

# Data Transmission Scheme Considering Node Failure for Blockchain

Jiao Li<sup>1</sup>

Published online: 6 February 2018

© Springer Science+Business Media, LLC, part of Springer Nature 2018

**Abstract** In recent years, Blockchain technology has attracted considerable attention from the industrial circle. Blockchain is a distributed ledger technology, which must be validated, stored and maintained by all nodes to ensure data security, transparency, and integrity. The communication efficiency of Blockchain is an important factor restricting its application. Existing algorithms can provide data routing schemes for Blockchain but without considering the node failure. On Blockchain, node failure is a common phenomenon due to the nodes' selfishness and nodes' mobility. Node failure degrades the network performance or even sometimes makes the network useless. This paper proposes a data transmission scheme considering node failure for finishing validation of block data on Blockchain, which firstly sets response threshold level to detect failure node, and then using greedy idea constructs communication tree to organize all nodes forwarding block data. Based to the multi-link concurrent communication tree model, this scheme maximizes the potential transmitting capacity of nodes and assigns proper tasks to other nodes beside source node, so it can shorten the validation time of Blockchain transaction, and improve resistance to node failure. Theoretical proof and experimental results show the effectiveness and the efficiency of the proposed data transmission scheme.

**Keywords** Blockchain · Node failure · Communication tree · Communication time

## 1 Introduction

In recent years, Blockchain technology, the core and basic framework of Bitcoin, has rapidly gained popularity and more attention. Bitcoin is the first successful decentralized global digital cash system, and Blockchain is essentially decentralized P2P computing

---

✉ Jiao Li  
lijiao@xsyu.edu.cn

<sup>1</sup> School of Computer Science, Xi'an Shiyou University, Xi'an, China

mode. Traditional transaction usually relies on third-party to solve trust problem and to facilitate the trades. However Blockchain establishes and maintains a global ledger that stores records for every transaction. Blockchain allows the trades mutually unbeknownst to transact securely without a centrally trusted intermediary [1–5]. On Blockchain, transaction records, called block data, are distributed to all nodes to validate and confirm; in other words, Blockchain requires computational resources, network resources and storage resources to replace the third-party in order to solve trust problem between trading partners [6–8]. At present, transactions on Blockchain can be validated and confirmed once every 10 min, and Blockchain can process only 8 transactions per second, so Blockchain is unsuitable for the high-frequency trades in business application [9]. The low ability of the business processing is a major hindrance to the broad adoption of Blockchain. How to organize all nodes to finish validation quickly involves Blockchain-based communication algorithms.

For Blockchain-based communication algorithms, there are two main issues to be considered. Firstly, a reasonable data transmission structure and an efficient communication mechanism are essential to Blockchain network performance. Secondly, a node may stop working either because of some internal reasons or some environmental issues like external physical damage. In order to deal with real chronic conditions, there must be some self-detection or self-configuration mechanism for the nodes. Therefore, proper data transmission scheme can help reduce the delay and increase the efficiency while considering the node failure of Blockchain network.

Node failure is a common phenomenon on Blockchain. Nodes may leave voluntarily or be failure randomly because of nodes' selfishness and nodes' mobility. Node failure partitions the network which may lead to an unstable network and degrade the communication performance. Therefore, this paper proposes a data transmission scheme considering node failure for Blockchain. The main contributions of this paper are presented as follows:

1. The multi-link concurrent communication tree model is established. Due to the heterogeneity of nodes, nodes are different in transmitting ability, so each node has multi-connections with other nodes for forwarding at the same time, and this model belongs to the multi-link communication which can make maximum use of nodes' potential communication capacity. It is obvious that the efficiency of the multi-link communication is superior to that of the single-link communication (used in star topology or in linear topology). Due to the introduction of concurrent communication mechanism, communication not only depends on the source node, and other powerful nodes are assigned to undertake forwarding tasks. Several transmission tasks are forwarded at the same time, so this model has high concurrency. In addition, due to the adoption of tree structure, the failure of individual nodes only affects their sub-trees, and does not destroy the whole communication. So this model has good malfunction isolation.
2. Based on this model, the data transmission algorithm considering node failure for Blockchain is proposed. In order to avoid the event that nodes are failed during transmitting, the proposed algorithm requires that the source node must receive the confirmation message from the participating nodes before transmitting block data. This belongs to prevention strategy in advance. Then this algorithm exploits the idea of greedy to construct a communication tree. Powerful nodes are selected to locate in the top or the upper of communication tree, and weak nodes are located in the lower of communication tree even located as leaf nodes.

3. Theoretical proof and comprehensive simulation is done to evaluate the proposed algorithm. Though theorem proof, it is found the relationship between the number of failure nodes and the concurrent communication time, and when one node is failed the number of its cascaded failure nodes can be calculated under equal-link communication tree. The first simulation experiment is performance parameters comparison under different percentage of failure nodes. The results show that even if 15% of nodes are failed, better communication performance also can be obtained. However, the failure nodes account for 30%, the concurrent communication time and the average end-to-end delay increase obviously. The second simulation experiment is performance analysis under failure nodes with different transmitting capacity. It is concluded from simulation results that the failure nodes, which are poor in transmitting even have no ability, do not affect the communication performance. But nodes with strong transmitting ability are failed, the communication performance degrades greatly.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 establishes the multi-link concurrent communication tree model for Blockchain. Based on the model, data transmission algorithm considering node failure for Blockchain is described in Sect. 4. Section 5 gives theorems and simulation. Finally, conclusions and directions for future work are presented in Sect. 6.

## 2 Related Work

Blockchain is originated in 2008 from the paper “Bitcoin: A Peer-to-Peer Electronic Cash System” by a Japanese scholar under the pseudonym of Satoshi Nakamoto [9]. It is reported in 2016 by Mike Hearn, a major Bitcoin developer, that the Bitcoin currency is a failed project. But Blockchain as the underlying technology of Bitcoin has attracted more attention in recent years. Blockchain is an innovative application mode of Internet, which integrates with distributed data storage, P2P, encryption algorithm, consensus mechanism and other computer technology, etc. Blockchain has wide application prospects in finance, security investment, global payment, trade items, etc. [10]. Compared with vigorous development of Blockchain application, the basic research of Blockchain technology has lagged. By retrieving literatures, literatures are few on Blockchain, so Blockchain is a new technology and still in its infancy. There are many factors, such as storage capacity, processing power, and transmitting efficiency, blocking the development of Blockchain. This paper focuses on transmitting efficiency problem.

