public blockchain, the second is a consortium blockchain, and the third is a private blockchain. Thus, consensus algorithms can be partitioned into public blockchain consensus, private blockchain consensus, and consortium blockchain consensus. The consensus algorithm, as a central part of the blockchain technique, assumes a significant role in guaranteeing decentralization, scalability, and security. Consensus algorithm design that accomplishes these objectives is of the utmost significance. Consensus algorithms, such as, ”Proof-of-Stake” (PoS) [6], ”Proof-of-Work” (PoW) [1] and ”Delegated Proof-of-Stake” (DPoS) [7] are normally utilized. PoW, requires nodes compete for the right to validate transactions through working to solve mathematical problems. However, PoW suffers from a host of problems such as, unfair concentration of computing power and significant energy consumption. To

address the problem of energy consumption in PoW, PoS has been proposed, which selects verification nodes based on the stake each node has, but this method suffers from weak decentralization in many cases. For this reason, DPoS was proposed, which improves scalability by electing a limited number of trusted delegates to validate transactions. While DPoS offers quick transaction speeds, it might raise concerns with respect to centralization. In a decentralized system, the failure of a solitary node doesn't disturb the whole network's activity, making it strong and secure. While proof-based consensus systems show great scalability, they experience the ill effects of low throughput and time delays. Furthermore, when the network expands, these systems consume substantial amounts of memory and processing power, resulting in enormous asset waste. As blockchain technique advances, consortium chains have acquired prevalence, turning into the favored decision for many applications. Consortium blockchains frequently utilize lightweight algorithms like Paxos [8], practical Byzantian fault tolerance PBFT [9] and Raft [10]. Raft and Paxos are generally utilized in systems without Byzantine nodes, while PBFT offers benefits when Byzantine nodes exist because of its ability to deal with network assaults and it doesn't require intensive computations. A distributed system can agree on a suggested value even in the presence of faulty or malicious nodes due to the consensus mechanism known as the PBFT algorithm. M. Castro and B. Liskov [9] initially presented it in 1999. It is a common option for blockchain platforms and other distributed systems due to its several benefits in comparison to alternative consensus methods, such as its rapid throughput and minimal latency. In outline, the development of blockchain innovation has moved from the period of public blockchains to the period of consortium chains. Consortium chains commonly use lightweight algorithms, with PBFT being noticeable within the presence of Byzantine nodes. These algorithms offer choices to proof-based mechanisms, addressing explicit difficulties while aiming to keep up with decentralization, security, and scalability. In any case, it likewise has some limitations and challenges that should be addressed to work on its performance and scalability. The high correspondence above of PBFT is one of its essential issues, since it could go about as a bottleneck for large-scale systems. This is because PBFT requires constant communication between all nodes inside the network, possibly causing huge idleness and network congestion. To handle this issue, researchers have recommended numerous improvements to the PBFT. To eliminate the quantity of messages expected for consensus, a few scholastics have proposed, for example, separating the network into smaller sub-networks by means of a sharing methodology. Others have proposed bringing down how much communication is required between nodes by employing gossip protocols or other methods. The weakness of PBFT to assaults by pernicious nodes is another issue. PBFT makes the supposition that something like 33 percent of the network's nodes are evil or faulty. Specialists have recommended various improvements to PBFT's fault tolerance to conquer this issue. To make it harder for malignant nodes to rig the consensus process, a few scholars have proposed, for example, utilizing threshold cryptography to ensure that only a specific number of nodes are signing off on a transaction. The last trouble is the frequent occurrence of view changes, which happen when the primary node responsible for directing the consensus process malfunctions. Especially in networks with large churn rates, view alterations could dial back the consensus process and make it less effective overall. In general, PBFT is a strong consensus algorithm, in spite of the fact that it has downsides and troubles. For distributed systems that use PBFT for consensus to continue to be developed and adopted, these problems must be resolved. This research presents an enhanced PBFT method that utilizes an iterative and dynamic computing model (ID-Trust) to address the issues of high communication complexity and low consensus efficiency in large-scale node networks. The following is the outline of the paper: Section II. describes related work; Section III. presents the system design of the proposed TI-PBFT consensus; Section IV. discusses and analyses experiment results with TI-PBFT; and Section V. presents recommendations and conclusions.

II. Related Work

The PBFT algorithm, developed by Castro and Liskov in 1999 [9], aimed to decrease the complexity from exponential to polynomial, hence enabling its use in practical scenarios. Even though 33 percent of hubs are Byzantine and over 66 percent of nodes are normal, the system can still function properly thanks to the PBFT algorithm. A three-phase protocol, a view-change protocol, and a garbage collection mechanism are all part of the algorithm. Fig .1shows the three-phase protocol in action, which is used to achieve network consensus in a distributed environment. However, it becomes increasingly difficult to develop huge apps because of the exponential growth in communication complexity during the preparation and confirmation phases as the network size increases.

