DTNB: A Blockchain Transaction Framework With Discrete Token Negotiation for the Delay Tolerant Network

Xin Cong, Lingling Zi, and Ding-Zhu Du

***Abstract*—The current blockchain deployment solutions rely on a continuous connectivity network. Unfortunately, the delay tolerant network does not meet this condition. Therefore, we construct a novel blockchain transaction framework with discrete token negotiation called DTNB, which can be deployed on the delay tolerant network to provide transaction services. Specifically, we present the structure of add-chains by modifying the existing block structure. Then, we design a mining qualification determining scheme to achieve fair transactions, including discrete token generation algorithm and mining qualification attribution algorithm, and this scheme avoids the problem that nodes with more stakes in the PoS and DPoS algorithms can obtain mining qualifications with a higher probability. Furthermore, we present two mining schemes and also design a fork processing algorithm, which ensures that blocks on the add-chain generated by the local network can be appended to the main chain with the equal probability. Finally, we design the second consensus algorithm to avoid the problem of false and repeated transactions of the blocks on the add-chains in the local network. Theoretical analysis shows three properties of DTNB, including safety, reliability and activeness, and the experimental simulations demonstrate DTNB has advantages in throughput, block generation time and fork rate.**

***Index Terms*—Blockchain, delay tolerant network, discrete token, negotiation mechanism.**

I. Introduction

THE Internet is becoming the basic requirement of people. According to the data on global internet usage in 2019 [1], about 53.6% of world population have access to the Internet and accept network services, of which 14.5% users adopt the fixed broadband and 83% users adopt the mobile broadband. However, there are still 6.71 billion people who cannot enjoy the convenience and swiftness of the Internet due to various restrictions. One of the restrictions is caused by people's geographic location, such as living in remote mountains and islands, driving in vehicles far from the city. These unfavorable geographical locations make it difficult for Internet operators to provide network access services for the reason that the investment of infrastructure construction and operating revenue are disproportionate. Currently, a feasible way for these locations is to utilize opportunistic connectivity of ad-hoc networks and delay tolerant networks, which can have non-continuous connectivity access services. However, the quality of services is limited and cannot meet the requirements in scenarios requiring large amounts of real-time data exchange. For example, it cannot support transaction activities in the blockchain network tempted by transaction security and virtual currency appreciation.

Blockchain is the latest distributed transaction system, which changes the current transaction system with trusted third parties as the core [2] (such as credit cards, shopping website platforms, etc.), so non-trusted users can directly conduct the exchange of virtual currency without worrying about fraud and security issues. The core advantage of blockchain is decentralization and by using data encryption, time stamping, distributed consensus and economic incentives, it can effectively remedy the shortcomings of the current transaction process, such as high costs, inefficiencies and data storage insecurity. Therefore, blockchain has attracted a lot of interest from academia and industry, and some scholars have preliminarily proposed solutions for applying blockchain to emerging fields like 5G [3], social networks [4], Internet of Things [5] and artificial intelligence [6]. However, based on the fact that the essence of blockchain is the competition of mineral rights between nodes (such as computing power, bandwidth, etc.), compared with the continuous connectivity network, the nodes with delay in the delay tolerant network will be at a disadvantage in transaction activities, thus losing the possibility of gaining revenue in the blockchain network. So, it is important to construct a blockchain transaction framework that supports the delay tolerant network and fulfills a goal of deploying blockchain in a non-continuous connectivity environment.

The current blockchain can only be deployed on a continuous connectivity network, hence throwing a big challenge to the blockchain deployment on the non-continuous connectivity network. For example, how to solve the problem of

Manuscript received November 16, 2020; revised December 30, 2020 and January 28, 2021; accepted March 6, 2021. Date of publication March 9, 2021; date of current version July 7, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61602227 and Grant 61702241, in part by China Scholarship Council, and in part by National Science Foundation under Grant 1907472. Recommended for acceptance by Dr. Yulei Wu. *(Corresponding author: Lingling Zi.)*

Xin Cong and Lingling Zi are with the School of Electronic, and Information Engineering, Liaoning Technical University, Huludao 125105, China, and also with the Department of Computer Science, University of Texas at Dallas, TX 75080 USA (e-mail: chongzi610@163.com; lingling19812004@126.com).

Ding-Zhu Du is with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080 USA (e-mail: dzdu@utdallas.edu).

Digital Object Identifier 10.1109/TNSE.2021.3065058

appending the blocks generated under network interruption to the main chain. The existing studies [7], [8] on the lighting networks and payment channels may provide available technologies to solve this problem. However, two factors limit the application of these technologies in the delay tolerant network. The first is that both parties must stay connected at all times, which is impossible in the delay tolerant network. The last is that the establishment of a transaction must be approved by both parties, which is unrealistic due to the non-continuous connectivity of the delay tolerant network. So in this paper, we overcome this challenge by constructing a blockchain transaction framework that supports the delay tolerant network, which contains two aspects: delay tolerant network connection and blockchain operation. Delay tolerant network connection technology takes community-run base stations as the core and satellites as the relay to connect nodes in remote geographical locations to the Internet, examples are Nokia Kuha base station [9], Telstra small cells [10], Huawei RuralStar [11], etc. This technology has the characteristics of periodic network connection interruption. On the other hand, for achieving blockchain operation in non-continuous connectivity network, it is necessary to improve the current blockchain system, including designing to append legally the blocks generated under network interruption to the main chain when the network is reconnected, modifying existing mining mechanism to adapt to the nodes with lower computing power while reducing the consumption of mining costs. Therefore, we propose a novel blockchain transaction framework, named DTNB, by innovating a newly designed blockchain to conduct blockchain transaction in the delay tolerant network. To be specific, the contributions of our paper can be summarized as follows:

1) We construct DTNB, a novel blockchain transaction framework for the delay tolerant network, where the nodes perform different blockchain activities in the states of network connection and network non-connection.
2) For deploying blockchain in the delay tolerant network, we present the structure of add-chains by modifying the existing block structure, that is, adding the identifier field. Also we present the method of accommodating add-chains on the blocks of the main chain, improving the scalability of blockchain.
3) To achieve fair blockchain transaction in DTNB, we design a mining qualification determining scheme which has the characteristics of security and uniqueness. This scheme includes discrete token generation algorithm and mining qualification attribution algorithm based on discrete token negotiation.
4) According to the characteristics of the delay tolerant network, we design two mining schemes for DTNB and present a fork processing algorithm for add-chains based on the principle of balancing maximum computing power and fairness.
5) In DTNB, we design two types of consensus and present their corresponding algorithms, including the consensus algorithm for the generated blocks and the second consensus algorithm for preventing transactions from being repeatedly processed or false transactions.
6) We analyze the performance of DTNB. To be specific, we theoretically prove safety, reliability and activeness of the proposed framework. Also we show the effectiveness of DTNB from simulation results.

The remaining part of this paper is structured as follows: Section II gives a brief review on the related work. Section III describes the proposed DTNB framework in detail. Section IV shows theoretical analysis and experimental simulations. Finally, Section V concludes this paper.

II. Related Work

Currently, the application of blockchain in combing with emerging fields has gradually become a research hotspot in academia; however, the premise of application is that blockchain can be deployed on the continuous connectivity network, and thereby it is rarely involved in non-continuous connectivity networks such as the delay tolerant network. Aiming at constructing a blockchain transaction framework for the delay tolerant network, we introduce the closely related technologies, including network connectivity, block mining and blockchain scalability.

Network connectivity technology. Delay tolerant network is one of the solutions for users in remote mountainous areas or islands to connect to the Internet, using satellites and other equipment to provide periodic network connectivity services with a delay. Pentland uses distributed mobile coverage technology to provide data services to northern Cambodia [12]. Blattman builds a community network with the core of the delay tolerant network to meet the needs of network access in remote areas of India [13]. Software defined networking (SDN) is one of the technologies for handling the data transactions between devices. It mainly divides the network into a control layer and data layer, which makes the network configuration and protocol deployment easy and flexible. Miao *et al.* [14] presented an analytical model to investigate the performance of SDN when the data arrives in bursts and correlations. Some papers such as [15]–[17] utilize the the blockchain technology to solve the security of SDN. However, how to deploy the blockchain to a delay tolerant network with SDN is still an issue to be solved. Besides, Hu *et al.* [18] presented a blockchain-based delay-tolerant payment scheme, focusing on deploying multiple proxy nodes connected to the remote communities. They introduced the smart contracts for payment service management, including user account initiation, interactions with credit operator and rewards for miners. Then a data upload method was presented to transfer data from remote areas to Internet. This scheme detailed calculates the relationship between the size of the generated block and the upload time, and designs the calculation method of the connection number of neighbors. But the adaptability of blockchain in the delay tolerant network is not given a detailed description. Therefore, how to design a general blockchain model deployed on the delay tolerant network is still an open issue.

Block mining technology. Block mining is one of the key technologies of blockchain, also known as block generation algorithm. The representative algorithms include Proof of Work (PoW), Proof of Stake (PoS), Proof of Vote (PoV), etc. PoW first appeared in the paper by Satoshi Nakamoto, which adopts a hash algorithm to repeatedly calculate until a random number that satisfies the threshold is found. Therefore, it takes a lot of CPU or GPU computing power to generate a new block. Moreover, blocks are combined into a chain structure in a linear manner [19]. The longer the length of the chain, the more computing power it has, and the more difficult it is to tamper with. As the first block consensus algorithm, the disadvantage of PoW is that a large amount of computing power is wasted when choosing a block generator, so the cost is relatively large. In order to solve the problem of a waste of computing power, PoS was proposed, which determines the block generator by allocating voting rights in proportion to each node, thus reducing the calculation number of the hash function. PeerCoin [20] is the first blockchain system to deploy PoS, in which nodes allocate stakes according to the amount of locked assets, the committee uses the encryption algorithm to calculate a specific hash value, and the set of nodes with the hash value is used as the block generator. Then the committee determines whether receiving the block. However, the advent of the committee alleviates the impact of distributed feature of blockchain. PoV is used in the consortium blockchain system [21] and it leverages the central node to control other nodes, gives network participants different security identities, confirms the block generator by voting and performs verification confirmation for new blocks, thus avoiding the third party arbitration and the existence of uncontrollable public nodes. Its advantage is to improve the throughput, while the disadvantage is the relatively poor reliability. Fastchain [22] increases the throughput of blockchain by reducing the block propagation time. The miners in Fastchain choose nodes with the higher bandwidth as their neighbors and upload block data as quickly as possible. The directed acyclic graph(DAG) mechanism is introduced in [23]–[26]. Some weakness may affect the deployment on the delay tolerant network. For example, it is easy to combine 33% of the nodes to control the entire blockchain network. And the blockchain with the DAG structure cannot avoid repeated transactions, and requires that the two nodes that built the channel are online in real time. In summary, the above blockchains are designed for the scenario of continuous connectivity network, but our task is to mine blocks under the non-continuous connectivity network. So new algorithms are urgently needed to accomplish this task.