In Bitcoin, as the first typical application of Blockchain, block data generated by one node is distributed to other nodes using Flooding broadcast algorithms. Although flooding broadcast algorithms are widely used to finish routing function and are easy to implement, due to high-frequency trading of Blockchain, flooding may cause serious information redundancy, bring broadcast storms and result in network congestion [11, 12].

It is generally known Blockchain is essentially a P2P network in the view of the network structure. But it is different from P2P network and owns its new characteristics. Node mobility and high-frequency transactions make communication environment significantly more complex and harsh [13]. Therefore proper topology structure and communication mechanism are needed to schedule the routing for transaction validation.

Tree topology structure is often chosen to improve and optimize the communication efficiency [14, 15]. Paper [16] proposed a concurrent communication tree algorithm. With  $N$  nodes, the algorithm shrinks the communication time to  $O(\ln(N))$ . Compared with the

line and star structure, tree topology obtains better performance in communication efficiency. Using this algorithm, it is simple to construct a tree for communication. However, it is assumed that all the nodes are equally capable to perform the network transmitting task which is not true in the real environment. Paper [17] proposed a P2P communication tree algorithm based on multi-link. It introduces the concept of node communication link number, and assigns proper communication task to nodes according to their transmitting capacity in order to shorten the concurrent communication time. This algorithm considers the communication capacity of nodes, so the algorithm in Paper [16] is a kind of special case of this algorithm. Paper [18] proposed a IFT algorithm which integrates communication factors (such as node communication link number, node trust degree) and improves both the efficiency and the reliability of Blockchain communication. Paper [19] designed a stable clustering algorithm and a corresponding clustering routing protocol for large-scale mobile Ad Hoc networks. The proposed routing protocol uses proactive protocol between nodes within individual clusters and reactive protocol between clusters. The proposed clustering algorithm improves the stability by decreasing the clusters' number and nodes' switch times between clusters.

However, these algorithms mentioned above ignore the fact of node failure. On Blockchain, a node may stop working either because of some internal reasons (the running out of battery power, hardware malfunction, etc.) or some environmental issues (network congestion, the failure of link connections, physical external damage, etc.).

Paper [20] introduced the background and the causes of node failure of P2P network and then reviewed the main progresses of node failure prediction, node data recovery, super-node selection and routing node algorithm etc. Paper [21] proposed a localized hybrid timer based cut-vertex node failure recovery approach to handle partitions and restore connectivity with the use of multiple backup nodes for the guaranteed partitioned recovery.

The existing literatures proposed different strategies to solve node failure problem for a specific network environment (wireless sensor network [22–24], virtual network [25, 26], internet of things [27]). The existing algorithms or strategies mainly focused on routing recovery after node failure. And they may ignore the following (1) the relationship between the node failure ratio and the decline of communication efficiency; (2) the impact of communication efficiency under failure nodes with different transmission capacity; In addition, There is no considerable work have been seen dealing on optimized routing scheme with the transmitting efficiency problem with respect to Blockchain while considering the node failure as well. This is where our work comes into play and that is the main difference from previous work. Our work aims to provide a data transmission scheme considering node failure, which has high concurrency and high efficiency. It also can maintain good performance in communication even if there are 15% failure nodes. The proposed scheme gives a routing plan for Blockchain validation.

### 3 Multi-link Concurrent Communication Tree Model

#### 3.1 Nodes' Behavior Analysis on Blockchain

Nowadays, people can produce and consume information anytime and anywhere by mobile wireless networks, and more customers choose mobile payment technology to finish their transactions. Payment by mobile phones and PDA terminals, would be more convenient compared with the traditional means of payment. Since a large amount of mobile terminals

rush into Blockchain and participate in transaction validation. Blockchain may face new problems and new challenges. On Blockchain, nodes' behavior characteristics are as follows.

1. The dynamicity and mobility of nodes. More people resort to mobile terminals by wireless mode to transfer payment and participate in transaction validation. But on Blockchain mobile terminals move and change their physical location, even leave the Blockchain network. The mobility of nodes may result in the failure of sub-tree or the failure of route. Blockchain is a self-organized P2P network. We need to establish logical connections between nodes for transaction validation. The logical connections between nodes are based on the physical topology. So there exists a certain correlation between logical connections and actual physical topology. Mobile nodes frequently moving may lead to the topology mismatch problem. The established connections may be unavailable or even failure.
2. The selfishness of nodes. Because Blockchain adopts P2P network, Blockchain also advocates nodes' collaboration partnership and sharing spirit. But nodes are naturally and essentially self-centered, and they act selfishly to maximize their own interest. Nodes ask for information or resource and don't undertake the transaction validation and other communication tasks. Once they obtain information or resource, they abort or leave Blockchain. In the event that such nodes are selected to act as intermediaries between the source node to the destination node, their selfishness and unreliability can increase the rate of transmission failure.
3. The malfunction of nodes. If some failures (hardware malfunction, out-of-energy, auto power-off, and so on) occur, nodes can not repair themselves and require human intervention. They can not recover quickly, so these nodes become unavailable permanently for the whole transaction validation.

Through the analysis of nodes' behavior characteristics, it is known that the node failure is unavoidable and unexpected on Blockchain. No matter what reason nodes are failed, this could result in the poor performance of the network, even sometimes the paralysis of the whole network. On Blockchain, different data transmission schemes are used to forward the block data. These transmission schemes must consider the node failure cases and how to handle them. This paper proposes a data transmission algorithm with node failure for Blockchain.

