Hindawi
Security and Communication Networks
Volume 2022, Article ID 6144664, 7 pages
https://doi.org/10.1155/2022/6144664

WILEY | Hindawi

Research Article

Improved PBFT Algorithm Based on Vague Sets

Guangxia Xu and Yishuai Wang

School of Software Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

Correspondence should be addressed to Guangxia Xu; xugx@cqupt.edu.cn

Received 28 October 2021; Revised 12 February 2022; Accepted 10 March 2022; Published 29 March 2022

Academic Editor: Jiewu Leng

The traditional PBFT consensus algorithm has several limitations in the consortium blockchain environment, such as unclear selection of primary node, excessive communication times, etc. To solve these limitations, an improved consensus algorithm VS-PBFT based on vague sets was proposed. VS-PBFT has three phases: node partition, primary node selection, and global consensus. Firstly, the nodes of the whole network are partitioned using the consistent hashing-like consensus algorithm, and then the local primary node is selected by the primary node selection algorithm in each partition. The local primary nodes run the four-phase PBFT consensus algorithm to complete the global consensus. The analysis of the VS-PBFT consistency algorithm shows that the algorithm can improve the fault-tolerant rate and reduce communication complexity, and the algorithm is dynamic; that is, node can join and quit adaptively.

1. Introduction

In recent years, the concept of blockchain has become very common. The definition of a blockchain is that blockchain is a mixed use of the chain structure, consensus algorithms, cryptography techniques, distributed data storage, peer-to-peer transmission, and automated smart contracts [1]. It is essentially a decentralized database. The data or information stored in the blockchain have the characteristics of decentralization, tamper resistance, traceability, collective maintenance, openness, and transparency [2]. In 2008, a person with the anonym Nakamoto published "Bitcoin: a peer-to-peer electronic cash system" [3]. Bitcoin was first applied to the financial industry as a grassroots electronic money. From this time on, the blockchain has gradually become a new idea for everyone to solve limitations.

The blockchain has four core technologies: distributed ledger [4], asymmetric cryptography algorithm [5], smart contract [6], and consensus mechanism [7]. Distributed ledger refers to the fact that transaction accounting is completed by multiple nodes distributed in different places, and each node records a complete account, so they can participate in the supervision of the legality of the transaction, and they can also testify for it together [8]. Asymmetric cryptographic algorithms are used to ensure the security of the data on the blockchain and the privacy of individuals. Each node on the blockchain has a pair of public key and private key. The public key is disclosed to the nodes of the whole network. On the contrary, the private key is private to the nodes of the whole network. One node signs the transaction with the private key, and the other node verifies the signature with the public key. As early as 1994, Szabo put forward the concept of smart contracts. Szabo described smart contracts as "a series of commitments specified in digital form, including agreements between parties to fulfill these commitments" [9]. In 2013, Dickerson et al. applied smart contracts to the real system for the first time [10]. Consensus mechanism is one of the core technologies of the blockchain, which is mainly used to make the nodes of the whole network reach consensus in a distributed environment. It needs to satisfy two properties:

Consistency: All the non-Byzantine nodes in the whole network store the same data.

Validity: All the information published by the non-Byzantine nodes will eventually be recorded in their ledger by all other non-Byzantine nodes [11].

Blockchains are usually divided into three types: Public chains, Consortium chains, and Private chains. The current mainstream consensus algorithms in Public chains include Proof of Work (PoW) [12], Proof of Stake (PoS) [13], and Delegated Proof of Stake (DPoS) [14]. The most popular consensus algorithm in the Consortium chains is the PBFT (Practical Byzantine Fault Tolerance) [15] and its derivatives. There are many restrictions in the Private chains, so they generally consider the use of a strong consensus protocol Raft [16] to achieve consensus under non-Byzantine failures.

In this article, Section 2 introduces several existing consensus algorithms. Section 3 discusses the preliminary knowledge of VS-PBFT, including the basic knowledge of vague sets and the traditional PBFT consensus algorithm. Section 4 describes the VS-PBFT algorithm, including node partition, primary node selection, and global consensus, in detail. Section 5 evaluates the VS-PBFT algorithm and its advantages are pointed out. Section 6 summarizes the whole article.