Blockchain scalability technology. Scalability is one of research directions of blockchain, of which the goal is to achieve the coexistence of multiple applications and services and the ability of multiple blockchains to communicate and exchange data with each other. Researchers have proposed some schemes to address the problem of scalability. Chen *et al.* [27] divided the blockchain network into multiple groups (called shards) and then sharding can work on disjoint transactions, preform blockchain operation in parallel and maintain independent ledgers. Chatzopoulos *et al.* [28] presented a DAG-based distributed ledger to suit for the mobile devices in device-to-device ecosystems, in which two consensus protocals named Proof-of-Context and Proof-of-Equivalence were designed. The former is used for adding data according to the users' context and the latter is utilized to decrease the requirement of storage of nodes. Poon *et al.* [29] proposed a bitcoin lightning network, in which both parties create a channel on the main chain and place transaction fees in shared multisignature addresses. Then, the conducted transactions on the channel can detach the main chain, without waiting time. In addition, the sidechain technology is an effective way to achieve scalability [30] and its core idea is as follows. Before the transaction, Bitcoin is transferred to a dedicated account and is in a frozen state, which is equivalent to creating a currency in a blockchain, and the currency can be used in other blockchains that recognize it through the account. After the transaction, the remaining frozen currency will be released and available. The same with real currencies, it is necessary to control the exchange rate of currencies according to the price changes. In our scenario, the nodes in the delay tolerant network need to participate in the blockchain activities, which requires a lot of data exchange in both states of network connection and disconnection. However, the existing methods cannot meet the application requirements of this situation, so it is necessary to explore new schemes for blockchain scalability.

III. The Proposed DTNB Framework

For users in remote geographical locations such as mountains and islands, one of the cost-effective ways to access the Internet is to construct a delay tolerant network that uses satellite as a relay. On this basis, the blockchain transaction framework in the non-continuous connectivity network is deployed to provide services for nodes to participate in blockchain activities.

A. The Blockchain Transaction Model in the Delay Tolerant Network

Considering that the network operators have established a complete infrastructure for network access in urban areas, they can provide continuous high-quality network services. Meanwhile, the infrastructure in remote areas (such as highways, mountain and islands) is relatively weak or not established, we design a delay tolerant network with blockchain deployment, shown in Fig. 1.

In Fig. 1, users' locations can be categorized into two types: urban areas and remote areas. Users located in urban areas can continuously access the Internet to participate in blockchain activities, and users located in remote areas can periodically access the Internet through satellites to participate in blockchain activities. Assume that the entire network is denoted as EN, containing the network in both urban areas and remote areas. The network in the remote area, which is the local network, is denoted as NU, $NU = \bigcup_{i \in U} Nu_i$, where U represents a natural number and Nu_i represents the i-th remote area. In each remote area, it includes nodes and operator

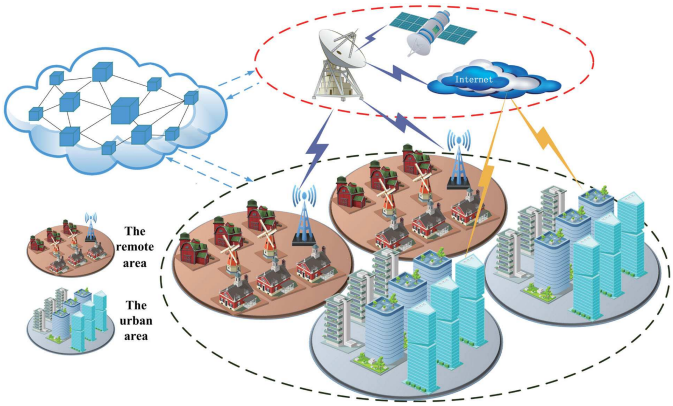


Fig. 1. The composition of a delay tolerant network with blockchain deployment.

servers and the nodes keep connected to each other in a peer-to-peer network (P2P) manner to construct a delay tolerant network. Based on this, we deploy a blockchain system, containing two types of nodes: lightweight nodes and full nodes. To be specific, lightweight nodes are represented by mobile phones and characterized by the low computing power, storage and bandwidth capabilities, which can only initiate transactions in blockchain. Full nodes are represented by computers and servers, and characterized by high computing power, storage and bandwidth capabilities, which can participate in all blockchain activities such as block packaging, block generation and block consensus.

Due to the characteristics of network non-connectivity or delay in the delay tolerant network, the blockchain transactions can be classified according to the location of the transaction initiators, shown as follows:

The initiators' transaction in the remote area: it means the newly received transactions under the network non-connection state, denoted as NUT.

The initiators' transaction at any location: it means the newly received transactions under the network connection state, denoted as ALT.

Assume that the node in Nu_i records the time when the network is disconnected, denoted as t_{discon}, and the time when the network is reconnected is denoted as t_{recon}. When $t \in [t_{discon}, t_{recon}]$, the nodes in Nu_i form a closed P2P local network, which can independently perform the blockchain system. At this time, the transaction messages received by the nodes belong to NUT and these transactions need to be placed in the newly generated block before t_{recon} as much as possible.

The proposed DTNB framework is demonstrated in Fig. 2, which contains two stages: network connection stage(CS) and network non-connection stage(NCS), in which the criterion is whether to access the Internet. When time $t \in [t_{discon}, t_{recon}]$, Nu_i is in the NCS and at this moment, the mining nodes can choose the following activities. (1) These nodes can continue the mining activity before t_{discon}. The advantage is that if the duration of NCS is very short, they can continue to participate in the block consensus of EN to obtain block generation revenue and transaction fees when the network is reconnected. However, the disadvantage is that if the duration of NCS is very long, the generated block has been conducted the consensus by EN and no revenue can be obtained when the network resumes connection. (2) These nodes can immediately give up the current mining activity, terminate the update of ledger in the local blockchain, record the hash value of the last block of blockchain, package the transaction records with timestamp after t_{discon} into blocks, and carry out mining and consensus. All the above activities are performed when Nu_i is in the NCS.

When time $t \geq t_{recon}$, Nu_i is in the CS, and the nodes can perform the following activities. (1) They download the latest blockchain from other nodes in the EN and maintain the local ledger to keep it up to date. (2) They broadcast the blocks generated by Nu_i to the EN for the second consensus, and append the blocks after the consensus to the local ledger. (3) They broadcast the unprocessed transaction records in Nu_i within $[t_{discon}, t_{recon}]$ to the EN, which is as newly generated transaction records.

Combining Fig. 1 and Fig. 2, the transaction process of DTNB can be briefly described as follows. First, lightweight nodes and full nodes can be used as transaction initiators to generate transaction records and send them to the mining pool. Second, full nodes apply for mining qualifications from decision-making institutions using discrete tokens (see Section III-C), then the qualified miners select the transaction records from the mining pool and package them, and then use a hash algorithm to generate blocks in a competitive manner, and submit the generated blocks to the decision-making institutions (see Section III-D). Finally, the decision-making institutions determine the legitimacy of the generated blocks, and append the new blocks to blockchain (see Section III-E). The above description is applicable to the two stages, including CS and NCS, but when the network is reconnected, it is necessary to broadcast the blocks generated by the local networks to the EN for the second consensus (see Section III-E). For easy presentation, Table I summarizes the notations used in the proposed DTNB framework.

B. Block Structure

The advantage of block structure is that data can be stored on each node in a distributed chain while maintaining consistency. However, the current block structure greatly limits the scalability of blockchain and cannot meet the diverse application demands. For example, we need to append the generated blocks in the NCS to the main chain in our scenario. So we modify the existing block structure by presenting the add-chain in blockchain, as illustrated in Fig. 3.

In Fig. 3, we show the existing data field through a rectangular box with a white background. To be specific, A is the block size with 4 bytes, B is the block header with 80 bytes, C is the transaction counter with 1-9 bytes, and D is the transaction record with variable bytes. B includes the following contents: the version number E with 4 bytes, the parent hash value F with 32 bytes, the mining difficulty G with 4 bytes, the timestamp H with 4 bytes, the nonce I with 4 bytes, and the hash value of the root node of the Merkle Tree J with 32 bytes.