### 3.2 Node Classification and Transmitting Role

Nodes are classified into three types by the role in transmission task. They are source node, forwarding nodes and non-forwarding nodes respectively. Node classification and transmitting role is shown in Table 1. In the case of the forwarding nodes' failure, it may result in the failure of sub-trees, and affect the efficiency of transaction validation. Therefore, this paper just needs to consider the failure of the forwarding nodes.

### 3.3 Establishing Model

The tree topology has been widely used in P2P networks and performs the feasibility and effectiveness in data forwarding, information sharing and resource searching [14–18]. The tree topology has good malfunction isolation and good pollution isolation. An acyclic tree topology can reduce redundant retransmission and network overhead. In order to finish the

**Table 1** Node classification and transmitting role

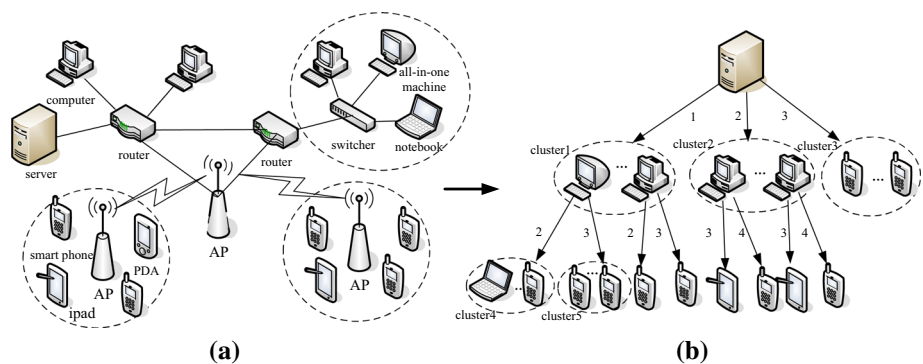
Node classification	Position in communication tree	Communication reparability if failure	Impact of communication performance if failure
Source node	The root of the communication tree	No	Communication abort
Forwarding nodes	Located in the middle depth of the communication tree	Yes	The deeper the location of node, the smaller the impact of performance
Non-forwarding nodes	Located as leaf nodes	Yes	No impact

transaction validation of Blockchain, according to the physical network topology, a tree rooted from a communication source node is established by all other nodes.

Figure 1a shows physical topology of Blockchain. Blockchain is a distributed P2P network which is composed of nodes, called miners in Bitcoin system. On Blockchain, nodes can be regarded as mobile terminals, PCs, servers or a miner pool which is made up of any number of nodes. To be one node, it requires that the node must have computing power regardless for its physical hardware configuration. All nodes of Blockchain are distributed in the broad range of geographical area, and they are meshed together physically by wired or wireless connections.

Figure 1b shows multi-link concurrent communication tree. All nodes are abstracted logically into tree topology structure to finish transmission. This topology is a self-configuring network of nodes. All nodes cooperatively maintain network connectivity without the aid of any fixed infrastructure units such as routers, switchers, or access points. Each node must have routing function and forwards block data via intermediate nodes.

In Fig. 1a the lines between nodes belong to physical connections for actual network and in Fig. 1b the directed lines between nodes belong to logical connections for finishing an assigned communication task. The number on line stands for the sequential order of transmitting.



**Fig. 1** Multi-link concurrent communication tree model. **a** the physical topology of Blockchain, **b** the multi-link concurrent communication tree

How to construct a communication tree depends on algorithms. This is where we focus on. The principle of establishing communication tree is to reach load balance and to shorten communication delay. For the communication tree with  $N$  nodes, a good communication tree established based on algorithm is relatively low height, stout, and with spreading branches. That is, the tree has higher concurrency and shorter communication delay. To achieve this aim, make sure that nodes located in the top or the upper of the communication tree have powerful transmission capacity. That is, nodes closer to the source node are expected to have powerful transmitting ability. Because of concurrent communication mechanism, these nodes can undertake more communication tasks in follow-up communication.

On Blockchain all nodes are different in transmitting ability. We continue to use the communication link number for nodes in order to represent the heterogeneity of nodes' communication capacity on Blockchain-based communication [18]. So the proposed model belongs to multi-link.

In the multi-link concurrent communication tree model, the following assumptions are taken without losing generality:

1. Communication task (except for the routing information) between any two nodes are the same. For a transaction validation of Blockchain, the source node  $v_0$  must distribute the block data to all other nodes for validation, so all nodes receive the same block data.
2. Any node  $v_i$  receives the block data and quickly forwards it to other nodes without delay. So the proposed model is non-intermittent.
3. Several communication tasks are executed and transmitted at the same time, so the model is concurrent.

## 4 Data Transmission Scheme Considering Node Failure

This paper proposes a data transmission algorithm considering node failure, which consists of two steps: detecting failure nodes and establishing communication tree. Before establishing the communication tree, detecting failure nodes in advance belongs to preventive measures to avoid nodes' failure during transmitting. The detecting method is that the source node sends a confirmation message to all nodes of Blockchain before establishing the communication tree. Blockchain sets a response threshold denoted by  $T_{\text{threshold}}$ . If the response time which the source node  $v_0$  spends to receive the response message from nodes is longer than the given threshold, and nodes become non-responsive because of the failures or faults, the source node  $v_0$  regards these nodes as the failure nodes.  $T_{\text{threshold}}$  is pre-fixed for a specific Blockchain transaction. In private network, the  $T_{\text{threshold}}$  can be ignored [16].

Before establishing the tree, detecting all the participating nodes and identifying the failure nodes might be time-consuming. But it is necessary to do that, because it reduces the risk of nodes' failure in the process of transaction validation, and improves the reliability of transmitting.

After these detecting steps, we can get two node sets, which are the failure node set  $V_F$  and the unfailure node set  $V_U$ . The nodes in  $V_U$  can undertake transmitting tasks, so they are suggested to locate in the top or the upper of communication tree. The nodes in  $V_F$  should not be assigned tasks, so they are located as leaf nodes. In order to maximize the

potential communication capacity of nodes and to shorten the validation time of Blockchain transaction, greedy mechanism is adopted to construct communication tree. The selection order of nodes to be added to communication tree is calculated based upon their capability in transmitting. So the powerful nodes are located in the upper level of communication tree, and they can undertake more communication tasks in follow-up communication.