2. Related Work

The consensus mechanism of the blockchain can be classified according to the type of blockchain. The most well-known consensus algorithm in Public chains is PoW. In 1998, Dwork and Naor proposed the PoW algorithm for the first time [17]. In 2008, PoW was first applied to bitcoin. The main idea of the PoW consensus algorithm is to select the node responsible for generating blocks by finding the fastest node to calculate the difficulty value. The work of PoW is the certain amount of calculations that every node needs to perform; it will take a certain amount of time to find the hash result out. The node solving the hash equation in less time can get the right to generate blocks. We need to calculate the hash value of this block through the hash value of the previous block and the random value nonce, which is the solution to the hash equation. Once the node finds the nonce, it can calculate the solution of the hash equation. There is no fixed solution for this hash equation, so we can only find the final solution through constant trial and error. This method is also called hash collision. Hash collision is a probabilistic event. The more the attempts, the faster the calculation time and the greater the collision probability.

Consortium chains meet the requirements of the enterprise level, and its application is more targeted and efficient. Therefore, Consortium chains are becoming a hot topic of the blockchain in the future. The algorithm proposed in this article is also applicable to Consortium chains. The main consensus algorithm in Consortium chains is the PBFT. In 1999, Miguel and Barbara proposed the PBFT consensus algorithm [15], which reduces the complexity of the Byzantine protocol to a polynomial level, so that the Byzantine protocol can be used in distributed systems. In Section 3, we will introduce the detailed flow of the PBFT algorithm. In the blockchain project Hyperledger, the PBFT consensus algorithm was publicly implemented for the first time [18]. Although the PBFT consensus algorithm has $3f + 1$ fault tolerance and can guarantee a certain performance at the same time, it has many limitations, such as excessively high communication times, low scalability, and unclear primary node selection. In an unstable network, the system delay of PBFT is very high. At present, the improved algorithm mainly aimed at these points. In 2009, Clement et al. proposed a new method, Aardvark, to establish a Byzantine fault-tolerant replication system [19]. When f servers and any number of clients have Byzantine failures, this method enables the system to achieve a high Byzantine failure, peak performance, and high throughput, but this method does not solve the problem of excessive communication times of the PBFT algorithm, and its system scalability is not enough. GG Gueta et al. proposed the SBFT algorithm in 2019 [20], which addressed the problem of excessive communication times in the PBFT algorithm. To reduce the communication times to a linear level, a method, which used a fast path to reduce client communication, was proposed using collectors and threshold signatures. However, the selection of the primary node is still fuzzy.

3. Preliminaries

The materials and methods section should contain sufficient details so that all procedures can be repeated. If several methods are described, it can be divided into heading sections.

3.1. Overview of PBFT. The origin of the PBFT consensus algorithm can be attributed to the Byzantine failures. The efficiency of solving Byzantine fault is improved, and the complexity of the algorithm is reduced from exponential level to polynomial level, which makes Byzantine fault-tolerant algorithm feasible in practical system applications. The PBFT consensus algorithm is the first practical algorithm in the BFT class to work under a weakly synchronous network. It has three roles: client, primary node, and replica node. After the client puts forward the transaction request, it will be immediately sent to the primary node, and the primary node initiate the transaction voting in the global network, and then the replica node and the primary node will jointly maintain the fairness of the transaction voting. When the primary node fails, the view change program will be triggered to elect a new primary node.

We will briefly introduce the overall process of the PBFT algorithm. As shown in Supplementary Figure 1, it is the process of the PBFT algorithm, and replica node 3 is a Byzantine node.

(a) Request: At this phase, the client node sends a transaction request to the primary node.

(b) Pre-prepare: After the primary node receives the transaction request, it will verify the request and send a pre-prepared message to the replica node if the transaction is legal, otherwise the transaction is invalid and it will be discarded.

(c) Prepare: The replica node verifies the validity of the pre-prepared message. Once it is legal, the replica node will send the prepared message to other nodes

in the whole network and receive the prepared message from other nodes at the same time. After the node receives the prepared message, it will verify the legitimacy of the prepared message as soon as possible.