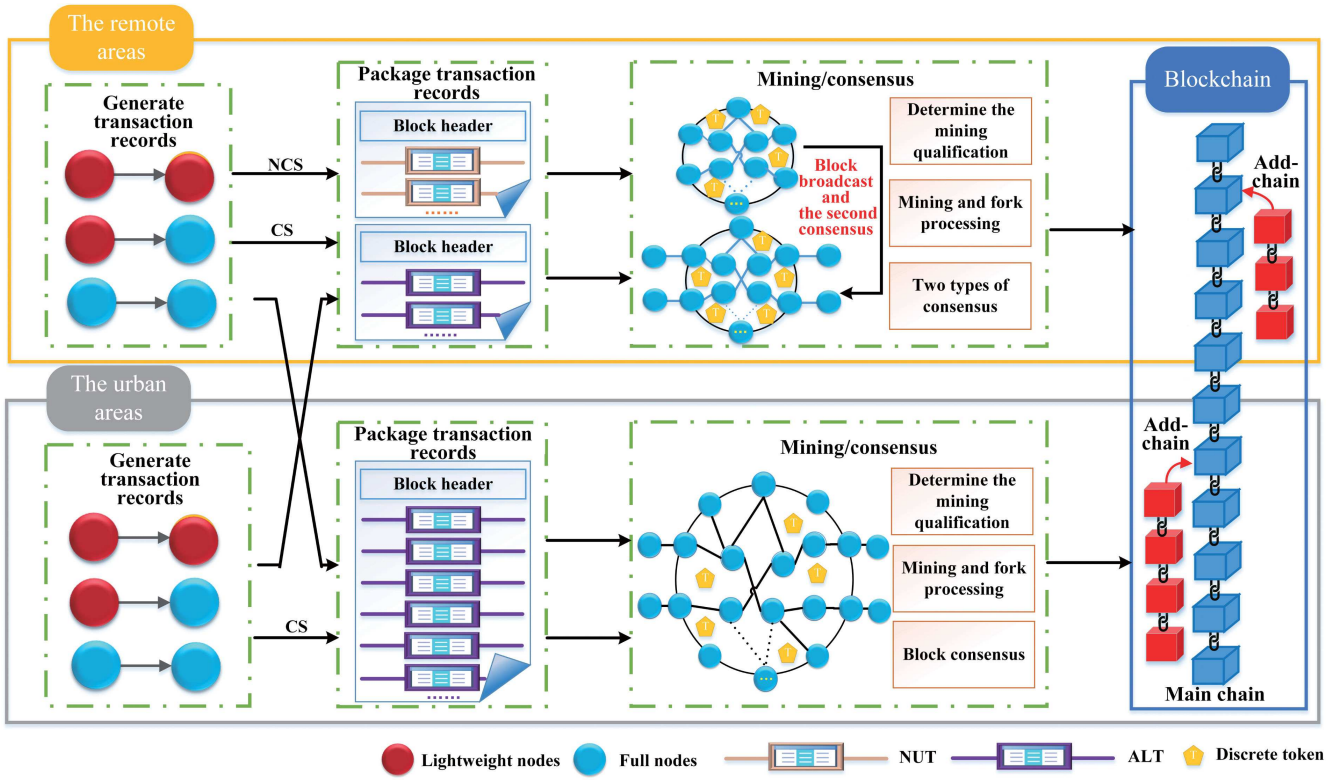


Fig. 2. The proposed DTNB framework.

TABLE I
NOTATIONS

Notation	Description
EN	The entire network
NU	The network in the remote areas, which is the local network
NUT	The initiators' transaction in the remote areas
ALT	The initiators' transaction at any location
t_{discon}	The time when the network is disconnected
t_{recon}	The time when the network is reconnected
CS	Network connection stage of DTNB
NCS	Network non-connection stage of DTNB
ID	The hash identification code of the miner
$DToken$	The discrete token for DTNB
HV	The hash value of the string formed in the process of mining qualification attribution

The implementation of the add-chain is shown below. We add the identifier field with 1 B, marked with X, in which 0 denotes the block on the main chain and 1 denotes the block on the add-chain. The added data field is marked through the rectangular box with a blue background in Fig. 3 and the add-chain is shown in the red square on the right side of Fig. 2.

C. The Mining Qualification Determining Scheme

Since the nodes are in the local network when the network is in the NCS, compared with EN, these nodes have limited computing power, which is difficult to generate blocks with the mining difficulty of EN within a certain period of time (the PoW way). In order to reduce the mining difficulty and improve the computing power, the mining

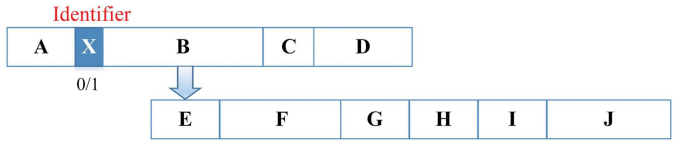


Fig. 3. Block structure.

qualification determining scheme for nodes in the DTNB framework is designed.

The core of this scheme is the discrete token and its role is to uniquely determine the identity of the miner in the process of applying for mining qualifications, which can not be easily tampered with by other miners. And the discrete token is generated with low computing power and low latency, which does not depend on the central nodes. Moreover, mining qualifications are determined by the decision-making institution. The members of the institution are dynamically generated during each round of mining, including M managers and N leaders negotiated by these managers, in which $N \approx \frac{M}{2}$. The main steps of the scheme are shown in Fig. 4. First, the miners apply to the decision-making institution for mining qualification, that is, discrete token generation (step 1). Second, the managers negotiate to determine the leaders, and unify applicants (step 2). Then leaders negotiate to determine the mining qualification attribution (step 3). Last, they send the negotiation results to the miners and broadcast to the nodes in the EN (step 4 and step 4'). The detailed implementation is shown as follows.

Discrete Token Generation: The discrete token takes the miner ID, which is assigned when a miner enters the blockchain, and a

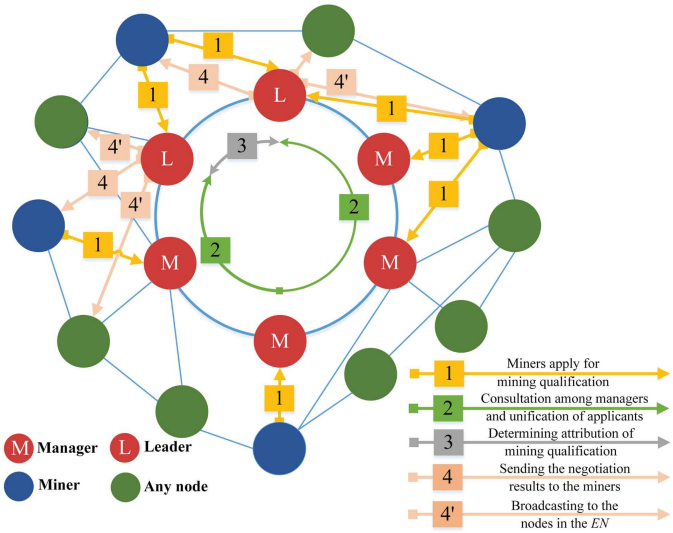


Fig. 4. The main steps of mining qualification determining scheme.

random number as input data, and uses a hash function to form a 256-bit string. Assume that the hash identification code of the miner is denoted as ID and the random number is denoted as $random$, the discrete token $DToken$ is calculated as:

$$DToken = f(ID, random) \tag{1}$$

In order to protect the privacy of miners, ID needs to be encrypted by a hash function, that is:

$$ID = Hash(ID) \tag{2}$$

$Hash()$ can be used many times to encrypt ID.

The random number is generated by the programming language, in which the number of bits is 16 (the random number generated by the random function in the mainstream programming language), the generated quantity is K, and it is not unique. To identify the unique of the random number, the kth random number generated by ID, i.e. $random_k$ is

$$random_k = Hash(ID||random_k) \tag{3}$$

Where $random_k = random()$, $random()$ is a random function in the programming language, $||$ denotes the concatenation of strings, i.e.

$$random = random_1||random_2|| \cdots ||random_K \tag{4}$$

Where $random_1$, $random_2, \ldots,$ $random_k$ are ranked by strings in the non-descending order.

According to 1 to 4, the generation function of $DToken$ is:

$$DToken = Hash(ID||random) \tag{5}$$

We present the $DToken$ generation algorithm, shown in Algorithm 1.

Mining Qualification Attribution: The miner sends the generated discrete token to one or more managers as a certificate for applying for the current mining qualification. In this case,

Algorithm 1: The discrete token generation algorithm.

Input: ID.
Output: $DToken$.
1: New blocks have been generated and appended to the local blockchain after consensus is reached.
2: $ID = Hash(ID)$.
3: Generate M random numbers, i.e. $random_1$ to $random_K$ and rank them in the non-descending order.
4: **for** ($k = 1$ to K) **do**
5: $random_k = Hash(ID||random_k)$.
6: **end for**
7: $random = random_1||random_2|| \cdots ||random_K$.
8: $DToken$=$Hash(ID||random)$.
9: Return $DToken$.

even if a single or a limited number of managers (less than half of managers) do not forward messages, the honest managers will forward discrete tokens to other managers. So, the application of one miner can always be processed. Then M managers send the received discrete tokens to the N leaders for negotiation. After removing the repeated discrete tokens, the leaders connect them in the non-descending order to form a string, and use the hash function to calculate the hash value of this string, denoted as HV. HV is sent to other leaders and if it can obtain the consent of more than half of the leaders, then HV will be used as the benchmark for the current mining qualification. The leader with HV takes the discrete tokens satisfying 1) $\lfloor M2 \rfloor$ discrete tokens with the value of the closest and less than or equal to HV and 2) $\lceil M2 \rceil$ discrete tokens with the value of the closest and greater than HV, as the mining qualification granted in this round. After N leaders negotiate and unify the above mining qualifications, they will broadcast the mining qualifications to the EN. Refer to Algorithm 2 for details of the mining qualification attributable based on discrete token negotiation.

After confirming the $DToken$ set as the current round of mining qualification, the leaders will broadcast the $DToken$ set to the EN. Each node compares the $DToken$ generated by itself with the contents of the $DToken$ set. If it belongs to the set, it is determined to have the mining qualification of the current round. At this time, it will send its own $DToken$ and IP address to the leaders (or change the source address of the message receiving the $DToken$ set to the destination address). The leaders record these IP addresses as the next round of managers and broadcast them to the EN. If it is found that the same $DToken$ corresponds to multiple IP addresses, it may be a forgery of the malicious node. At this time, the owner of each IP address is required to provide both ID and the random value for generating $DToken$, as shown in Eq.5.

Leader Negotiation Method: In order to avoid the association of the manager's IP and the hash identification code of miner, it is recommended that the managers not apply for the current mining qualification. The managers are the miners with the mining qualification in the previous round, and the IPs are released to the EN by the managers in the previous round. Each manager sends its own IP address to other managers, and uses a hash function to convert the collected IP

Algorithm 2: Mining qualification attribution algorithm based on discrete token negotiation.

Input: $DToken$ for each node applying for qualification.
Output: The qualified $DToken$ set.