Based on the actual communication information, such as the geographical location information of nodes, the connections between nodes, and the transmitting ability of nodes, Blockchain network is abstracted as graph  $G = (V, E, L)$  with node set  $V$ , edge set  $E$ , communication link number set  $L$ .  $V = \{v_0, v_1, \dots, v_i, \dots, v_n\}, V_F \cup V_U = V, E = \{ \langle v_0, v_1 \rangle, \langle v_0, v_2 \rangle, \dots, \langle v_i, v_j \rangle, \dots, \langle v_n, v_{n-1} \rangle \}$ , An edge  $\langle v_i, v_j \rangle$  represents there exists the logical connection between  $v_i$  and  $v_j$ .  $\langle v_i, v_j \rangle \in E. L = \{l(v_0), l(v_1), \dots, l(v_i), \dots, l(v_n)\}$ , where  $l(v_i)$  represents the maximum communication link number for  $v_i$ . Set  $V_T$  and  $E_T$  represent node set  $V$  and edge set  $E$  for communication tree. The complement set of  $V_T, E_T, L_T$  is  $\overline{V_T}, \overline{E_T}, \overline{L_T}$  respectively. That is  $\overline{V_T} \cup V_T = V, \overline{E_T} \cup E_T = E, \overline{L_T} \cup L_T = L, f(t)$  is the concurrent communication time [18]. The algorithm of constructing the communication tree is described below.

*Step 1* Sort the set  $V_U$  by  $l(v_i)$  from largest to smallest, and obtain an ordered set  $V_U = \{ v_1, \dots, v_i, \dots, v_j, \dots, v_n \}, \forall v_i, \forall v_j \in V_U$  and  $l(v_i) \geq l(v_j)$ .

*Step 2* The source node is  $v_0$ , and the ordered set  $V_U = \{ v_1, \dots, v_i, \dots, v_j, \dots, v_k \}$ .

Let  $V_T = \{v_0 | v_0 \in V\}, E_T = \emptyset, L_T = \{l(v_0) | l(v_0) \in L\}$ . So  $\overline{V_T} = V_U \cup V_F, \overline{L_T} = \{l(v_1), \dots, l(v_i), \dots, l(v_n)\}$ ,

Let  $V_{Tmp} \leftarrow V_T, f(t) = 0$ .

*Step 3*  $k = 0$ .

*Step 4* Traverse node  $v_i$  in  $V_{Tmp}, \forall v_i \in V_{Tmp}$  and  $\forall v_i \notin V_F$  select  $m$  nodes  $\{v_1^i, \dots, v_k^i, \dots, v_m^i\}$  from the ordered set  $\overline{V_T}$  in turn, where  $m = \text{Min}\{\text{Num}(\overline{V_T}), l(v_i)\}, v_k^i \in \overline{V_T}, v_i$  transmits data to  $v_k^i$ , so  $v_i$  is the parent node of  $v_k^i$ .  $\text{Num}(X)$  is the number of elements of set  $X$ .

Let  $V_T \leftarrow V_T \cup \{v_1^i, \dots, v_k^i, \dots, v_m^i\}, E_T \leftarrow E_T \cup \{\langle v_i, v_1^i \rangle, \dots, \langle v_i, v_k^i \rangle, \dots, \langle v_i, v_m^i \rangle\}, L_T \leftarrow L_T \cup \{l(v_1^i), \dots, l(v_k^i), \dots, l(v_m^i)\}$ .

*Step 5* If  $k < \text{Num}(V_{Tmp})$  and  $\text{Num}(V_T) < N, k++$ , go to Step 4.

*Step 6* If  $\text{Num}(V_T) < N, V_{Tmp} \leftarrow V_T, f(t) = f(t) + 1$ . go to Step3, otherwise, algorithm ends.

The communication tree constructed based on the proposed algorithm has some characteristics. The upper the node  $v_i$  is located in, the larger the  $l(v_i)$  is. So the leaf nodes are poor in transmitting or have no ability to forward data. In addition, due to multi-link concurrent communication tree model, the communication tree is stout and has more spreading branches, which means diverse routes for transmitting. It can realize load balancing or fault tolerance.



## 5 Theorems and Simulation

### 5.1 Theorems for Communication Tree

**Theorem 1** For a single-link communication tree with  $N$  nodes, if the number of failure nodes denoted by  $N_{Fail}$  satisfies  $0 \leq N_{Fail} < N - 2^{(f(t)-1)}$ ,  $f_{Fail}(t) = f(t)$ , where  $f_{Fail}(t)$  indicates the concurrent communication time with considering node failure and  $f(t)$  indicates the concurrent communication time without considering node failure.

*Proof* According to the concurrent communication mechanism, the number of nodes in communication tree grows exponentially with time. For the communication tree of  $N$  nodes without considering node failure,  $f(t) = \lceil \log_2 N \rceil$ , where  $\lceil X \rceil$  indicates the smallest integer which is bigger than or equal to  $X$ . After the  $(f(t) - 1)$ th time compartment, there are  $2^{(f(t)-1)}$  nodes in communication tree. In the  $f(t)$ th time compartment,  $N - 2^{(f(t)-1)}$  nodes are added to the communication tree as leaf nodes. So If the number of failure nodes is no more than  $N - 2^{(f(t)-1)}$ , the concurrent communication time  $f_{Fail}(t)$  is equal to  $f(t)$ .

When the number of failure nodes is equal to  $N - 2^{(f(t)-1)}$ , the concurrent communication time  $f(t) = f(t) - 1$ . We generalize Theorem 1 in the case of multi-link condition and obtain Theorem 2.

**Theorem 2** For a multi-link communication tree with  $N$  nodes, the communication link number of any node  $v_i$  keeps at  $m$ , which means  $l(v_i) \equiv m$ . There exists a natural number  $k$ ,  $1 \leq k \leq f(t)$ . If  $N - (m + 1)^{(f(t)-k+1)} \leq N_{Fail} < N - (m + 1)^{(f(t)-k)}$  and  $0 \leq N_{Fail} \leq N$ , then  $f_{Fail}(t) = f(t) - k + 1$ .