(d) Commit: When the node receives $2f+1$ legal prepared messages, the node enters the commit phase, where f refers to the number of Byzantine nodes in the system. During the commit phase, the node will send a commit message to other nodes in the whole network.

(e) Reply: When the node receives $2f+1$ commit messages, it will send a reply message to the client node. After the client node received $f+1$ identical reply messages, the whole consensus is completed.

The garbage collection mechanism and view change program are not the focus of this article. For more details, please read the reference [15].

3.2. Vague Sets. Most of the existing voting-based blockchain consensus algorithms only consider agreement and disagreement, but it is obviously not enough in practical application. In 1965, Zadeh proposed the concept of fuzzy sets [21]. Fuzzy sets give us a neutral option to vote, and we can use fuzzy sets to optimize the voting process in the blockchain to make it more in line with human thinking.

Fuzzy set refers to a given domain U, then a mapping from U to the unit interval $[0, 1]$ is called a fuzzy set on U or a fuzzy subset of U. The fuzzy set can be denoted as S. The mapping (function) $\mu S(\cdot)$ or $S(\cdot)$ is called the membership function of the fuzzy set S. For each $x \in U$, $\mu S(x)$ is called the membership degree of element x to fuzzy set S.

Gau and Buehrer further improved the theory of vague sets in 1993 [22]; they pointed out that the members of vague sets are three subintervals between $[0, 1]$. The three subintervals correspond to three kinds of information of the element ($u \in U$): favor, against, and abstentions. We can use two functions to represent a vague set S in the domain U. $t_S(u)$ is usually used to represent a truth membership function, and $t_S(u)$ is a lower bound on the grade of membership of u derived from the evidence for u. $f_S(u)$ is usually used to represent a false membership function, and $f_S(u)$ is a lower bound on the negation of u derived from the evidence against u. Both $t_S(v)$ and $f_S(u)$ are a certain number between $[0, 1]$, where $t_S(u) + sf_S(u) \le 1$.

When U is continuous, a vague set S can be denoted as

$$S = \frac{\int^U [t_S(u), 1 - f_S(u)]}{u}. \tag{1}$$

When U is discrete, a vague set S can be denoted as

$$S = \frac{\sum_{i=1}^{n} [t_S(u_i), 1 - f_S(u_i)]}{u_i}. \tag{2}$$

In general, the value of a vague set of an element can be denoted as

$$[t_S(u), 1 - f_S(u)]. \tag{3}$$

The concept of vague set mentioned above can be seen as a voting model. Assuming that S is the vague set of element u in U, the value of S is $[0.3, 0.7]$; from (3), we can calculate $t_S(u) = 0.3$, $f_S(u) = 1–0.7 = 0.3$. It means that the degree that u belongs to S is 0.3 and the degree that u does not belong to S is 0.3. If the total number of votes is 10, $(0.3, 0.7)$, it means that the number of votes favor is 3, the number of votes against is 3, and the number of votes abstention is 4.

Yong et al. proposed a general formula in 2008 to convert the vague sets into the final fuzzy score [23]:

$$\mu_S F = t_S(u) + \frac{1}{2}\left[1 + \frac{t_S(u) - f_S(u)}{t_S(u) + f_S(v) + 2\lambda}\right] \cdot \left[1 - t_S(u) - f_S(u)\right], \quad \lambda > 0. \tag{4}$$

In this article, we use this final score to select the primary node. We set to choose the highest final score; when the highest scores are equal, we randomly select a node as the primary node.

4. VS-PBFT Algorithm

4.1. Algorithm Overview. Blockchain technology has gained a lot of preference in the world due to its own anonymity and decentralization, and the research on its main core technology consensus mechanism is one of the most significant parts. The PBFT algorithm is widely used in Consortium chains with its own advantages. However, if the number of nodes in the used system increases, specifically, after reaching 100, the communication times of the system will rise sharply, and the selection of the primary node that plays a key role in the algorithm is unclear. Therefore, a new improved PBFT consensus algorithm based on vague sets was proposed. The overall algorithm flowchart is shown in Supplementary Figure 2.

The algorithm we proposed has three phases: node partition, primary node selection, and global consensus. Firstly, we will partition the nodes in the whole network, then we will select the primary node in each partition. The selection method of the primary node is based on the idea of vague set, so the selection of the primary node is more in line with people's thinking compared to ordinary voting-based scheme, and each partition selects one local primary node.