1: Timing starts when the new block is appended to blockchain and the duration is T.
2: Within time T, the node applying for qualification sends its own $DToken$ to one or more managers. //avoiding losing the application qualification due to the manager is a malicious node.
3: The manage obtains the hash value of the stored IPs of other $M-1$ managers in the current round through a hash function. //the managers negotiate to elect the leaders.
4: Rank the hash values of IP addresses in ascending order, and select the top N IP addresses as the leaders.
5: Send the selected N IP addresses to other managers.
6: **for** ($k = 1$ to $M-1$) **do**
7: The manager compares with the N IP addresses selected by calculation.
8: **if** (IP address matches) **then**
9: $count++$.
10: **end if**
11: **end for**
12: **if** ($count$ is greater than $\lceil M2 \rceil$) **then**
13: The leaders negotiate successfully.
14: **else**
15: The manager sends its own IP address to other managers, GOTO 2.
16: **end if**
17: Each manager sends the received $DToken$ within time T to the leaders. //Negotiate the qualifications of applicants
18: The leaders eliminate the duplicate $DToken$ and rank $DToken$ in the non-descending order.
19: Connect $DToken$ and use the hash function to get HV.
20: Send HV to other leaders.
21: **for** ($k = 1$ to $N-1$) **do**
22: Compare HV with the HV sent by other leaders.
23: **if** (same) **then**
24: $count++$.
25: **end if**
26: **end for**
27: **if** (count++ $>$ $\lceil N2 \rceil$) **then**
28: Successful negotiate and take HV as the benchmark for this round.
29: **end if**
30: The leader with HV uses the $DToken$ satisfying 1) $\lfloor M2 \rfloor$ DToken with the value of the closest and less than or equal to HV and 2) $\lceil M2 \rceil$ $DToken$ with the value of the closest and greater than HV, ranks $DToken$ in the non-descending order and uses a hash function to compute the hash value. Then the obtained hash value is sent to other leaders.
31: **if** (the hash value is approved by the $\lceil N2 \rceil$ leaders) **then**
32: Return the qualified $DToken$ set.
33: **else**
34: GOTO 2.
35: **end if**

addresses into hash values. Then they are ranked by the hash values in ascending order, and take the first N hash values, in which their owners are the leaders of the current round.

Manager Initialization Method: When a new network operator node in the remote area that uses satellite as Internet access joins the delay tolerant network, its local network needs to identify the managers who supports blockchain activities when the network is interrupted for the first time. Since the IP address of the network operator node is known, the nodes in the local network can apply to the network operators for manager qualification, similar to the approach of mining qualification. Each applied node generates a discrete token $DToken$ using Algorithm 1 and sends it to the network operator node. After ranking the $DToken$ in ascending order, the node uses a hash function to generate a hash code HV, and takes the owners of the $DToken$ satisfying both 1) $\lfloor M2 \rfloor$ $DToken$ with the value of the closest and less than or equal to HV and 2) $\lceil M2 \rceil$ $DToken$ with the value of the closest and greater than HV, as the current round of managers.

It should be noted that after a miner becomes a manager, other miners will know the IP address. In this case, miners may suffer from Dos or DDos attacks launched by malicious nodes, the main purpose of which is to obtain revenue of virtual currency. Some existing studies [31], [32] can be used to detect these attacks and minimize the damage. However, in our scenario, these malicious nodes will not be able to obtain certain revenue due to the following reasons. 1) In the process of determining the managers, the difference between the input data leads to great changes and uncertainties for the miners who become managers. 2) As the input data changes, the addchain that can be appended to the blockchain also changes greatly, which makes the miners who can obtain virtual currency uncertain and unpredictable. Therefore, we pay less attention to defending against these two attacks in this paper.

In summary, the election process of managers is as follows. In the initial stage of the blockchain network (the network is disconnected for the first time), there are no managers in the network. At this moment, the miner broadcasts the hash value of the local IP address encrypted by the hash function to other miners. After a certain period of time, the miner ranks his own and the received hash values in the non-descending order to form a string, uses the hash function to calculate the hash value HV and then broadcasts it to other miners. When the times of receiving a certain HV reaches a preset threshold, the miner uses this HV as a benchmark and selects the owners who meet $\lfloor M2 \rfloor$ strings forward and $\lceil M2 \rceil$ strings backward as the current round of managers. In the non-initial stage, the miner uses Eq.5 to generate the discrete token, and sends it to the managers. Each manager ranks the received discrete tokens in the non-descending order to form a string, uses the hash function to calculate the hash value HV and then broadcasts it to other managers. When the times of receiving a certain HV reaches a preset threshold, the miner uses this HV as a benchmark and selects the owners who meet $\lfloor M2 \rfloor$ strings forward and $\lceil M2 \rceil$ strings backward as the next round of managers.

D. Mining and Fork Processing

After the node knows that it has the mining qualification in the current round, it needs to perform the mining process

within a specified time period to obtain revenue. The goal of node mining is to generate blocks according to the data structure shown in Fig. 3. The mining process is divided into two steps, block transaction package and hash value calculation. Block transaction package is to select transaction records with higher transaction processing fees from the transaction pool and place them in the blocks. Hash value calculation is based on the currently mining difficulty, and the nonce is tried to make the distance between the value of Merkel root of the block and target value of the mining within a threshold. In the environment of delay tolerant network, nodes in the CS adopt the scheme of mining in the entire network and nodes in the NCS adopt the scheme of mining in the local network. Two mining schemes fill in the different data of block structure in the process of transaction package and the details are shown as follows.

The Scheme of Mining in the Entire Network: In the block structure, X is set as 0. The miners package transaction records from the mining pool and perform the hash calculation with the current mining difficulty until reaching the target value. After the blocks have been reached the consensus, miners charge the block generation fee and transaction processing fee.

The Scheme of Mining in the Local Network: In the block structure, X is set as 1. The miners package transaction records and perform the hash calculation according to the mining difficulty set by the current local network until reaching the target value. Different from the first scheme, miners only charge the transaction processing fee.

When the network is connected, if the miner sets X to 1 in the process of packaging, the block is on the add-chain, which needs to conduct the consensus of the managers in the local network and the second consensus of managers in the EN. However the block on the add-chain packaged by the miner lacks the above consensus and will not be appended to the blockchain. When the network is disconnected, if the miner sets X to 0 in the process of packaging, the block has a great probability that it has been generated by the miner under the connected network due to a large network delay. In this case, the block cannot be appended to the blockchain.

Due to the existence of network delay, the forks will occur when generating the blocks. The basis of the current blockchain system for processing forks is to select the fork with the largest computing power as the main chain, that is, to select the longest chain. In the delay tolerant network, there are three possible forks: (1) when the network is disconnected, forks occur on the add-chain in the local network; (2) forks occur on the main chain; (3) forks occur on multiple add-chains generated by the same block on the main chain. (1) and (2) can be solved using the longest chain. (3) means that when the network is interrupted, two or more local networks record the hash value of the last block of the current blockchain, and use it as the parent hash value (i.e. F in the block structure) to mine in the local network and then append to the main chain when resuming connection. Compared with (1) and (2), (3) will not eliminate a fork due to the natural selection of blockchain nodes, so a relatively fair scheme needs to be designed to determine which forks or many forks are abandoned. In (3), the length of fork is related to the computing power in the local network and non-connection time.

Algorithm 3: The fork processing algorithm.

Input: Multiple add-chains with the same hash value of the parent block.

Output: The reserved add-chain.

1: If there exists the add-chains on the sixth block from the last of the main chain, managers will process it at this time.
2: According to the order of each add-chain connection, calculate the hash value of the block to which it belongs, and after connecting in sequence, calculate the final hash value as the hash value of the add-chain.
3: Rank the hash value of each add-chain in ascending order, connect them into a string, and calculate its HV.
4: Find the hash value of the add-chain closest to HV and take this add-chain as a candidate add-chain.
5: **if** (the difference between the number of blocks on the candidate add-chain and the number of blocks on the longest chain is within the threshold) **then**
6: Return the candidate add-chain.//Ensure the fairness of different local networks
7: **else**
8: Return the longest chain.//Make the computing power not wasted
9: **end if**

Generally speaking, the greater the computing power, the more blocks are generated within a certain time; the longer the non-connection time, the more blocks are generated. If the longest fork is reserved according to the current situation, the add-chain generated by the local network with a large computing power will always be retained, or the add-chain generated by the local network with the longer non-connection time will always be retained, which is extremely unfair for other local networks. According to the above analysis, on the basis of the idea of the longest fork, we design the fork processing algorithm for the add-chains, as shown in Algorithm 3.

It is worth noting that the longest chain principle is used in Algorithm 3 to process the fork, however, malicious miners with high hash power may actively disconnect from their networks and mine blocks using the low mining difficulty, thereby gaining benefits. So in order to overcome the shortcoming of always appending the add-chain to the blockchain due to the high hash power of a single EN, a fair algorithm (see Algorithm 3) is designed. Unless the selected add-chain is particularly short and then replaced by the longest add-chain, the longest add-chain has no advantage in the probability of being appended to the blockchain. Based on the above characteristics, the fork processing algorithm using the longest chain in the local network can satisfy the proposed blockchain transaction scheme. In addition, if the method of disconnecting the network is used to simulate the add-chain generated by the delay tolerant network, the probability of always being appended to the blockchain is extremely low due to the different input data.

E. Consensus and Appending

After the block is generated by the miner with mining qualification, it is necessary to conduct the consensus to verify the

Algorithm 4: The consensus algorithm.

Input: The generated block $Block$.
Output: When $Block$ is legal, return True, otherwise False.