Li, Wenyu, et al. [11] introduced a "scalable multi-layer PBFT consensus for blockchain" that organizes nodes into hierarchical layers, restricting communication within each group. They initially present an optimal double-layer PBFT and demonstrate a substantial reduction in communication complexity. Notably, they establish that even the distribution of nodes within sub-groups in the next layer minimizes communication complexity. Both the Faulty Number Determined (FND) and the Faulty Probability Determined (FPD) models are used to examine the security threshold. Zheng,

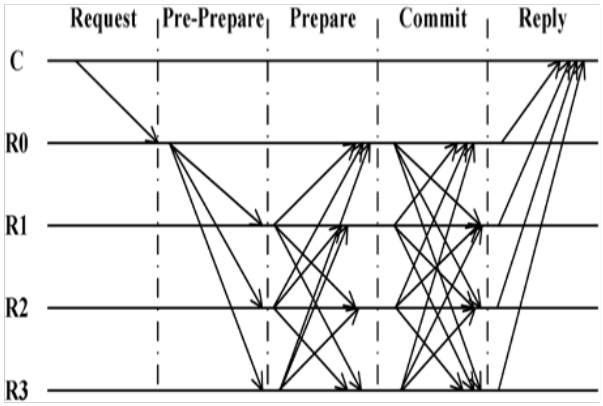

Fig. 1. Three phase protocol of PBFT [9].

Xiandong, et al. [12] introduced an enhanced PBFT algorithm. The algorithm begins by refining C4.5 through the integration of average information gain weight, addressing similar influences among restricted properties to enhance classification accuracy. Subsequently, nodes undergo classification using the enhanced C4.5, and those with high trust levels are selected to form the primary consensus group. Finally, the algorithm includes a built-in voting mechanism to designate the leader node. Zhong et al. [13] introduced an enhanced PBFT consensus algorithm, HR-PBFT, utilizing a hash ring. This approach effectively addresses the challenge of escalating communication due to the growing number of network nodes and mitigates the occurrence of frequent view changes. Tang, et al. [14] introduced tPBFT that is improved PBFT in three ways. Firstly, node election for leaders with this simplified procedure, an effective and trustworthy leader can be chosen for every consensus meeting. Secondly reduced communication overhead is achieved by employing a more efficient technique to generate the hash of the transaction list. Lastly Selecting consensus nodes to improve scalability, the consensus process is limited to a subset of nodes trusted nodes. Liu, Shannan, et al. [15] introduced an algorithm named Credit-Based Fault Tolerance (CBFT), enriched with a credit and grouping model. In CBFT, network nodes are grouped according to how quickly they react to management nodes. forming distinct consensus sets. Consensus is then achieved separately for the group inside and outside, effectively minimizing communication and enhancing security. Additionally, nodes are classified into various kinds depending on the credit model, assigning various responsibilities to each type, thereby diminishing the likelihood of the master node being a malicious actor. Liu, Wei, et al. [16] introduced an enhanced PBFT algorithm that integrates QoS-aware trust service evaluation to ensure adequate service transactions. The "QoS-aware trust practical Byzantine fault tolerance (QTPBFT) algorithm" establishes efficient consensus while notably curbing resource consumption. It has a global evaluation mechanism for trust services that takes quality of service into account, ranking the consistency of services in real-time. To optimize blockchain traffic, it employs a node selection mechanism, clustering a group of nodes with similar values, which will then cast their votes according to the trust service's global evaluation results. Wang, et al. [17] proposed a consensus technique called Grouped Practical Byzantine Fault Tolerance (GPBFT) that relies on feature trust. The program assesses the trust level of nodes in the transaction process using the Eigen Trust model. It then utilizes the trust level of nodes to select master nodes and proxy nodes. The algorithm categorizes nodes in the blockchain system into separate groups. Consensus within each group operates independently and does not impact other groups. This significantly decreases communication overhead during the consensus process with a high number of nodes. Jiang, Wangxi, et al. [18] presented an enhanced PBFT algorithm, termed TB-PBFT, founded on a comprehensive evaluation model. Initially, nodes are categorized into groups through a strategy that controls the multi-formation of a UAV cluster, markedly reducing communication complex issues. Subsequently, an evaluation model is introduced, amalgamating both entropy and TOPSIS methods, as well as Borda count. This model employs node behavior for evaluation. Other nodes' preferences are used to decide the final result. In order to guarantee the stability and security of the blockchain network, this model ultimately chooses the primary node based on the highest-ranking node. Table I shows a summary of previous studies:

III. System Model

A crucial part of the proposed improved PBFT system is the iterative trust algorithm (ID-Trust), which employs a score to categorize nodes. Here, ID-trust is introduced and its utilization in constructing improved PBFT (TI-PBFT) is demonstrated.