In the global consensus phase, we will select a global primary node among all the local primary nodes. This global primary node undertakes the similar task to the primary node in the PBFT consensus algorithm. We will reduce communication times by reducing the consensus phase. The effect of communication times makes the whole consensus process more efficient. After the consensus is completed, the new block will be added to the whole blockchain, and then the next round of consensus will begin.

4.2. Node Partition. The communication complexity of the PBFT algorithm running under the consortium blockchain condition is $O(N^2)$. In the case of large-scale nodes, the

number of communication times will increase exponentially. In a distributed system, partition is mainly for improving the scalability and availability of the system. In 1997, Karger and others of the Massachusetts Institute of Technology proposed consistent hashing algorithm. This article will use its idea to partition nodes.

Through node partition, The N nodes of the whole network are divided into k groups. Each group is represented by n_k and k is the group number. We use the hash value of IP corresponding to each node as the unique identifier of each node.

Firstly, we create a hash function H with a value space $[0, 2^{32} - 1]$. We organize the whole hash value space into a virtual circle, and the whole space is organized in a clockwise direction, that is, 0 and $(2^{32} - 1)$ coincide at zero. Thereafter, we randomly generate k points on the hash ring, and we need to continue to randomly generate k points until k mutually exclusive points are generated, which means that we randomly divide the hash ring into k areas. We name these k areas $1, 2, \ldots, i, \ldots, k$, respectively. In the next step, the IP corresponding to each node uses the same hash function H to calculate the hash value and determine the location of this node on the ring. Assuming that this position is in the i^{th} area, then this node is divided into the i^{th} area. If the calculated hash value is equal to the value of a certain boundary, we put it in the smaller area.

We assume that N is 4 and k is 3, which means that we need to generate three mutually exclusive hash values and divide the hash ring into three areas; and we calculate the hashes of N node IP and divide them into corresponding area. As shown in Supplementary Figure 3, three areas: area1, area2, and area3 are generated. IP0, IP2, and IP3 are divided into area2, area3, and area1, respectively. Since the hash value of IP1 is the same as that of random hash 2, IP1 is assigned to area2.

This situation occurs during the partition process, where there are too many nodes in one area, and too few nodes in the other. In those circumstances, the local primary node selected by the partition with uneven node distribution cannot represent the node of the whole network. In this situation, we introduce a virtual node mechanism, that is, calculate multiple hashes for each certain random hash, and each calculated hash is used as a new random hash point, called a virtual node. This can be achieved by adding a number at the back of the IP. According to the regional partition formed between virtual nodes, the number of virtual nodes is usually set to 32 or even larger, so the nodes can be relatively evenly distributed.

4.3. Primary Node Selection. When the partition is completed, it means that nodes with similar IP hash values have been allocated to the same area. Next, we need to run the primary node selection to elect the local primary node. All nodes in one area vote for the most suitable node to be the local primary node. We added the option of abstention in the voting process, and used the general model transformed vague set to obtain a comprehensive score, and the highest comprehensive score became the local primary node. The flowchart of primary node selection is shown in Supplementary Figure 4.

The primary node selection algorithm is as follows:

(1) All nodes in the same area will vote for other nodes in the same area within the specified time. There are three choices of favor, against, and abstention, and the three votes of each node are counted.

(2) Calculate the vague set value of each node by formula (3) according to the number of votes counted in step (1).

(3) Calculate the comprehensive score according to formula (4).

(4) Sort the comprehensive scores and use the node own the highest comprehensive score as the local primary node. If there are multiple same highest scores, one is randomly selected as the local primary node.

The local primary node of an area is equivalent to the leader of all nodes in the area, delegating other nodes to complete the consensus, and the state of the node in this area is consistent with the state of the local primary node. When an error occurs in the local primary node, a new node can be voted again to replace the old local primary node. Because of the access rules of Consortium chains, the probability of this circumstances happen is very small, and we can almost ignore it.