*Proof* Proving of Theorem 2 is the same as that of Theorem 1, so it can be omitted.

Applying Theorem 2 above, Let  $m = 1$  and  $k = 1$ , and Theorem 1 can be obtained. Therefore Theorem 1 is regarded as a special case of Theorem 2. Theorem 2 shows the relationship between the number of failure nodes and the concurrent communication time.

**Theorem 3** For a single-link communication tree with  $N$  nodes, if the failure node  $v_i$  is located at the depth  $Level(v_i)$  of the communication tree, the number of all unavailable nodes in its following sub-tree is denoted by  $N_{Fail}(v_i)$ , which satisfies  $2^{(\lceil \log_2 N \rceil - Level(v_i) - 1)} \leq N_{Fail}(v_i) \leq 2^{(\lceil \log_2 N \rceil - Level(v_i))}$ , where  $Level(v_i)$  is the depth of node  $v_i$  in communication tree.

*Proof* The concurrent communication time  $f(t)$  is  $\lceil \log_2 N \rceil$  for a single-link communication tree. The depth  $Level(v_i)$  means that node  $v_i$  has received data after  $Level(v_i)$  time compartments. In the next  $(\lceil \log_2 N \rceil - Level(v_i) - 1)$  time compartment, the node  $v_i$  is fully connected with other nodes and the communication number of node  $v_i$  reaches the maximum value  $l(v_i)$ . But in the last  $(\lceil \log_2 N \rceil - Level(v_i))$  time compartment, the node  $v_i$  may be not needed to transmit to others. So according to the concurrent communication mechanism, if the node  $v_i$  is failed,  $2^{(\lceil \log_2 N \rceil - Level(v_i) - 1)} \leq N_{Fail}(v_i) \leq 2^{(\lceil \log_2 N \rceil - Level(v_i))}$ .

We expand Theorem 3 to more general case of multi-link condition and obtain Theorem 4.

**Theorem 4** For a multi-link communication tree with  $N$  nodes, the communication link number of any node  $v_i$  keeps at  $m$ , which means  $l(v_i) \equiv m$ . If the failure node  $v_i$  is located at the depth  $Level(v_i)$  of the communication tree, the number of all unavailable nodes in its

following sub-tree is denoted by  $N_{Fail}(v_i)$ , which satisfies  $(m + 1)^{\lceil \log_{(m+1)} N \rceil - Level(v_i) - 1} \leq N_{Fail}(v_i) \leq (m + 1)^{\lceil \log_{(m+1)} N \rceil - Level(v_i)}$ .

*Proof* Proving of Theorem 4 is the same as that of Theorem 3, so it can be omitted.

According to Theorem 4, the node  $v_i$  is leaf node, and its depth  $Level(v_i)$  is equal to  $\lceil \log_{(m+1)} N \rceil$ . So if leaf nodes are failed, they do not affect other nodes. The node  $v_0$  is the source node and  $Level(v_0)$  is 0. So if the node  $v_0$  is failed, it has an impact on the whole communication tree. Through the above analysis, it is known that the deeper of depth the node  $v_i$  is located at, the less impact on the whole tree.

According the Theorems 3 and 4, if one node is failed, we can calculate the number of its cascaded failure nodes in the case of  $l(v_i) \equiv m$ .

We obtain these Theorems in the case of  $l(v_i) \equiv m$  which has the certain specificity. However, if  $l(v_i)$  is different, we only depend on simulations to evaluate performance of the proposed algorithm. Next is simulation experiment. The proposed algorithm is implemented and validated in C# on Microsoft Visual Studio 2015.

### 5.2 Performance Parameters Comparison under Different Percentage of Failure Nodes

On Blockchain, due to the mobility, selfishness and malfunction of nodes, nodes may be failure. Node failure affects the efficiency of communication. Here is a problem—what percentage of failure nodes on Blockchain network does not affect communication efficiency. Below we analyze communication performance parameters with different percentage of failure nodes on Blockchain-based communication.

Figures 2, 3, 4, 5, and 6 show the simulation results of performance parameters under different percentage of failure nodes. The four different colour lines represent the result under different percentage of failure nodes. The X axis signs node number for 5, 10, 30, 50, 80, 100, 150, 200, 250, 300, 500 and 1000.