Request the sender of the block to prove that it is the original data of the miner with mining qualification in this round, including ID in Eq.2 and $random_k$ in Eq.3.
if (Original data is received within the specified time) **then**
  Use Eq.4 and Eq.5 to generate $DToken$.
else
  Return False.
end if
Query the identification code set of the miners with the mining qualification in this round.
if ($Dtoken \notin$ the identification code set) **then**
  Return False.
else
  Generate Merkel root based on the mining difficulty.
  if (illegal) **then**
    Return False.
  end if
end if
if (X = 0)// Blocks generated during network connection conduct the consensus across the entire network **then**
  if ($Block.F \notin$ the hash value of blocks on the main chain) **then**
    Store the block to the local pending block set.
    Request the missing parent block from the sender of the block and wait.
  end if
else
  if ($Block.F \notin$ the hash value of blocks on the main chain) **then**
    if ($Block.F \notin$ the hash value of blocks on the add-chains) **then**
      Store the block to the local pending block set.
      Request the missing parent block from the sender of the block and wait.
    end if
  end if
end if
if ($Block.F =$ the hash value of the last block on the main chain) **then**
  Append blocks to the blockchain.
  Set the block as the last block on the main chain.
  Return True.
else if ($Block.F \in$ forks) **then**
  Link $Block$ to $Block.F$.
  if (Distance between $Block.F$ to the genesis block $=$ the length of main chain) **then**
    Set the fork belonging to $Block.F$ as the main chain.
    Delete other forks.
  end if
  Return True.
else
  Link $Block$ to $Block.F$ to be a fork.
  for ($p \in$ the pending block set) **do**
    if ($p.F = Block$) **then**
      Remove p from pending block set and GOTO 4.
    end if
  end for
  Return True.
end if

legitimacy of the block, and then append it to the blockchain. In DTNB, there are two types of consensus: (1) consensus of the generated blocks across the entire network and the local network; (2) the second consensus of the blocks generated by the local network across the entire network after the network is reconnected. The first one needs to check the legitimacy of the following items, including block generator and Merkel root. The last one needs to further check the legitimacy of transaction records. The first consensus algorithm is detailed in Algorithm 4. When the same block is approved by more than half of the leaders, it is appended to the blockchain.

Noting that in Algorithm 3 and 4, the incentive mechanism is considered. The miners adopt the principle of first service and then mining, of which the goal is to incentivize miners to provide services for the blockchain system, and also regulate the behaviors of miners, making them act honestly rather than maliciously. Miners need to complete the following services, including generating discrete tokens, determining the set of miners with mining qualifications for the next round, negotiating the next round of decision-making institution, broadcasting the mining qualifications and the address of the next round of receiving discrete tokens, verifying the legality of successfully mined blocks, and broadcasting the successful block information to append to the blockchain.

The purpose of the second type of consensus is to prevent transactions from being repeatedly processed or false transactions. When the network is disconnected, the miners in the local network are undergoing mining activities and may successfully generate blocks later. After the network is reconnected, the block is appended to the blockchain, but the transaction records contained in this block may have been completed by the miners in the non-local networks, resulting in repeated transactions. For example, user U_1 has 3 virtual coins and he or she trades 2 virtual coins with user U_2 when the network is connected, which is not appended to blockchain. In addition, he or she also trades 2 virtual coins with user U_3 when the network is disconnected. What may happen is that the first transaction is packaged and generated a block by the miner in the non-local network, in which the block is appended to blockchain. And the second transaction is packaged and generated to a block by the miner in the local network when the network is disconnected, and the generated block is also appended to blockchain. At this time, it will cause a false transaction. In order to prevent the above situation, a second consensus algorithm is designed, see Algorithm 5 for details.

Due to network interruption, the baseline time of the nodes in the blockchain is inconsistent, resulting the inconsistency of the blockchain stored by each node, which can be solved by the second consensus algorithm designed in our paper. The idea of the second consensus algorithm is that after determining the add-chain to be appended to the blockchain, the nodes in the EN check the transaction records of each block in detail to confirm whether the initiator of each transaction has enough virtual currency to pay for its transaction fee. This algorithm can prevent false and repeated transactions, and can remedy the defect of inconsistency of the blockchain caused by the network interruption. In addition, operator nodes are key nodes for accessing the Internet in the proposed DTNB

Algorithm 5: The second consensus algorithm.

Input: The generated block in the local network $Block$.
Output: True or False. // When $Block$ is legal, return True, otherwise False

When the network is reconnected, the managers in the local network request the neighbor nodes in the non-local network to inform the IP address of the managers in the EN.
The managers of EN receive the newly generated block set in the local network.
if (The number of different managers sending the same block set is greater than $\lceil M2 \rceil$) **then**
  if (The transaction records in the newly generated block set are included in the blocks of blockchain) **then**
    Abandon the block set.
    Return False.
  else
    if (The parent hash value of the first block of the block set = the block on the main chain) **then**
      Attach the block set to the main chain as an add-chain.
      Call Algorithm 3.
    else
      Abandon the block set.
      Return False.
    end if
  end if
else
  Wait for other managers to send the block set.
end if

framework. When the network is connected, they need to download the latest blockchain and this latest blockchain is used by the nodes in the local network, which are too late to update the blockchain in the state of network connection due to network speed and other factors. Meanwhile, operator nodes can also be responsible for the task of broadcasting the generated add-chains to the EN during the network interruption. This task is not necessary and only guarantees that the add-chains can be broadcast to the nodes in the EN for verification when the network is reconnected.

IV. Theoretical Analysis and Experimental Simulations

The proposed DTNB framework adopts the "applying-approving-verifying" way to achieve all transaction activities of blockchain. DTNB with the goal of blockchain deployment on the delay tolerant network will be evaluated from theoretical analysis and experimental simulations. For the theoretical analysis, we mainly prove three properties of DTNB, namely, safety, reliability and activeness. For the experimental simulations, we conduct experiments to test such attributes as throughput, block generation time and fork rate.

A. Theoretical Analysis

In this subsection, we analyze three properties of DTNB, thus theoretically prove the effectiveness of the proposed framework.

Lemma 1: Safety: The generated blocks are safe and legal under the condition that at least $\lfloor N2 \rfloor + 1$ leaders are non-malicious nodes.

Proof: Based on Algorithm 4, the generated blocks need to be verified by more than half of leaders in the process of block consensus. Suppose an illegal block can be appended to the blockchain, the block has been approved by at least $\lfloor N2 \rfloor + 1$ leaders. But in fact, the leader who is a non-malicious node will not pass the illegal block verification. So, the illegal block can be approved by $N - (\lfloor N2 \rfloor + 1) \leq \lfloor N2 \rfloor$ leaders, that is, at most half of the leaders' approval, contradicting the assumption. Therefore, when the number of leaders who are the non-malicious nodes is greater than or equal to $\lfloor N2 \rfloor + 1$, the generated blocks are safe and legal. ■

Lemma 2: Reliability: The leader team is reliable, that is, the probability of being controlled by the malicious nodes is low.

Proof: Based on Algorithm 2, managers come from all applicants, also known as the manager nodes. Suppose there are Q nodes applying to become managers in DTNB, in which R nodes are malicious nodes. Then, the leaders selected from the managers form a leader team, which is responsible mining qualification allocation and block consensus in the current round. Suppose there are K malicious nodes among M managers and the number of leaders is N. Based on this, we calculate the probability of being controlled by the malicious nodes.

First: The prerequisite for the leader team to be controlled by malicious nodes is that the number of malicious nodes accounts for more than 50% of the number of team nodes, that is, the number of malicious nodes in the team exceeds $\lfloor \frac{N}{2} + 1 \rfloor$. When the number of malicious nodes is equal to $\lfloor \frac{N}{2} + 1 \rfloor$, the probability can be calculated as:

$$P\left(X|X = \lfloor \frac{N}{2} + 1 \rfloor\right) = \frac{N!}{X!(N-X)!} \bullet \left(\frac{K}{M}\right)\left(\frac{K-1}{M-1}\right) \cdots \left(\frac{K-X}{M-X}\right) \bullet \left(1 - \frac{K}{M}\right)\left(1 - \frac{K-1}{M-1}\right) \cdots \left(1 - \frac{K-X}{M-X}\right) \tag{6}$$

In Eq.6, the hash value calculated by each sample through the hash function is distributed on the sample space with equal probability under the condition of a larger number of samples. Thus, for the initial selection, the probability of each malicious node becoming a leader is K/M, and the probability of a non-malicious node becoming a leader is $1 - K/M$. At this moment, the probability of a malicious node becoming a leader is the largest. As malicious nodes become leaders, the probability of other malicious nodes becoming the leaders decreases, that is $\frac{K}{M} > \frac{K-1}{M-1}$. In other words, the probability of non-malicious nodes becoming the leaders increases.

Therefore, Eq.6 can be modified as:

$$P\left(X|X = \lfloor \frac{N}{2} + 1 \rfloor\right) < \frac{N!}{X!(N-X)!} \cdot \left(\frac{K}{M}\right)^X \cdot \left(1 - \frac{K}{M}\right)^{N-X} \tag{7}$$

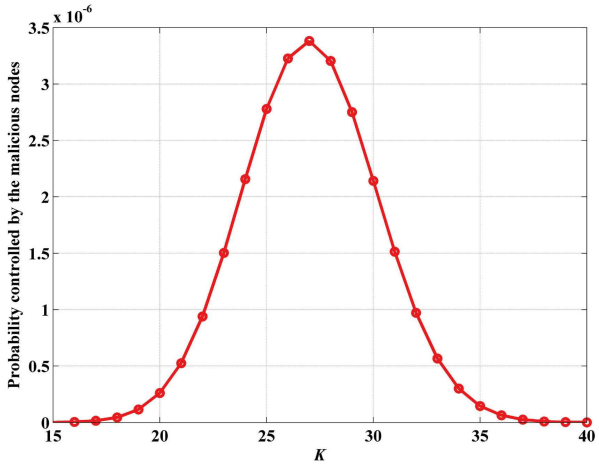


Fig. 5. The probability of the leader team being controlled by the malicious nodes.

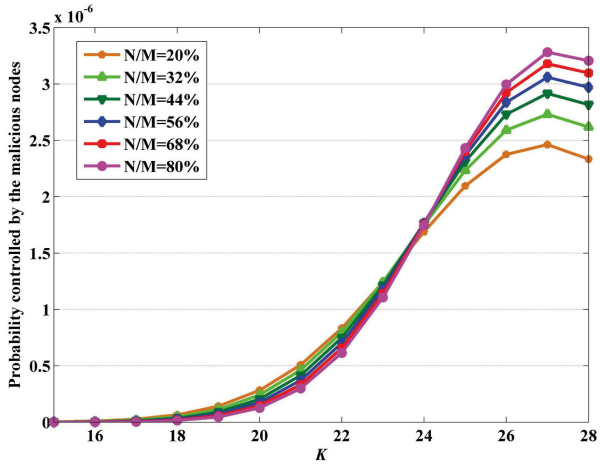


Fig. 6. The impacts of different values of N/M on the probability of the leader team being controlled by the malicious nodes.