A. ID-Trust

ID Trust [19] constitutes a hierarchical distributed computing architecture that models and gauges trust based on explicit and implicit customer feedback, ensuring the utmost accuracy and integrity. The design improves the distribution computing of P2P commercial communication by utilizing three layers: evidence collecting, trust computation, and P2P communication. This topology is known as a three-layer P2P architecture. ID Trust adopts an iterative and dynamic trust-computing model, relying solely on the latest trust evidence, outperforming the traditional model that considers all trust evidence. It incorporates direct trust, indirect trust, global trust, and decision trust

TABLE I.
THE SUMMARY OF PREVIOUS STUDIES

Ref.	Year	Techniques	Algorithm	Performance evaluation metrices
[11]	2020	grouping nodes into different levels	-	Communication complexity, Success rate
[12]	2021	divides nodes into multiple groups	Improved C4.5	Communication complexity, security probability
[13]	2021	divides nodes into multiple groups	Hash function	Success rate, Communication complexity, Byzantian nodes
[14]	2022	Isolated ordinary nodes from consensus nodes	Eigen Trust model	Throughput, Error rate, Communication complexity
[15]	2022	divides nodes into multiple groups	credit grading	Latency, Communication overhead
[16]	2022	Isolated candidate nodes from consensus nodes	QoS-aware trust service global evaluation	Throughput, Latency, Communication complexity
[17]	2022	divides nodes into multiple groups	Eigen Trust model	Latency, throughput
[18]	2023	divides nodes into multiple groups	Comprehensive evaluation model	Success rate, Communication complexity, Byzantian nodes

to address individual malicious and collusion node behaviors, thereby enhancing the effectiveness of trust management.

The findings from the experiment highlight the remarkable efficiency of the ID Trust architecture. Particularly, the iterative calculation employed by ID Trust significantly enhances computational performance, with a considerably smaller time cost compared to the non-iterative approach. In contrast, the eigen trust model exhibits relatively lower performance due to its lack of a defined trust decision. The experiment underscores ID Trust's high sensitivity to malicious nodes, showcasing its proficiency in recognizing malicious behavior. Furthermore, ID Trust proves valuable for evaluating the trust score of PBFT nodes by leveraging trust evidence gathered during the PBFT consensus process.

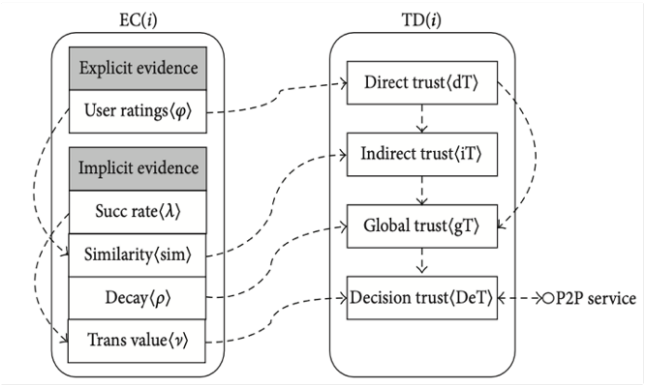

Fig. 2. Relations between evidences and trust model in ID Trust [19]

For determining the direct trust value between nodes, one can leverage evidence from their direct transactions. For instance, if node D receives a message from node C in the PBFT consensus process, node D can assign a direct trust value to node C by considering the success rate of past transactions between the two nodes. Nodes lacking direct transactions can rely on recommended trust values derived from other trusted nodes, a concept known as indirect trust. Additionally, the global trust value of each node can be computed by taking into account both direct and indirect trust values.

Moreover, the decision trust value can assess the reliability of each node's decisions in the consensus mechanism. This assessment is determined by evaluating the consistency of a node's decisions with those of other trusted nodes. Fig. 2 visually depicts the relationships between evidence and the trust model. By using ID Trust to calculate the trust score of PBFT nodes, The trustworthiness of nodes participating in the consensus behavior can be ensured, which can enhance the overall security and reliability of the PBFT consensus algorithm. Within ID-Trust, we initiate the creation of a distributed iterative computing architecture, illustrated in Fig. 3. The evidence collector (EC) retrieves historical transactions from the P2P system, providing an evidence vector to TD. Following this, TD calculates the trustworthiness of target nodes using the evidence vector and transmits the trust degree back to the P2P system in response to request instructions, as illustrated in Fig. 3.