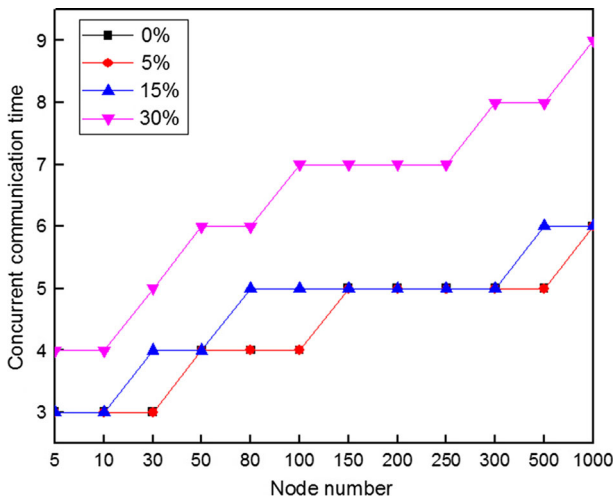


Fig. 2 Concurrent communication time  $f(t)$

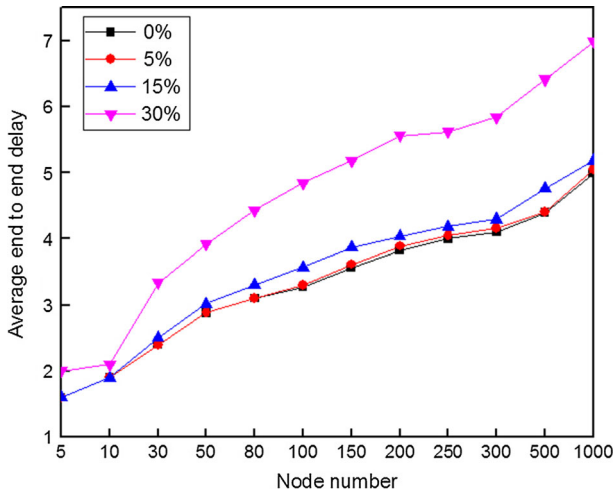


Fig. 3 Average end to end delay  $f_{AED}(t)$

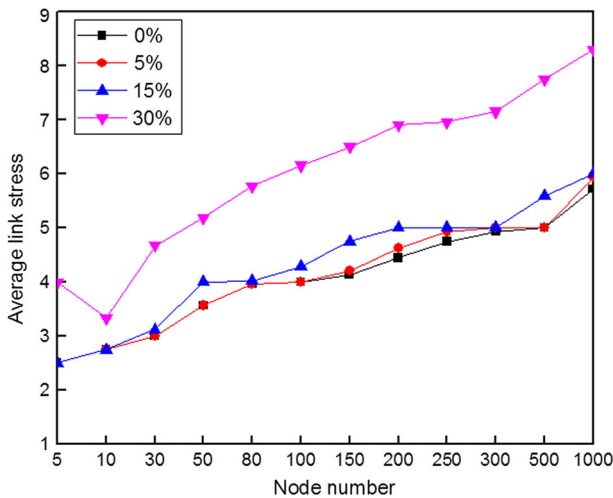


Fig. 4 Average link stress  $A_{ls}$

In actual communication, the few nodes have powerful capacity, and most of nodes are poor in transmitting, even have no ability to forward. Therefore, nodes whose communication link number value is 3, 2, 1, 0, are randomly selected in proportion of 5, 10, 45 and 40% respectively. In addition, Failure nodes are randomly selected in certain proportion 0, 5, 15, 30 of  $N$  respectively. Failure nodes whose  $l(v_i)$  is 3, 2, 1, 0 are randomly selected in proportion of 20, 30, 50 and 0% respectively.

Figure 2 shows the concurrent communication time  $f(t)$  under different percentage of failure nodes. The concurrent communication time, denoted by  $f(t)$ , is the most important parameter to measure communication efficiency.  $f(t)$  is the elapsed time consumed finishing the whole transmission. In Fig. 2,  $f(t)$  doesn't increase significantly along with the increasing of nodes  $N$ , and it is relatively flat and has a small increase. The reason is that

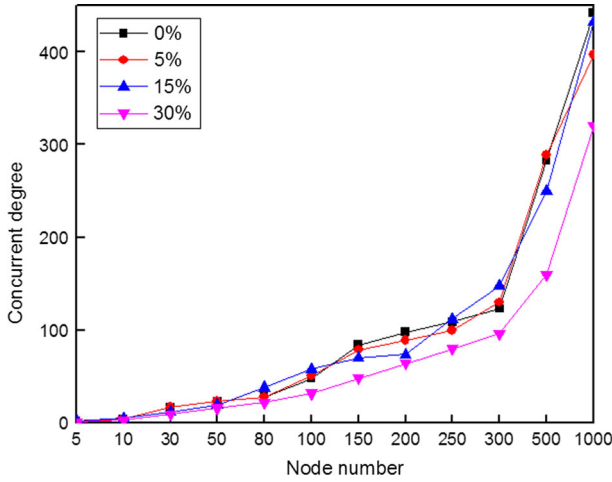


Fig. 5 Concurrent degree  $C_d$

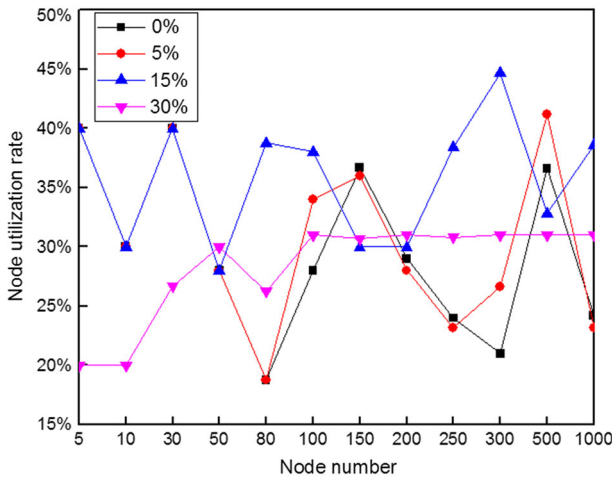


Fig. 6 Node utilization rate  $U_r$

the proposed data transmission scheme follows the multi-link concurrent communication tree model which means that every node makes its best effort to forward and all nodes who have received the data must forward data at the same time. In simulation, we only select 5% of  $N$  as the powerful nodes whose  $l(v_i)$  is 3. But only 5% powerful nodes can perform their amazing abilities to forward the block data to other nodes, and  $f(t)$  keeps a slight growth compared with the increasing of nodes  $N$ . Because in the communication tree established by the proposed data transmission scheme, powerful nodes are selected to locate in the top or the upper of the communication tree, so the concurrency and the efficiency is high. In addition, in the case that there are no failure nodes, that is, failure nodes account for 0% of  $N$ , this is the ideal situation and  $f(t)$  can reach optimal value, as described in paper [17]. From Fig. 2, it is clear that  $f(t)$  under 5% failure nodes is equal to the optimal value, and  $f(t)$  under 15% failure nodes is only slightly above the optimal

value. This reflects it does not matter to  $f(t)$  if 15% of  $N$  are failed in communication tree. However, if the failure nodes reach 30%,  $f(t)$  has a dramatic increase and it is prolonged by about 50% compared with the optimal value.