In the blockchain network, the probability that the leader team is controlled by malicious nodes is:

$$P\left(X|X \geq \left\lfloor \frac{N}{2}+1 \right\rfloor\right) < \frac{1}{M} \cdot \sum_{X=\lfloor \frac{N}{2}+1 \rfloor}^{N} \frac{N!}{X!(N-X)!} \cdot \left(\frac{K}{M}\right)^{X} \cdot \left(1-\frac{K}{M}\right)^{N-X} \quad (8)$$

Second: Since M managers come from Q nodes applying to become managers, and K out of R malicious nodes will become managers, so the probability of K malicious nodes among M managers is:

$$P(Y|Y=K) < \frac{M!}{Y!(M-Y)!} \left(\frac{R}{Q}\right)^{Y} \left(1-\frac{R}{Q}\right)^{M-Y} \quad (9)$$

Finally: In the blockchain network, the probability P of the number of malicious nodes in the leader team exceeding 50% is finally calculated as:

$$P = P(Y|Y=K) \cdot P\left(X|X \geq \left\lfloor \frac{N}{2}+1 \right\rfloor\right) < \frac{1}{M} \bullet \frac{M!}{Y!(M-Y)!} \left(\frac{R}{Q}\right)^{Y} \left(1-\frac{R}{Q}\right)^{M-Y} \bullet \sum_{X=\lfloor \frac{N}{2}+1 \rfloor}^{N} \frac{N!}{X!(N-X)!} \cdot \left(\frac{K}{M}\right)^{X} \cdot \left(1-\frac{K}{M}\right)^{N-X} \quad (10)$$

We use Eq.10 to obtain probability P that the leader team is controlled by malicious nodes. The parameter settings are shown as follows: There are $Q = 20\,000$ nodes in the blockchain network, in which the number of malicious nodes R accounts for 51%, the number of manager nodes M accounts for 0.25%, and the number of leader nodes N is half of the number of manager nodes. At this time, the value of P is shown in Fig. 5. In this figure, the x coordinate represents K and the value range is $(\lfloor \frac{N}{2} \rfloor, M]$. When K is near the median of M, the probability that the leader team is controlled by the malicious nodes is the largest, about $3.381 * 10^{-6}$. And when K is close to M, the probability is the smallest, about $1.195 * 10^{-19}$.

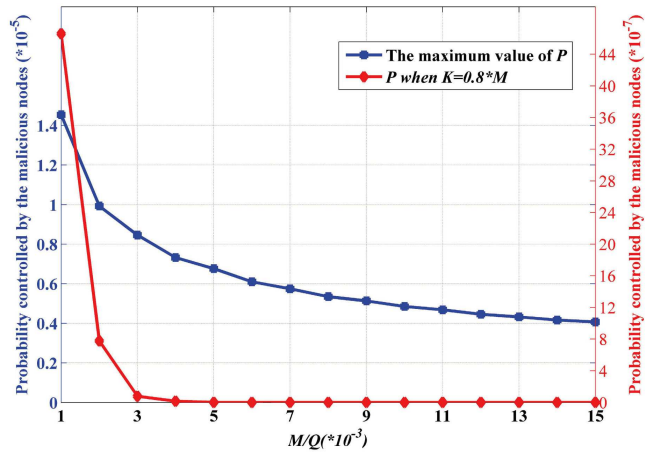


Fig. 7. The impacts of different values of M/Q on the probability of the leader team being controlled by the malicious nodes.

We also show the impacts of the change of the number of leaders on probability P, as illustrated in Fig. 6. The larger the value of N/M, the larger the value of P, but when the value of N/M varies from 0.2 to 0.8, the maximum value of P increases from $2.462 * 10^{-6}$ to $3.281 * 10^{-6}$, indicating little change. Therefore, the proportion of the leaders to the managers has a limited impact on probability P. In addition, Fig. 7 shows the impacts of the change of the number of managers on probability P, including the change in the maximum value of P and the change in the value of P when $K = 0.8\ M$. It can be seen from the figure that the greater the number of managers, the smaller the value of P. However, when the number of managers is too large, the negotiation cost will increase and the probability of reaching agreement with each other will decrease, that is, the negotiation will take too long. Moreover,

when the number of managers increases sharply, the decrease in the value of P tends to be flat.

In summary, the maximum order of magnitude of the probability that the leader team is controlled by malicious nodes is 10^{-6}, and its value is relatively low. Therefore, the leader team is reliable. ■

Lemma 3: Activeness: Within a period from the start that the miners apply for mining qualification to the end that miners mine blocks, a new block is always generated, and the block is unique.

Proof: Based on Algorithm 2, M nodes are granted mining qualifications and can conduct mining activities within each period. Compared with the current Bitcoin blockchain system, the mining difficulty will be reduced. So for M nodes, one or more new blocks are always generated. Algorithm 4 ensures that a new block with the hash value of the same parent block is received or abandoned by other nodes within a certain period. Therefore, the newly generated block is unique. ■

Correctness: In the proposed DTNB framework, the process of appending the add-chain to the parent block on the main chain is the transaction confirmation process. During this process, the correctness of the add-chain can be guaranteed from two aspects, namely the verification of add-chain generator and the add-chain consensus. On the one hand, the block generator of the add-chain is determined by the current round of managers, and elections are adopted to determine the managers of each round (the election process of managers is shown in Section III-C and Algorithm 2), so managers are changing. Before the election of managers, each miner needs to apply for mining qualifications, and their discrete tokens also change, which leads to changes of HV. And HV has the following characteristics: unpredictability (hash function property), uncontrollability (malicious miners cannot always obtain mining qualifications), and tamper resistance (unable to pass the consensus of managers), thereby ensuring the fairness of mining qualification allocation. In addition, during the consensus process of local networks, managers need the original data from the block generator (Eq. (2) and Eq. (4)). Note that the original data is generated by a hash function, and according to the hash function property, it cannot be forged, making the identity of malicious nodes unable to pass the verification of managers, thus the blocks generated by these malicious nodes cannot pass consensus.

On the other hand, the process of add-chain consensus includes the block consensus in the local network and the second consensus of the blockchain network. When the network is interrupted, the account balance of the transaction initiator cannot be accurately obtained within the local network. At this time, block verification mainly focuses on the identity of block generator, the mining difficulty, and the verification of transaction records based on the local blockchain, ensuring the correctness of blocks on the add-chain. After broadcasting the add-chain to the blockchain network, other miners need to conduct a second consensus, focusing on the verification of transaction records and the elimination of illegal transaction records (such as insufficient balance and repeated transactions) to guarantee the correctness of the add-chain. At the same time, the fork processing (Algorithm 3) ensures that in different remote areas, the specified add-chain cannot be appended to the main chain due to factors such as miner coordination and high computing power. The reason is that multiple add-chains requesting to be appended to the same parent block participate in the calculation of the benchmark hash value, which cannot be estimated and controlled according to the hash function property. Thus, the behavior of the local network being controlled by malicious miners or forging the local network to generate the add-chain does not affect DTNB, which can also ensure the correctness of blocks on the add-chain. In summary, the verification of add-chain generator and the two types of add-chain consensus cover the entire process from the generation of the add-chain to the appending of the main chain, which can guarantee the correctness.

Scalability: The applicable scenario of DTNB is a delay tolerant network, which extends the network requirements of blockchain operation from a continuous connection network to a disconnection network. This can meet the mining needs of miners who cannot obtain the latest blockchain in real time. From the technical perspective, DTNB realizes that the add-chain can be appended to the main chain, and completes the combination of the relatively independent main chain and the add-chain, thus demonstrating the blockchain scalability.

B. Experimental Simulations

To evaluate the performance of the proposed DTNB framework, we used the PeerSim simulator [33] to build a P2P network environment which is suitable for blockchain. In the Node method, the miner's unique identifier, computing power, virtual account and other information are defined. The network takes the main router nodes as the core and divides into urban areas and remote areas. The median value of 32 ms (one-way transmission) is set as the data transmission delay between nodes in the urban network and the typical value of satellite communication, i.e. 270 ms (one-way transmission), is set as data transmission delay in the remote area. We adopted the Bitcoin protocol as the basic blockchain transaction protocol and comparison object, and used the TCP/IP protocol to transmit data in the network. The nodes in the blockchain network are abstracted as data objects, including the miner ID, the set of neighbor nodes, the list of connected neighbor nodes, the list of connectable neighbor nodes, current manager nodes, the list of IP address of leader nodes, and the list of IP address of leader nodes in the next round.

Based on the above-mentioned P2P network, we established a delay tolerant network to run DTNB, and adopted key events on nodes to update the status of the transaction framework. The key events are as follows. (1) The event of negotiation between managers and leaders, that is, in the current round of mining, they determine the next round of managers and leaders, and at the same time, the miner nodes update the list of IP address of leader nodes in the next round. (2) The event of new block generation indicates that the miner has completed the block mining and uploads a copy of the block to the leader for approval. (3) The event of message reception, that is, the

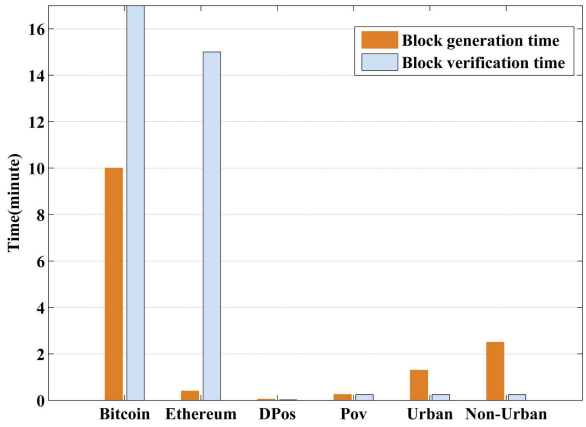


Fig. 8. Comparison of block generation time and block verification time between different blockchain systems.