4.4. Global Consensus. Each area conducted a primary node selection, and selected k local primary nodes to participate in the global consensus. Supplementary Figure 5 is a network topology diagram after the primary node selection. Nodes 0, 1, and 2 choose node 2 as the local primary node; nodes 3, 4, and 5 choose node 4 as the local primary node; nodes 6, 7, and 8 choose node 6 as the local primary node; and nodes 2, 4, and 6 participate in the global consensus.

Global consensus also needs to run a primary node selection to select the primary node. We will not repeat this process here, we use $*$ to represent node 2 as the global primary node. The global primary node plays the same role as the primary node in the PBFT consensus algorithm. In the traditional PBFT consensus algorithm, with the increase in the number of nodes, the communication times of the algorithm will increase dramatically. The main function of the pre-preparation phase of the PBFT algorithm is to ensure that in the case of network disconnection or link disconnection, the nodes will reach agreement too, but this situation is almost impossible to occur in today's era of highly developed networks, so we reduce the number of communications by subtracting the pre-preparation phase in this article. Supplementary Figure 6 is a flowchart of the PBFT algorithm with the pre-preparation phase cut. The global consensus also follows this process.

The simplified version of the PBFT consensus process is as follows:

(1) Request: The client sends a transaction request m to the global primary node. In Supplementary Figure 6,

client C sends a transaction request m to global primary node 2*.

(2) Prepare: After the global primary node received the clients' transaction request m, it will broadcast the preparation message to the whole network nodes immediately. The scheme of the prepared message is $\langle \text{PREPARED}, m, v, n, d, t, n_i, i, Q_i \rangle$, where m is the original text of the request message, v is the current view number, n is the message sequence number of m, and d is the hash value of message m. t is the timestamp of message m, n_i is the partition number, i is the current node number, and Q_i is the digital signature of node i. After receiving the prepared message, the node will verify the message. After verification, the node will enter the prepared state and send a commit message $\langle \text{COMMIT}, m, v, n, d, t, n_i, i, Q_i \rangle$ to other nodes in whole network.

(3) Commit: After the node received the commit message, it will verify the message as in the prepared phase. When the same message sent by $2f + 1$ different nodes is verified, the node will send a reply message to the client.

(4) Reply: When the client receives $f + 1$ identical reply messages $\langle \text{REPLY}, m, v, n, d, t, n_i, i, Q_i \rangle$, the consensus is completed, the transaction is added into the blockchain.

5. Evaluation

In this part, we prove the superiority of the VS-PBFT algorithm through theoretical analysis.

5.1. Dynamic Analysis. The nodes in the blockchain are dynamic, and at any time, there may be nodes joining or exiting the blockchain. The traditional PBFT algorithm cannot detect the joining or exiting of nodes in time and dynamically adapt to the network environment.

The VS-PBFT algorithm proposed in this article uses the idea of hash consensus algorithm to place N nodes in the whole network into k areas, and each area selects a local primary node to participate in the global consensus. When a new node joins the blockchain network, it will run the hash algorithm to calculate the area that the node belongs to and divide it into the designated partition directly. When a node in one area exits, other nodes in the area can still vote for local primary nodes to participate in the global consensus. The algorithm is dynamic.

5.2. Communication Times Analysis. The traditional PBFT algorithm has five phases, and each phase needs to send a message for communication. First, the client sends a transaction request to the master node in the request phase, and the number of communications is 1. Thereafter, the primary node sends a pre-prepared message to other replica nodes, and the number of communications is $(N - 1)$. The prepared message is sent from the node to other nodes in the whole network in the preparation phase, and the number of communications is $(N-1)^2$. All nodes send commit messages to other nodes in the commit phase, and the number of communications is $N(N-1)$. All nodes send a completion message to the client in the reply phase, and the number of communications is N. Adding the communication times of the above five phases to get a consensus, the communication times T_1 of the traditional PBFT algorithm is

$$T_1 = 2N^2 - N + 1. \tag{5}$$

In our proposed VS-PBFT, we need to divide the N nodes of the whole network into k areas. We know that the number of nodes in the PBFT algorithm must not be less than 3, so the number of nodes participating in the global consensus must not be less than 3, thus $k \geq 3$. Since the number of nodes in each area is at least 1, $N \geq 3$.