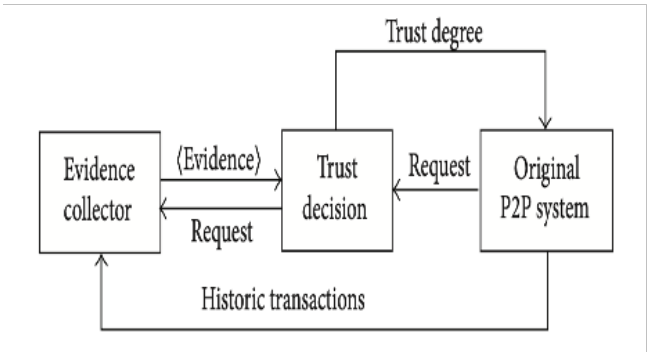

Fig. 3. Relations between EC, TD and P2P system based on ID-Trust [19]

B. TI-PBFT

Our algorithm, known as TI-PBFT (Trust Isolation Practical Byzantian Fault Tolerance), is an improvement upon the original PBFT algorithm. A grouping structure is introduced, and Byzantine nodes are isolated from groups based on their trust score, which is determined using the ID Trust algorithm as shown in Fig. 4.

This algorithm calculates the global trust of every node. Initially, each node's trust score is set to 1/n, where n is the total number of nodes in the network. Then it divides the nodes into groups m, with m being no less than three, and elects a primary node for each group. Each group must also have at least three nodes. The TI-PBFT algorithm's consensus flow as seen in Fig. 5. As the consensus mechanism begins, our algorithm excludes Byzantine nodes from the consensus nodes based on their trust score, defining them as ordinary nodes that do not share in the consensus. At the major node of each group, the client initiates transaction distribution. The PBFT algorithm is then applied until consensus behavior is attained in every group. Each group runs the PBFT algorithm, and at least three nodes must be present in each group for the consensus process to be valid. Fig .6 illustrates the consensus process of the TI-PBFT algorithm, with a detailed description provided below:

Request: a request message is sent to the primary nodes by the client as **(REQUEST, m, t, c)** $_{\sigma c}$ where: t is request's timestamp of message m, while σ_c is client and is the signature information of request message.

Pre-prepare: after receipt the previous mentioned request message m, the primary node transmits the message **(PRE-PREAPARE, n, v, m, p, d(m))**$_{\sigma p}$ to group's other nodes, where n and v are the sequence and the view numbers, respectively, while, σ_p is the main node number of the group, d(m) is the summary of m and is the signature information of pre-prepare message.