Figure 3 shows the average end to end delay  $f_{\text{AED}}(t)$  under different percentage of failure nodes. The average end to end delay, denoted by  $f_{\text{AED}}(t)$ , reflects the average delay spent by all nodes. If the average end to end delay  $f_{\text{AED}}(t)$  is lower, and the performance of the proposed data transmission scheme is better. From Fig. 3, the  $f_{\text{AED}}(t)$  has an increase with the increasing of nodes  $N$  for the whole. The fact that two lines under 0 and 5% failure nodes almost appear as one reflects the  $f_{\text{AED}}(t)$  does not increase in the case of 5% failure nodes. If 15% of nodes are failed, the  $f_{\text{AED}}(t)$  has a smaller increase. Yet the  $f_{\text{AED}}(t)$  under 30% failure nodes is significantly higher than that under other percentage of failure nodes. It means that if failure nodes account for 30%, the  $f_{\text{AED}}(t)$  is too long for the whole communication.

Figure 4 shows the average link stress  $A_{ls}$  under different percentage of failure nodes. The average link stress, denoted by  $A_{ls}$ , stands for the average route traffic load. To finish the transaction validation, the amount of communication tasks must be constant, so the more the routes, the less the average link stress. From Fig. 4, in the case of the same number of nodes, an increase in the node failure ratio leads to an increase in the  $A_{ls}$ . The  $A_{ls}$  is the highest under 30% failure nodes, and the routes are busiest if 30% of nodes on Blockchain are failed. In addition, the  $A_{ls}$  increases smoothly along with the increasing of nodes  $N$ .

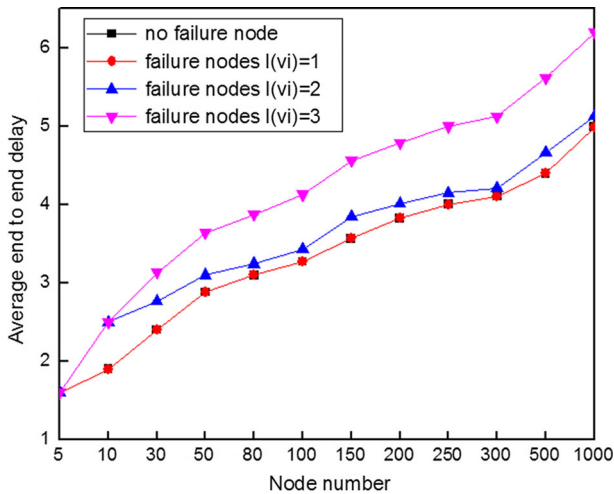
Figure 5 shows the concurrent degree  $C_d$  under different percentage of failure nodes. The concurrent degree, denoted by  $C_d$ , is an important performance parameter to measure concurrency. From Fig. 5, the  $C_d$  increases sharply with the increasing of nodes  $N$ . The more the nodes, the higher the concurrency. For the fixed number of nodes, the  $C_d$  under 30% failure nodes is smaller than that under other percentage failure nodes. The reason is that there is not enough nodes to participate in transmitting.

Figure 6 shows the node utilization rate  $U_r$  under different percentage of failure nodes. The node utilization rate, denoted by  $U_r$ , means how many nodes to be selected to undertake communication forwarding tasks. In Fig. 6, the  $U_r$  is between 20 and 45%, and varies in an irregular way with increasing of nodes  $N$ . In the case of 30% nodes failure, the  $U_r$  changes smoothly along with the increasing of nodes  $N$ . The reason is that failure nodes account for 30% and non-forwarding nodes may account for 40%, so there is no other nodes to be selected and the  $U_r$  is around 30%.

### 5.3 Performance Analysis under Failure Nodes with Different Transmitting Capacity

Fixed terminals (desktop computer, all-in-one machine, server, workstation, etc.) and mobile terminals (mobile phone, iPad, notebook, etc.) are regarded as nodes on Blockchain. Because of different nodes equipped with different hardware configuration, the nodes on Blockchain are heterogeneous in forwarding capacity. The failure and the invalidation of different nodes have different impact on the Blockchain validation. If powerful nodes are failed, that degrades communication performance, even aborts the communication. When weak nodes are failed, that may have a small impact on Blockchain performance. Next, we study how different nodes impose their failure influence on communication performance.

Figure 7 shows the average end to end delay  $f_{\text{AED}}(t)$  under failure nodes with different transmitting capacity. We assume that the proportion of failure nodes is 10%. We



**Fig. 7**  $f_{\text{AED}}(t)$  under failure nodes with different transmitting capacity

randomly select weak nodes ( $l(v_i) \equiv 1$ ), moderate nodes ( $l(v_i) \equiv 2$ ), powerful nodes ( $l(v_i) \equiv 3$ ) to get them failed respectively. Why don't we select nodes whose  $l(v_i)$  is zero as the failure nodes? Because these nodes located as the leaf nodes in communication tree, don't undertake forwarding tasks, even if they are failed, they only affect their own and don't impact the overall communication performance.

It is well known that there are no failure nodes on Blockchain, the communication performance is optimal and it is an ideal state. From Fig. 7, when all failure nodes whose  $l(v_i)$  is 1 are weak nodes, the  $f_{\text{AED}}(t)$  is close to or reach the optimal performance. When all failure nodes'  $l(v_i)$  is 2, the  $f_{\text{AED}}(t)$  has a much smaller increase than the optimal performance, and the increasing proportion of average end to end delay  $f_{\text{AED}}(t)$  is unstable, ranging from 4 to 31%. While powerful nodes whose  $l(v_i)$  is 3 are not available, the  $f_{\text{AED}}(t)$  has a significant increase, and it is prolonged by about 30%. So whether powerful nodes are failed or not, they are vital to the overall communication performance.

## 6 Conclusions

Based on the multi-link concurrent communication tree model, a data transmission algorithm considering node failure is proposed for giving the routing scheme on Blockchain. The proposed algorithm can improve the communication efficiency through maximizing the nodes' transmitting capacity. The performance of the proposed algorithm has been validated through theoretical proof and simulation experiments. The results show that the proposed algorithm still works effectively even though the failed nodes reach 15%. When the failed nodes account for 30%, the concurrent communication time, the average end to end delay and the average link stress have increased greatly. In addition, the failure of weak nodes has a relatively small effect on communication performance, and the failure of powerful nodes leads to poorer performance. In the actual communication, as long as individual powerful nodes must be safeguarded and well maintained and they are not failed, the performance of Blockchain-based communication will not degrade greatly. In future, how to select a backup node and how to replace the failure node is an issue worth of

research. Moreover, according to this new approach, we will do a deeper research on the real-world implementations for Blockchain.