In the VS-PBFT algorithm, one round primary node selection needs to send $(N/k - 1)N/k$ messages to select a local primary node. There are k areas, so k primary node selections are required; in addition, global consensus need one round primary node selection. For one round consensus, $(k + 1)$ primary node selections are required in total, and the number of communications is $(N/k - 1)(k + 1)N/k$. In the four-phase consensus, the number of communications in the request phase, preparation phase, confirmation phase, and reply phase is 1, $(k - 1)$, $k(k - 1)$, and k, respectively. Therefore, the total number of communications T_2 of VS-PBFT is

$$T_2 = \left(\left(\frac{N}{k} - 1\right)\frac{N}{k} + k\right)(k + 1). \tag{6}$$

As $N \geq 3$ and $k \geq 3$, $T_2 < T_1$. Therefore, the communication times of our proposed VS-PBFT algorithm are better than that of the traditional PBFT algorithm.

5.3. Fault Tolerance Rate Analysis. We all know that the maximum number of fault-tolerant nodes of the traditional PBFT algorithm is f_1:

$$f_1 = \frac{N - 1}{3}. \tag{7}$$

In the VS-PBFT algorithm, the total number of nodes in the whole network is N, and the nodes in the whole network are divided into k areas. We assume that the number of nodes in each area is equal, and the number of nodes in each area is N_k. For each area, in theory, as long as the number of Byzantine nodes is less than the number of normal nodes, the most suitable node can be selected as the local primary node, so the maximum number of fault-tolerant nodes in each area is $N/2k$. Therefore, the maximum number of fault-tolerant nodes of the VS-PBFT algorithm is f_2:

$$f_2 = \frac{N}{2}. \tag{8}$$

As $N \geq 3$, $f_2 > f_1$. Therefore, the fault tolerance rate of our proposed VS-PBFT algorithm is higher than that of the traditional PBFT algorithm.

6. Conclusion

In this article, we proposed an improved PBFT algorithm based on vague sets, named VS-PBFT. Above all, we partition the nodes and vote based on vague sets within the partitions. Each partition selects a local primary node with the highest score to participate in the four-phase consensus, so as to achieve global consensus. Theoretical analysis shows that our VS-PBFT algorithm is superior to the PBFT algorithm in fault tolerance and communication times, our algorithm is dynamic, and it can adapt to the joining and exiting of nodes at any time.

VS-PBFT has only been proved to be effective in theory, but there will be many limitations in practical applications, and the effect needs to be verified in practical environment.

In this period, blockchains are facing many limitations, one of which is that network isolation makes it extremely difficult to coordinate actions among different blockchains. Cross-chain technology is a good solution to this problem [24], but due to the lack of a consensus mechanism suitable for cross-chain technology, the development of cross-chain technology is considerably slow. Therefore, how to design a dynamic and adaptive cross-chain consensus mechanism will be the future research direction.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation (Grant nos. 61772099, 61772098, and 61802039); the Science and Technology Innovation Leadership Support Program of Chongqing (Grant no. CSTCCXLJRC201917); the Innovation and Entrepreneurship Demonstration Team Cultivation Plan of Chongqing (Grant no. CSTC2017kjrc-cxcytd0063); and Chongqing Research Program of Basic Research and Frontier Technology (Grant no. cstc2018jcyjAX0617).

Supplementary Materials

Supplementary Figure 1: The process of PBFT algorithm; it is a process of the traditional PBFT algorithm, which contains five steps: request, pre-prepare, prepare, commit, and reply. Supplementary Figure 2: The process of VS-PBFT; it is a process of the proposed algorithm, which contains three steps: node partition, primary node selection, and global consensus. After the three steps, the transactions are updated to the blockchain. Supplementary Figure 3: Partition diagram; we divide the entire circular network into three areas, and obtain the area where the node is located by calculating the node IP. Supplementary Figure 4: The process of primary node selection, after this primary node selection, we will get k local primary node. Supplementary Figure 5: Network topology diagram; it is the network topology after the primary node selection, the three local primary node will run global consensus to find a global primary node. Supplementary Figure 6: The process of simplified PBFT; we cut the pre-prepare step to reduce the communication times. . (*Supplementary Materials*)

References

[1] J. Leng, S. Ye, M. Zhou, J. Leon Zhao, Q. L. Wei, and L. fu, "Blockchain-secured smart manufacturing in industry 4.0: a survey," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 237–252, 2021.