Prepare: in response to the main node's pre-prepare mes-

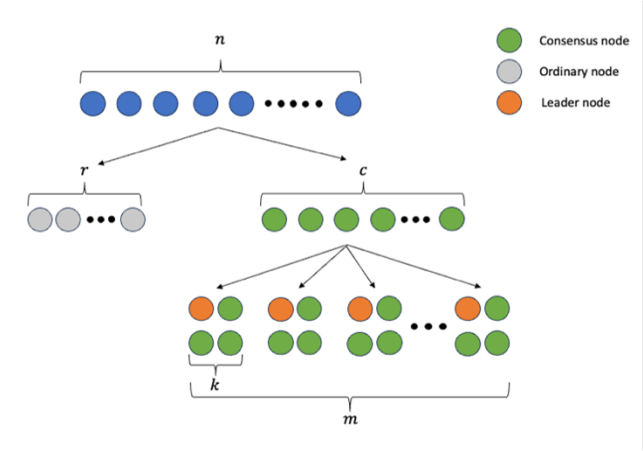

Fig. 4. Topology of the proposal TI-PBFT

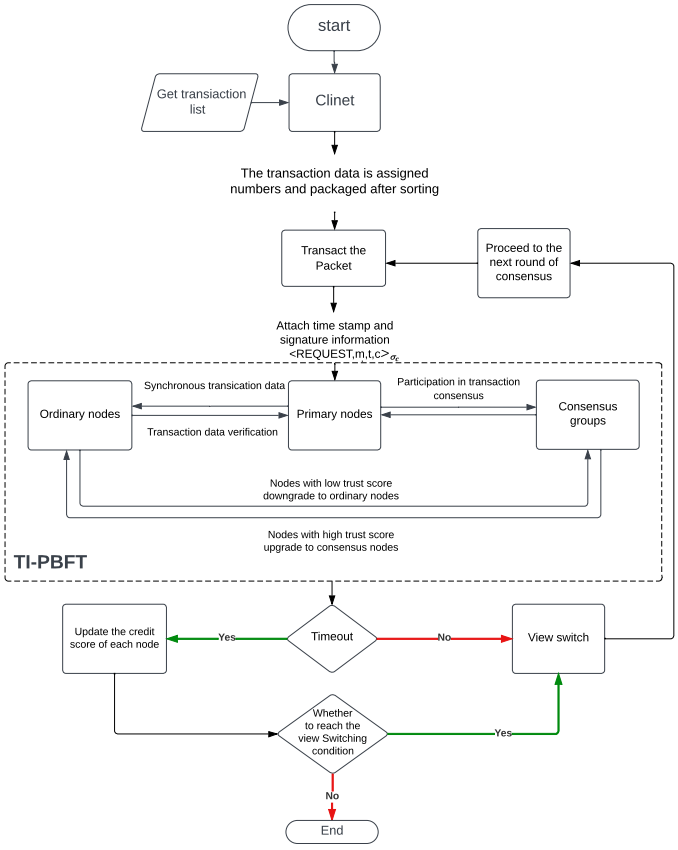

Fig. 5. Consensus flow of TI-PBFT algorithm

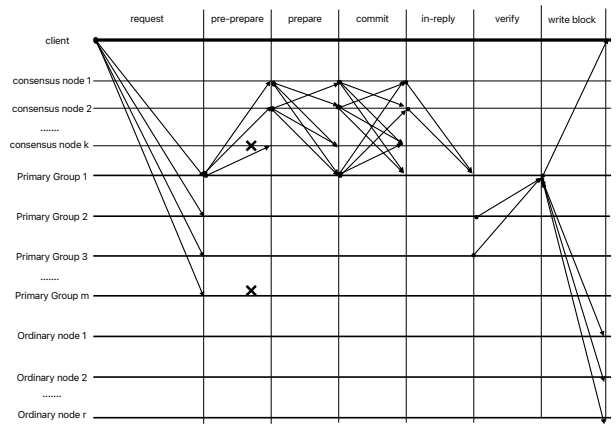

Fig. 6. The mechanism for reaching a consensus within the TI-PBFT algorithm

sage, as previously stated, after validating the message for accuracy, the message **(prepare, n, d, i, d(m))**$_{\sigma i}$ is sent by node i to the other nodes, where i is the node number in the group and σ_i is the signature information of prepare message.

Commit: after receipt more than $2f_m+1$ messages by node i, the validated prepare message is transmitted from other nodes in the group; and a commit phase is reached. Subsequently, node i transmits the messages (commit, n, v, i)$_{\sigma i}$ to group's other nodes, where is $f_m$represent the Byzantine nodes max number which can be handled in the group m.

In-reply: Once receiving valid commit messages, including its own, the primary node transforms a reply message of the form **(IN-REPLY, t, n, v, i, r)**$_{\sigma i}$ to the primary node. where r is the result of executing the requested operation.

Out-commit: once receiving more than $2f_m+1$ valid messages by primary node in-reply messages from other nodes in the group, a local consensus is reached by the group. All primary nodes will participate in the global consensus. Subsequently, the out-commit phase is reached by the primary node, then the messages **(OUT-COMMIT, v, n, r, p)**$_{\sigma p}$ are transmitted simultaneously to other main nodes where σ_p is the signature information of out-commit message.

Finally, after main node receiving further than replies $2f_g+1$, the main node will proof that the transaction is valid and write the transaction into the ledger and the primary node distributes the agreed transaction to ordinary nodes. where f_g is represent the number of faulty main nodes.

IV. Results and Discussion

In this section, Experiments were conducted on MATLAB, The TI-PBFT algorithm is evaluated by the metrics of communication complexity and maximum number of Byzantine fault tolerance.

A. Communication complexity:

Communication complexity is a crucial metric in the PBFT algorithm, assessing message overhead, bandwidth utilization, and network congestion. This study evaluates the TI-PBFT algorithm's communication complexity, providing insights into its impact on communication overhead compared to traditional PBFT. Understanding complexity helps identify bottlenecks and optimize protocol efficiency.