**Acknowledgements** This work was supported by Natural Science Foundation of China (Grant: 5170715), Scientific Research Foundation of Education Bureau of Shaanxi Province (Grant: 15JK1571) and Science & Technology Brainstorm Project of Shaanxi Province (Grant: 2016GY132).

## References

1. Godsiff, P. (2015). *Bitcoin: Bubble or blockchain* (pp. 191–203). Agent and Multi-agent Systems: Technologies and Applications.
2. Kraft, D. (2016). Difficulty control for blockchain-based consensus systems. *Peer-to-Peer Networking and Applications*, 9(2), 397–413.
3. Swan, M. (2015). Blockchain thinking: The brain as a decentralized autonomous corporation. *IEEE Technology and Society Magazine*, 34(4), 41–52.
4. Eldred, M. (2016). Blockchain thinking and euphoric hubris. *IEEE Technology and Society Magazine*, 35(2), 27–27.
5. Zyskind, G., Nathan, O., & Pentland, A. S. (2015). Decentralizing privacy: Using blockchain to protect personal data. In *IEEE security and privacy workshops* (pp. 180–184).
6. Wilson, D., & Ateniese, G. (2015). From pretty good to great: Enhancing PGP using bitcoin and the blockchain. In *International conference on network and system security* (pp. 368–375).
7. Kyriotaki, K., Zamani, E., & Giaglis, G. (2015). From bitcoin to decentralized autonomous corporations. In: *International conference on enterprise information systems* (pp. 284–290).
8. Hurlburt, G. (2016). Might the blockchain outlive bitcoin? *IEEE Educational Activities Department*, 18(2), 12–16.
9. Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. <https://bitcoin.org/bitcoin.pdf>.
10. Reijers, W., & Coeckelbergh, M. (2016). The blockchain as a narrative technology: Investigating the social ontology and normative configurations of cryptocurrencies. *Philosophy & Technology*. <https://doi.org/10.1007/s13347-016-0239-x>.
11. Kumar, S., & Mehruz, S. (2016). Intelligent probabilistic broadcasting in mobile ad hoc network: A PSO approach. *Journal of Reliable Intelligent Environments*, 2(2), 1–9.
12. He, M., Zhang, Y. J., & Meng, X. W. (2015). Resource location flooding strategy of unstructured P2P for user requirements. *Journal of Software*, 26(2), 640–662.
13. Viana, D. (2016). Two technical images: Blockchain and high-frequency trading. *Philosophy & Technology*. <https://doi.org/10.1007/s13347-016-0247-x>.
14. Pai, V., Kumar, K., Tamilmani, K., Sambamurthy, V., & Mohr, A. E. (2005). Chainsaw: Eliminating trees from overlay multicast. In *International conference on peer-to-peer systems* (pp. 127–140).
15. Biskupski, B., Schiely, M., Felber, P., & Meier, R. (2008). Tree-based analysis of mesh overlays for peer-to-peer streaming. In *Ifip Wg 6.1 international conference on distributed applications and inter-operable systems* (pp. 126–139).
16. Liu, T. S., & Zhao, S. Z. (2007). *P2P Distributed Database System*. Beijing: Science Press.
17. Li, J., & Liu, T. S. (2010). Study on a P2P communication tree algorithm based on multi-link. *Journal of Northwest University*, 40(6), 970–974.
18. Li, J., Liang, G. Q., & Liu, T. S. (2017). A novel multi-link integrated factor algorithm considering node trust degree for blockchain-based communication. *KSI Transactions on Internet & Information Systems*, 11(8), 3766–3788.
19. Huang, J., Fan, X., Wan, M., Zhuo, Z., Yang, Y., & Chen, S. (2016). Stable cluster-based routing protocol for mobile ad hoc networks. *Journal of Beijing University of Aeronautics & Astronautics*, 42(11), 2332–2339.
20. Jia, G., Fu, X. D., Gao, T. Y., Chen, B. B., & Fan, H. B. (2014). A survey on the node failure handling of P2P networks. In *Control and decision conference* (pp. 4077–4082).
21. Ranga, V., Dave, M., & Verma, A. K. (2014). A hybrid timer based single node failure recovery approach for WSANs. *Wireless Personal Communications*, 77(3), 2155–2182.
22. Uwitonze, A., Huang, J., Ye, Y., & Cheng, W. (2017). Connectivity restoration in wireless sensor networks via space network coding. *Sensors*, 17(4), 902.
23. Boudries, A., Amad, M., & Siarry, P. (2016). Novel approach for replacement of a failure node in wireless sensor network. *Telecommunication Systems*, 65, 1–10.

24. Wang, H., Ding, X., Huang, C., & Wu, X. (2016). Adaptive connectivity restoration from node failure(s) in wireless sensor networks. *Sensors*, *16*(10), 1487.
25. Liu, X., & Medhi, D. (2017). Optimally selecting standby virtual routers for node failures in a virtual network environment. *IEEE Transactions on Network and Service Management*, *14*(2), 275–288.
26. Shahriar, N., Ahmed, R., Chowdhury, S., Khan, A., Boutaba, R., & Mitra, J. (2017). Generalized recovery from node failure in virtual network embedding. *IEEE Transactions on Network and Service Management*, *14*(9), 261–274.
27. Abdullah, S., & Yang, K. (2014). An energy efficient message scheduling algorithm considering node failure in IoT environment. *Wireless Personal Communications*, *79*(3), 1815–1835.



**Jiao Li** received her Ph.D. degree from Northwestern Polytechnical University, China, in 2017. She is a lecturer in the School of Computer Science, Xi'an Shiyou University, China. Her research interests include Blockchain communication, the theory and application of Blockchain.