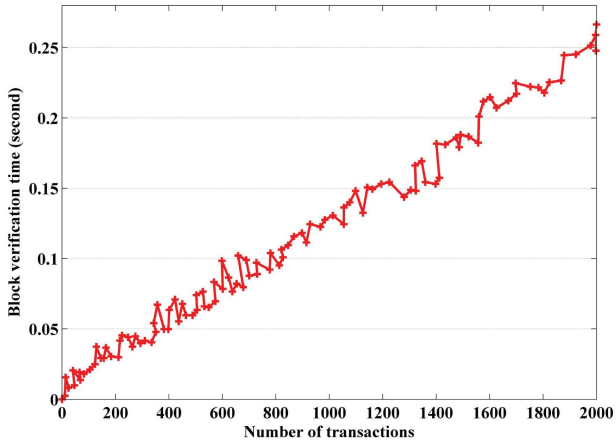


Fig. 9. The verification time for blocks containing the different number of transactions.

destination node receives the message from the source nodes and completes the corresponding operation according to the message type, such as sending the source data of the block ownership and updating the local blockchain. (4) The event of neighbor node selection, that is, the node chooses to disconnect some neighbor nodes according to the behavior of neighbor nodes (such as sending false messages, malicious attacks, etc.) or the bandwidth delay, and randomly selects the new neighbor nodes from the list of connectable neighbor nodes.

In the PeerSim, CDProtocol defines operations to be run in a Cycle-based model. For each cycle, when the condition (satisfies that the miner is the leader and the leader receives the message of the consensus block) is triggered, the consensus algorithm is executed. And when the condition of obtaining the mining qualification is triggered, the mining algorithm is executed. The consensus algorithm analyzes the block data and judges the value of X. When the value of X is 1, the included classes implement the following functions: receiving the pending consensus block and finding its parent block, judging the legality of the pending block, restoring the received block as the add-chain, calculating the hash value of each add-chain, resolving the forking of the add-chain, and negotiating between leaders to determine the appended add-chain; when the value of X is 0, the included classes implement: receiving the pending consensus block and finding its parent block, judging the legitimacy of the pending consensus block, solving the forking of the main chain, and negotiating between leaders to determine the appended block. For the mining algorithm, the included classes implement the following functions: monitoring the messages of mining qualification attribution, monitoring the arrivals of new blocks, calculating hash values to generate blocks, and requesting the latest block from neighbor nodes. Moreover, Linkable is the transmission protocol of PeerSim and we adopted the TCP protocol, providing services for accessing a set of neighbor nodes to realize mutual communication and data transmission between miners.

The simulation experiments are set as follows. The number of nodes in the network is 1000, the size of a single transaction is 250 bytes, the size of each data packet is 1480 bytes. The Hash256 function is used to hide sensitive information and mine blocks. The difficulty of block mining is adjusted based on changes in the number of nodes in the network. Fig. 8 shows the block generation time and block verification time using DTNB in urban areas and remote areas, and also shows the results of comparison with other blockchain systems, including Bitcoin, Ethereum, DPoS and PoV. In this experiment, the block generation time consists of two parts, namely the time for determine the managers and leaders and the block mining time. Compared with DPoS and PoV, DTNB does not have an advantage. The reason is that in order to improve the security of the blockchain, the block generation adopts the calculation nonce method in PoW, which takes a certain amount of time. In addition, the block rate using DPoS and PoV is faster, which causes more blocks being appended to the forks and therefore abandoned. For the block verification time, it means that the miner uploads a copy of the mined block to the leaders, and the leaders verify and negotiate to confirm whether the block is legal. The time consumed in this process is consistent with PoV. However, it should be noted that due to the large delay and interruption in the remote networks, the time for uploading the copy of the block to the EN is not counted.

Fig. 9 shows the relationship between the block size and the block verification time, in which the block size is realized by the number of transactions it contains. When the block is large, the leader nodes need more time to verify the legality of each transaction in the block. It should be noted that when negotiating with other leaders, because only the block identifier (the block generates 256-bit data through the HASH256 function) and legal result are propagated to the destination node, the time required to propagate different block sizes is consistent.

Fig. 10 shows the impacts of different numbers of manager nodes on the throughput, in which the throughput is defined as the number of transactions that can be processed per second. Theoretically, the larger the value of throughput, the closer to the demand for real-time transactions, but an increase in the value will either cause the block size to become larger or the verification time to become shorten. For the former, the increase of block size will reduce the number of full nodes in

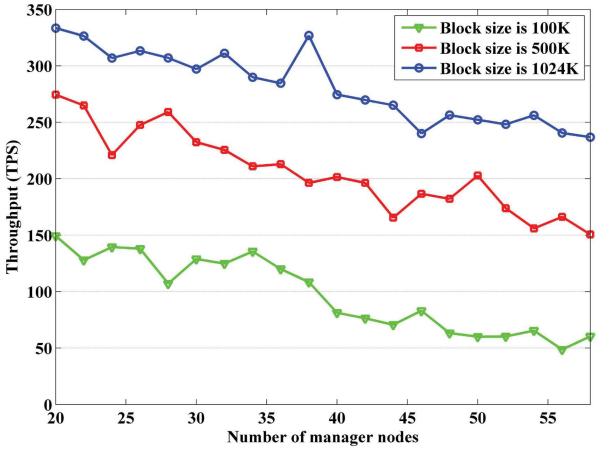


Fig. 10. The impacts of different numbers of manager nodes on the throughput (The throughput of Bitcoin is 7tps and the Ethereum is 10tps).

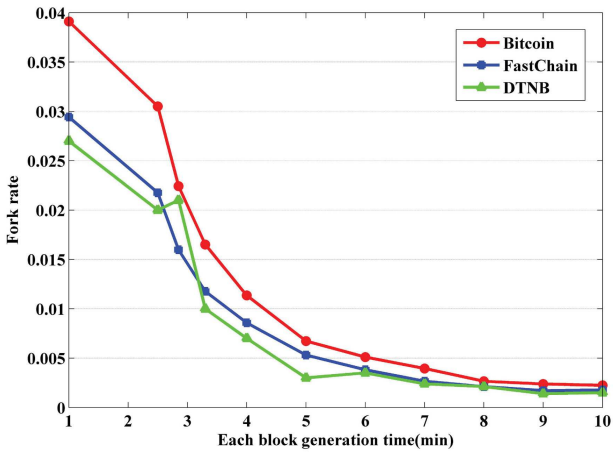


Fig. 11. The impact of block generation time on the fork rate.

the network, which violates the idea of decentralization of blockchain. And for the latter, the shorten of the verification time will increase the probability of fork generation, thereby increasing the probability of blocks being abandoned. It can be seen from the figure that as the number of manager nodes increases, the number of messages that need to be transmitted for negotiation between managers increases, resulting in an increase in the negotiation time. When the block size becomes larger, the number of transactions contained in the block will increase, resulting in an increase in throughput. In general, the increase in the number of managers will have a certain impact on the throughput, leading to its decrease, but the more the number of managers, the higher the security of blockchain.

Fig. 11 shows the impact of block generation time on the fork rate, in which the fork rate is defined as the number of abandoned blocks per second. Compared with Bitcoin and FastChain [22], the proposed DTNB is superior in the fork rate. FastChain uses bandwidth as a measure standard and adopts the neighbor node selection and update methods to spread the copy used for block consensus to the blockchain network through high-bandwidth neighbor nodes. Its core is still the competition of miners in all networks, resulting in

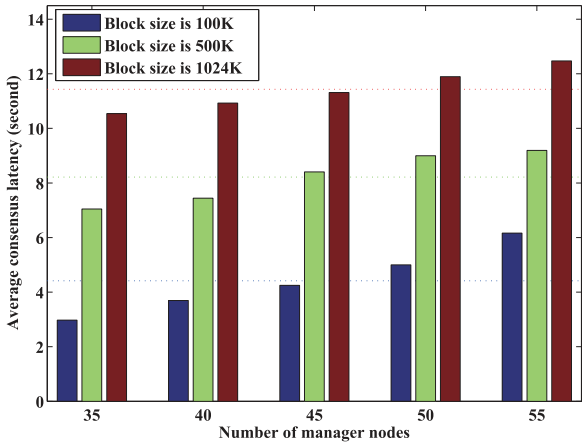


Fig. 12. The impacts of different numbers of manager nodes on the average consensus latency.

more blocks generated at the same time period and then more forks are created. The proposed DTNB fixes the miners who generate blocks in each round within a certain range, so the number of blocks generated in the same time period is limited, therefore, the fork rate can be reduced. It is worth noting that the local network with high computing power may generate more blocks under the disconnection network, resulting in a longer add-chain. At this time, it is possible that the blocks on the add-chain are not completely propagated during the network connection, or the add-chain is not accepted when using Algorithm 3 to process the forks. Then the number of abandoned blocks will increase, leading to an increase in the fork rate. This is the reason for slight fluctuations shown in Fig. 11.

Fig. 12 shows the change of the average consensus latency. The consensus latency refers to the time from when a block is generated to when the block is appended to the blockchain. So it contains two parts: the block transmission time and the block consensus time. Noting that if X=1, the block transmission time is the sum of the block transmission time in NU and EN, and the block consensus time is the sum of the time for block consensus in NU and EN. In this figure, when the number of managers increases, the average consensus latency will increase. The reason is that managers need more time to negotiate to reach a consensus. Also, the larger the block size, the more time it takes to transmit the block to neighbor nodes, resulting in longer consensus latency.

Block confirmation time is also one of evaluation indicators to measure the performance of the blockchain. In the proposed DTNB framework, it refers to the time from the block being broadcast to the nodes in the EN to being appended to the blockchain, which is composed of (1) the waiting time for reconnecting to the network (used for broadcasting the add-chains generated by NU when the network is interrupted), (2) the block transmission time, (3) the time for leaders to wait for block arrivals (Algorithm 4), and (4) the time for leaders to negotiate and determine the legal blocks or add-chains (Algorithm 3, 4 and 5). Specifically, (1) and (2) are uncertain, (3) can be set as a preset variable, and (4) is evaluated in Fig. 8 and Fig. 12. Moreover, (1) has a great impact on the block confirmation time, but is not related to

the proposed DTNB performance. Based on the above analysis, the block confirmation time is not given a detailed evaluation in this paper.