1) PBFT

Consider all of the nodes that make up the PBFT algorithm to be n. Request, pre-prepare, prepare, commit, and reply are the five phases that comprise the consensus process in the PBFT algorithm. The following equation is used to evaluate the communication complexity:

$$C_{PBFT} = 1+(n-1)+(n-1)(n-1)+n(n-1)+n \quad (1)$$

It simply turns to be:

$$C_{PBFT} = 2n^2-n+1 \quad (2)$$

2) TI-PBFT

Assuming the number of consensus nodes is (c=n-r) where r is number of ordinary nodes, then the consensus nodes divide to groups m each group has k nodes so the algorithm has seven phases, the request, pre-prepare, prepare, commit, in reply, out commit and write block phases. The communication complexity is evaluated using the following equation:

$$m+(k-1)m+(k-1)(k-1)m+k^2m+(k-1)m+m^2+r+1 \quad (3)$$

It simply turns to be:

$$C_{TI-PBFT} = 2k^2m+m^2+r+1 \quad (4)$$

This equation 4 calculates the communication complexity of TI-PBFT algorithm. The following Fig. 7 shows a Comparison between communication complexity of PBFT and the proposed TI-PBFT. As illustrated in Fig. 7, the value is observed to significantly and directly increase as the number of nodes is increased. Compared to the proposed methodology TI-PBFT, we note that the increase was very slight in the proposed methodology, as the communication complexity at 50 nodes in the network was about 500, while the communication complexity for the PBFT algorithm at 50 nodes in the network was about 5000. Therefore, the proposed methodology can be considered an actual improvement over the PBFT algorithm. An increase in communication complexity is observed in Fig.

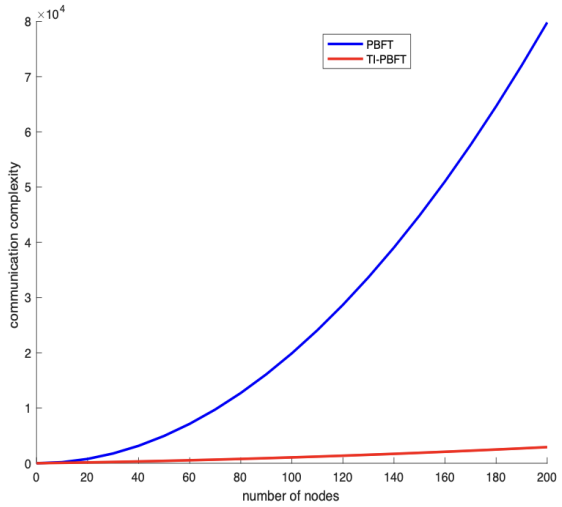

Fig. 7. Compare between communication complexity of PBFT and the proposed TI-PBFT.

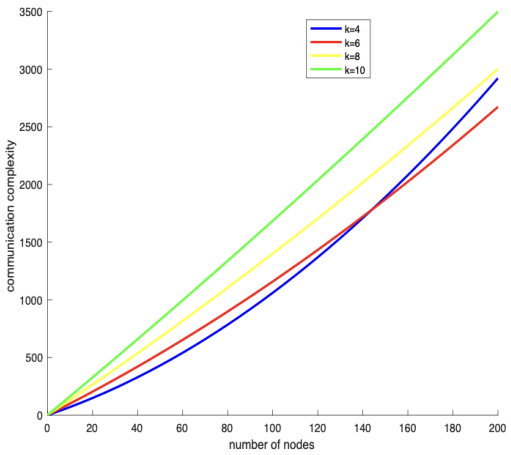

Fig. 8. The Communication complexity with different values of k and consensus nodes is 80% of n.

8, as the number of nodes k in each group of nodes increases. Therefore, we can say that the physical network's topology and the distribution of nodes determine the number of nodes for each group, and therefore, determining the value of k accurately depends on the results of testing and experimentation to reach the value that achieves the least complexity and the distribution of nodes throughout the network communications.

B. Maximum number of Byzantian fault tolerance:

Byzantine fault tolerance assesses a distributed system's ability to withstand malevolent or arbitrary actions. Security is essential in blockchain consensus methods such as PBFT. Evaluate the TI-PBFT algorithm's resilience to maximal Byzantine faults to determine its reliability in adversarial scenarios. Comparing standard Practical Byzantine Fault Tolerance (PBFT) with the proposed Trust-Isolated Practical Byzantine Fault Tolerance (TI-PBFT) enables improved system resilience. Let's commence with the PBFT algorithm,which can handle up to (n-1)/3 faults while achieving successful consensus.