[2] V. B. Thurner, "Blockchain and the future of energy," *Technology in Society*, vol. 57, pp. 38–45, 2019.

[3] S. Nakamoto, "A peer-to-peer electronic cashsystem," 2008, https://bitcoins.info/bitcoin.pdf.

[4] M. U. Hassan, M. H. Rehmani, and J. Chen, "Deal: differentially private auction for blockchain-based microgrids energy trading," *IEEE Transactions on Services Computing*, vol. 13, pp. 263–275, 2020.

[5] W. Z. Feng, "A hybrid cryptography scheme for nilm data security," *Electronics*, vol. 9, pp. 11–28, 2020.

[6] J. Leng, P. Jiang, K. Xu et al., "Makerchain: a blockchain with chemical signature for self-organizing process in social manufacturing," *Journal of Cleaner Production*, vol. 234, pp. 767–778, 2019.

[7] X. Fu, H. Wang, and P. Shi, "A survey of blockchain consensus algorithms: mechanism, design and applications," *Science China Information Sciences*, vol. 64, no. 2, pp. 1–15, 2021.

[8] C. Qin, B. Guo, Y. Shen, T. Li, Y. Zhang, and Z. Zhang, "A Secure and Effective Construction Scheme for Blockchain Networks," *Security and Communication Networks*, vol. 2020, Article ID 8881881, 20 pages, 2020.

[9] N. Szabo, "Smart contracts:building blocks for digital markets," *Extropy:The Journal of Trahumanist Thought*, vol. 56, p. 16, 1994.

[10] T. Dickerson, P. Gazzillo, M. Herlihy, and E. Koskinen, "Adding concurrency to smart contracts," *Distributed Computing*, vol. 33, no. 3, pp. 209–225, 2020.

[11] A. B. Alberto, L. Alfio, M. Giacomo, and Q. Salvatore, "On the use of blockchain technologies in wifi networks," *Computer Networks*, vol. 162106855 pages, 2019.

[12] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2018.

[13] S. Kudva, S. Badsha, S. Sengupta, I. Khalil, and A. Zomaya, "Towards secure and practical consensus for blockchain based VANET," *Information Sciences*, vol. 545, pp. 170–187, 2021.

[14] X. Guangxia, L. P. Yong, and K. Waqas, "Improvement of the dpos consensus mechanism in blockchain based on vague sets," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4252–4259, 2020.

[15] C. Miguel and L. Barbara, "Practical byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[16] D Ongaro and J Ousterhout, "In search of an understandable consensus algorithm," in *Proceedings of the 2014 USENIX Annual Technical Conference (Usenix ATC 14)*, pp. 305–319, Philadelphia, PA, USA, 2014.

[17] C. Dwork and M. Naor, "Pricing via processing or combatting junk mail," in *Advances in Cryptology—CRYPTO' 92. CRYPTO 1992*, E. F. Brickell, Ed., Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, 1993.

[18] E Androulaki, A Barger, V Bortnikov et al., "Hyperledger Fabric: A Distributed Operating System for Permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, New York, NY, USA, 2018.

[19] A. Clement, E. L. Wong, and L. Alvisiet al, "Making byzantine fault tolerant systems tolerate byzantine faults," *6th USENIX Symposium on Networked Systems Design and Implementation*, vol. 18, pp. 153–168, 2009.

[20] G. G. A. G. M. Pinkas, "A hybrid cryptography scheme for nilm data security, 2019 49th," *Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, vol. 69, pp. 568–580, 2019.

[21] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965.

[22] W.-L. Gau and D. J. Buehrer, "Vague sets," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 2, pp. 610–614, 1993.

[23] L. Yong, G. Wang, and F. Lin, "A general model for transforming vague sets into fuzzy sets," *Transactions on computational science II*, vol. 1, Article ID 133144, 2008.

[24] J. Leng, M. Zhou, J. L. Zhao, Y. Huang, and Y. Bian, "Blockchain security: a survey of techniques and research directions," *IEEE Transactions on Services Computing*, vol. 11 page, 2021.