V. Conclusion and Discussion

Remote mountains and islands are connected to the Internet through satellites, which is a delay tolerant network. For this kind of network, there is an unsolved problem of blockchain deployment, that is, how to append the blocks generated when the network is interrupted to the main chain. In this paper, we propose the DTNB framework to overcome the current shortcoming of only deploying blockchain on continuous connectivity networks, thereby solving the above problem. DTNB adopts the add-chain to improve the scalability of blockchain. And the discrete token generation algorithm takes the miner ID and random number as the core, which has the characteristics of security and uniqueness. Meanwhile, the mining qualification attribution algorithm based on discrete token negotiation has the characteristics of fairness and randomness, which ensures miners in DTNB get the mining qualification with equal probability. Furthermore, a fork processing algorithm based on the principle of balancing maximum computing power and fairness enables the blocks on the add-chains generated by different local networks are appended to the main chain with equal probability. And the second consensus scheme for generating the add-chain in the local network avoids the problem of false and repeated transactions of the blocks on the add-chains. Through theoretical analysis and experimental simulations, the proposed DTNB has advantages in terms of throughput, fork rate, block generation time and block verification time.

We discuss the impact of incomplete block transmission in DTNB. When the network is reconnected, miners located in the local network broadcast the generated blocks to the blockchain network for consensus. At this time, there may be a situation of incomplete block data broadcasting. This situation can cause the current block and the blockchain with this block as the parent block to fail to reach a consensus, but it does not affect the operation of DTNB. Meanwhile, miners in the local network can negotiate the number of blocks generated based on the broadcasting and bandwidth conditions to ensure the integrity of block data during broadcasting. Unlike the current blockchain that conducts the consensus on only one block in a consensus process, the consensus object in DTNB is multiple blocks, that is, the blockchain generated by the local network when the network is interrupted. During the transmission process, the miners participating in the consensus do not know how many blocks there are on the blockchain. So, when receiving a block, it may happen that the parent block of block i is not found. After a certain period of time, block i will be discarded. At this time, only blocks with a chain relationship can be conducted the consensus, which will cause multiple transactions to roll back, thereby prolonging the confirmation time of certain transactions.

References

[1] Accessed: Jul. 20, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Global_Internet_usage

[2] K. Anton, "The decade-long cryptocurrencies and the blockchain rollercoaster: Mapping the intellectual structure and charting future directions," *Res. Int. Bus. Finance*, vol. 51, 2020, Art. no. 101067.

[3] I. Mistry, S. Tanwar, S. Tyagi, and S. Kumar, "Blockchain for 5G-enabled IoT for industrial automation: A systematic review, solutions, and challenges," *Mech. Syst. Signal Process.*, vol. 135, 2020, Art. no. 106382.

[4] J. Le, and X. Zhang, "BCOSN: A blockchain-based decentralized online social network," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1454–1466, Dec. 2019.

[5] P. Danzi, A. E. Kalør, Č. Stefanović, and P. Popovski, "Delay and communication tradeoffs for blockchain systems with lightweight IoT clients," *IEEE Internet Things J.*, vol. 6, no.2, pp. 2354–2365, Apr. 2019.

[6] V. Lopes and L. Alexandre, "An overview of blockchain integration with robotics and artificial intelligence," 2018, *arXiv:1810.00329*.

[7] Y. Guo, J. Tong, and C. Feng, "A measurement study of bitcoin lightning network," *Proc. IEEE Int. Conf. Blockchain*, 2019, pp. 202–211.

[8] P. Li, T. Miyazaki, and W. Zhou, "Secure balance planning of off-blockchain payment channel networks," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1728–1737.

[9] Mobile base stations. Accessed: Oct. 23, 2020. [Online]. Available: https://www.telstra.com.au/consumer-advice/eme/base-stations

[10] Web station solution. Accessed: Oct. 23, 2020. [Online]. Available: https://carrier.huawei.com/cn/products/wireless-network/site

[11] KUHA mobile network. Accessed: Oct. 23, 2020. [Online]. Available: https://www.kuha.io

[12] A. Pentland, R. Fletcher, and A. Hasson, "DakNet: Rethinking connectivity in developing nations," *Computer*, vol. 37, no. 5, pp. 78–83, 2004.

[13] C. Blattman, J. Robert, and R. Raul, "Assessing the need and potential of community networking for developing countries: A case study from India," Harvard Center for International Development, 2002. [Online]. Available: edevelopment.media.mit.edu/SARI/papers/Community Networking.pdf

[14] W. Miao, G. Min, Y. Wu, H. Wang, and J. Hu, "Performance modelling and analysis of software-defined networking under bursty multimedia traffic," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 12, no. 5, pp. 1–19, 2016.

[15] C. Tselios, I. Politis, and S. Kotsopoulos, "Enhancing SDN security for IoT-related deployments through blockchain," in *Proc. IEEE Conf. Netw. Func. Virtual. Softw. Defined Netw.*, 2017, pp. 303–308.

[16] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, Q. Zhang, and K.-K. R. Choo, "An energy-efficient SDN controller architecture for IoT networks with blockchain-based security," *IEEE Trans. Services Comput.*, vol. 13, no. 4, pp. 625–638, Jul.–Aug. 2020.

[17] L. Xie, Y. Ding, H. Yang, and X. Wang, "Blockchain-based secure and trustworthy internet of things in SDN-Enabled 5G-VANETs," *IEEE Access*, vol. 7, pp. 56656–56666, Apr. 2019.

[18] Y. Hu *et al.*, "A delay-tolerant payment scheme based on the ethereum blockchain," *IEEE Access*, vol. 7, pp. 33159–33172, Mar. 2019.

[19] Q. H. Mahmoud, L. Michael, and A. May, "Research challenges and opportunities in blockchain and cryptocurrencies," *Internet Technol. Lett.*, vol. 2, no. 2, 2019, Art. no. e93.

[20] S. King and N. Scott, "Pcoin: Peer-to-peer crypto-currency with proof-of-stake," 2012. [Online]. Available: https://decred.org/research/king2012.pdf

[21] K. Li, H. Li, H. Hou, K. Li., and Y. Chen, "Proof of vote: A high-performance consensus protocol based on vote mechanism & consortium blockchain," in *Proc. IEEE 19th Int. Conf. High Perform. Comput. Commun., IEEE 15th Int. Conf. Smart City, IEEE 3rd Int. Conf. Data Sci. Syst.*, 2017, pp. 466–473.

[22] K. Wang and H. S. Kim, "FastChain: Scaling blockchain system with informed neighbor selection," in *Proc. IEEE Int. Conf. Blockchain*, 2019, pp. 376–383.

[23] H. Pervez, M. Muneeb, M. U. Irfan, and I. U. Haq,"A comparative analysis of DAG-Based blockchain architectures," in *Proc. 12th Int. Conf. Open Source Syst. Technol.*, 2018., pp. 27–34

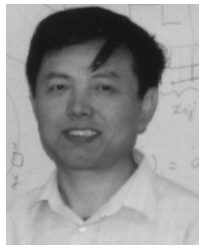
[24] L. Cui, S. Yang, Z. Chen, Y. Pan, M. Xu, and K. Xu, "An efficient and compacted DAG-based blockchain protocol for industrial internet of things," *IEEE Trans. Ind. Inform.*, vol. 16, no. 6, pp. 4134–4145, Jun. 2020.

[25] P. Ferraro, C. King, and R. Shorten, "On the stability of unverified transactions in a DAG-based distributed ledger, "*IEEE Trans. Autom. Control*, vol. 65, no. 9, pp. 3772–3783, Sep. 2020.

[26] W. Yang, X. Dai, J. Xiao, and H. Jin, "LDV: A lightweight DAG-based blockchain for vehicular social networks, "*IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5749–5759, Jun. 2020.
[27] H. Chen and Y. Wang, "SSChain: A full sharding protocol for public blockchain without data migration overhead," *Pervasive Mobile Comput.*, vol. 59, 2019, Art. no. 101055.
[28] D. Chatzopoulos, S. Gujar, B. Faltings, and P. Hui, "Mneme: A mobile distributed ledger," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1897–1906.
[29] J. Poon and D. Thaddeus, "The bitcoin lightning network: Scalable off-chain instant payments," Accessed: Jan. 14, 2018. [Online]. Available: https://lightning.networklightning-network-paper.pdf
[30] A. Singh *et al.*, "Sidechain technologies in blockchain networks: An examination and state-of-the-art review," *J. Netw. Comput. Appl.*, vol. 149,2020, Art. no. 102471.
[31] M. Saad, M. T. Thai, and A. Mohaisen, "POSTER: Deterring ddos attacks on blockchain-based cryptocurrencies through mempool optimization," in *Proc. Asia Conf. Comput. Commun. Secur.*, 2018, Art. no. 809–811.
[32] Z. A. E. Houda, A. Hafid, and L. Khoukhi, "Brain Chain-A machine learning approach for protecting blockchain applications using SDN," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.
[33] I. Kazmi and S. F. Y. Bukhari, "PeerSim: An efficient & scalable testbed for heterogeneous cluster-based P2P network protocols," in *Proc. UkSim 13th Int. Conf. Comput. Modell. Simul.*, 2011, pp. 420–425.

Xin Cong received the Ph.D. degree in computer science from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014. He is currently an Associate Professor with the School of Electronic and Information Engineering, Liaoning Technical University, Fuxin, China, and a Visiting Scholar with the Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA. His research interests include blockchain, P2P network, and cloud computing.

Lingling Zi received the Ph.D. degree in computer science from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014. She is currently an Associate Professor with the School of Electronic and Information Engineering, Liaoning Technical University, Fuxin, China, and a Visiting Scholar with the Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA. Her research interests include blockchain and P2P network.



Ding-Zhu Du received the M.S. degree from the Chinese Academy of Sciences, Beijing, China, in 1982 and the Ph.D. degree from the University of California, Santa Barbara, Santa Barbara, CA, USA, in 1985, under the supervision of Professor Ronald V. Book. Before settling with the University of Texas at Dallas, Richardson, TX, USA, he was a Professor with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA. He is currently the Editor-in-Chief of the *Journal of Combinatorial Optimization* and is also on the editorial boards for several other journals.