$$B_{PBFT} = (n-1)/3 \tag{5}$$

While the improved TI-PBFT can tolerate more than PBFT, for calculating the max number of Byzantian fault tolerances which can be tolerated by the system, firstly determine the max number of Byzantine groups that the system can handle is $F_{groupfault} = (c/(k-1))/3$ where the term "Byzantine group" denotes either the presence of a primary Byzantine node or the number of Byzantian nodes is greater than (k-1)/3 in the group) then each reminder groups ,which is not Byzantian groups can tolerate maximum Byzantian fault tolerance is $F_{group} = (k-1)/3$ so the total maximum number of Byzantian fault tolerance equation is :

$$F_{TI-PBFT} = m * F_{group_{fault}} + c/k - F_{groupfault} * F_{group} \tag{6}$$

Additionally, assuming that ordinary nodes that have behavior of malicious nodes and the ID-trust model that exclude from consensus nodes so it can be added and considered at Byzantine nodes for the whole system and add it to above equation to become:

$$F_{TI-PBFT} = m * F_{group_{fault}} + c/k - F_{groupfault} * F_{group} + r \tag{7}$$

it simply turns to be:

$$F_{TI-PBFT} = (5mc - 2c - 2m^2 - m)/9m + r \tag{8}$$

We conduct simulation tests with k = 4, 6, and 10, where the total of all nodes in the network is below 500. This enables us to compare, for a range of network sizes, the maximum number of Byzantine nodes in the TI-PBFT method with the PBFT algorithm. The findings, which are shown in Fig. 9, show that the TI-PBFT algorithm performs better in terms of fault tolerance than the original PBFT.Furthermore, the TI-PBFT algorithm introduces an ID-trust model for selecting optimal primary nodes. The principal node is determined by aggregating scores from every node in the group. Given the minimal likelihood of the primary node being Byzantine, the system's ability to tolerate the greatest number of Byzantine nodes under particular conditions is effectively improved by the ID-trust model. By comparing the proposed methodology with the methodologies mentioned in Section 2, the results appear shown in Fig. 10, which displays a comparison between

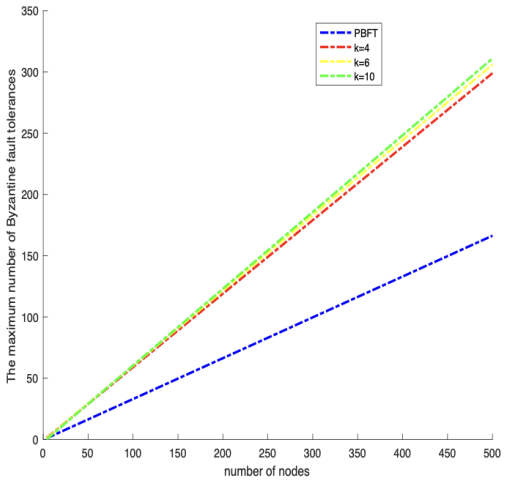

Fig. 9. F_{PBFT} and $F_{TI-PBFT}$ values.

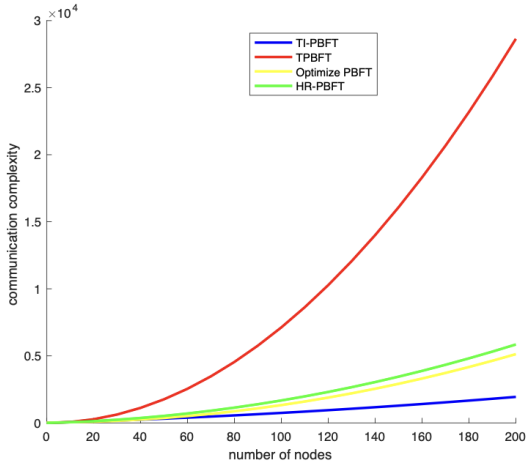

Fig. 10. A comparison between the communication complexity of the proposed methodology and previous works.

the communication complexity of the proposed methodology and previous works. The valueC_{TPBFT} in Fig. 10 is observed to be considerably higher compared to other algorithms, as the complexity value reached by the graphical curve is considered bad if the limited number of nodes (200) is considered, while in real-world scenarios the number of nodes be multiple times this number. The results of communication complexity of the proposed consensus is less than other previous works Optimized PBFT [12], HR-PBFT [13] and TPBFT [14]. It can be tested in real-world scenarios to ensure the stability of this performance or its variation as the number of nodes changes and the physical network structure changes. The proposed approach remains superior in terms of complexity of communications in the network, as it achieved the lowest complexity value compared to previous works, which confirms the great contribution that this proposal makes to improving communications in Blockchain networks. Although theoretical studies offer useful insights, real-world implementation and testing are essential to validate the suggested system's effectiveness and applicability. Future study could entail implementing the system in real-world situations or carrying out extensive simulations to assess its performance in different circumstances.

V. Conclusions And Recommendations

The study emphasizes the rising importance of blockchain technology in many industries and the increasing use of consortium chains, particularly on the PBFT consensus algorithm. The typical PBFT method has disadvantages such as communication complexity, frequent view changes, vulnerability to rogue nodes, and scalability concerns, which need enhancements for wider use. A grouping system is implemented to separate Byzantine nodes from groups based on their trust score calculated using the ID Trust method. The study proposes a method to compute the trustworthiness of each node in a network, removing low-trust nodes from the consensus process. The PBFT algorithm is used to achieve local agreement within each group, with each main node contributing to global consensus. The method aims to mitigate harmful actors' impact and enhance network security. To address computational and latency issues, consensus nodes should be grouped with main nodes. The study shows that the suggested system reduces communication complexity and increases the maximum number of Byzantine fault tolerances in a PBFT network compared to regular PBFT networks. It is speculated that these improvements would boost dynamism and safety, leading to the system's anticipated successful agreement. Procedures that enhance communication skills should be prioritized in continuous implementations and improvements, particularly in situations with several nodes. Resolving scalability challenges is crucial for the broader acceptance of blockchain innovation. Further exploration and refinement of the proposed technique are necessary to reduce communication complexity and enhance adaptability in situations involving additional nodes.

Conflict of Interest

The authors have no conflict of relevant interest to this article can be used.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] 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)*, pp. 557–564, Ieee, 2017.

[3] J. A. Abdulsaheb and D. J. Kadhim, "Classical and heuristic approaches for mobile robot path planning: A survey," *Robotics*, vol. 12, no. 4, p. 93, 2023.

[4] R. A. Falih, Y. Y. B. Jusoh, and D. J. Khadhim, "New research trends in designing e-government architecture based on blockchain technology," *Journal of Engineering*, vol. 29, no. 11, pp. 17–36, 2023.

[5] Z. H. Jaber, D. J. Kadhim, and A. S. Al-Araji, "Medium access control protocol design for wireless communications and networks review," *Int. J. Electr. Comput. Eng*, vol. 12, p. 1711, 2022.

[6] S. King and S. Nadal, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, no. 1, 2012.

[7] D. Larimer, "Delegated proof-of-stake (dpos)," *Bitshare whitepaper*, vol. 81, p. 85, 2014.

[8] L. Lamport, "Paxos made simple," *ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001)*, pp. 51–58, 2001.

[9] M. Castro, B. Liskov, *et al.*, "Practical byzantine fault tolerance," in *OsDI*, vol. 99, pp. 173–186, 1999.

[10] G. Moad, E. Rizzardo, and S. H. Thang, "Living radical polymerization by the raft process," *Australian journal of chemistry*, vol. 58, no. 6, pp. 379–410, 2005.

[11] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer pbft consensus for blockchain," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1146–1160, 2020.

[12] X. Zheng, W. Feng, M. Huang, and S. Feng, "Optimization of pbft algorithm based on improved c4. 5," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–7, 2021.

[13] W. Zhong, X. Zheng, W. Feng, M. Huang, and S. Feng, "Improve pbft based on hash ring," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–9, 2021.

[14] S. Tang, Z. Wang, J. Jiang, S. Ge, and G. Tan, "Improved pbft algorithm for high-frequency trading scenarios of alliance blockchain," *Scientific Reports*, vol. 12, no. 1, p. 4426, 2022.

[15] S. Liu, R. Zhang, C. Liu, C. Xu, and J. Wang, "An improved pbft consensus algorithm based on grouping and credit grading," *Scientific Reports*, vol. 13, no. 1, p. 13030, 2023.

[16] W. Liu, X. Zhang, W. Feng, M. Huang, and Y. Xu, "Optimization of pbft algorithm based on qos-aware trust service evaluation," *Sensors*, vol. 22, no. 12, p. 4590, 2022.

[17] Y. Wang, M. Zhong, and T. Cheng, "Research on pbft consensus algorithm for grouping based on feature trust," *Scientific Reports*, vol. 12, no. 1, p. 12515, 2022.

[18] W. Jiang, X. Wu, M. Song, J. Qin, and Z. Jia, "Improved pbft algorithm based on comprehensive evaluation model," *Applied Sciences*, vol. 13, no. 2, p. 1117, 2023.

[19] Z. Tan, X. Wang, X. Wang, *et al.*, "A novel iterative and dynamic trust computing model for large scaled p2p networks," *Mobile Information Systems*, vol. 2016, 2016.