Efficient Block Propagation in Wireless Blockchain Networks and Its Application in Bitcoin

Teng Long, Shan Qu, Qi Li, Huquan Kang, Luoyi Fu, Xinbing Wang, *Senior Member, IEEE*, and Chenghu Zhou

***Abstract*—As the supporting architecture of cryptocurrencies, blockchain is also showing its promising potential to be applied to mobile devices. Being WiFi supported, mobile devices nowadays are able to perform ad-hoc local communications in multi-hop manners, thus can potentially extend cryptocurrency service into areas without stable Internet connection. However, traditional blockchain architecture suffers from the intolerably high block propagation latency and communication cost which will be even trickier if adopted in wireless multi-hop networks (WMNs). This paper aims to reduce this latency and cost by leveraging node weights to distinguish the feedback speed of different nodes and constructing a low-length multicast tree to select a subset of nodes with higher feedback speed to participate in the block verification. Thus, block propagation is depicted by the minimum length multicast tree, intrinsically the Steiner Tree Problem. The primary challenge lies in that the WMNs only allow local communication and the distributed consensus protocol of the blockchain makes a predetermination of receiver nodes impossible. We design our algorithm via a ''toward source'' Steiner tree approach in favor of the distributed environment. Tree construction proceeds by progressively enlarging the searching areas until the accumulated weight of receiver nodes reaches a threshold. Our algorithm provably returns a tree length fairly close to that of the optimal Steiner tree with latency $O(\sqrt{n\log n})$ (where n represents the number of nodes in the network), the best so far. Furthermore, our algorithm is empirically validated to provide instructive insights for efficient block propagation with applications in Bitcoin network.**

***Index Terms*—Blockchain, Bitcoin, Wireless multi-hop networks, Distributed Steiner tree, Block propagation.**

Manuscript received April 25, 2021; revised July 27, 2021; accepted September 12, 2021. Date of publication September 16, 2021; date of current version December 9, 2021. This work was supported in part by the National Key R&D Program of China under Grant 2018YFB2100302, in part by NSF China under Grants 42050105, 61960206002, 61822206, 62020106005, 61829201, and 62002332, and in part by the Program of Shanghai Academic/Technology Research Leader under Grant 18XD1401800. Recommended for acceptance by Dr. Vojislav B. Misic. *(Corresponding author: Luoyi Fu.)*

Teng Long is with the School of Information Engineering, China University of Geosciences, Beijing 100083, China (e-mail: longteng@cugb.edu.cn).

Shan Qu, Qi Li, and Xinbing Wang are with the School of Electronic Information, and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: qushan@sjtu.edu.cn; liqilcn@sjtu.edu.cn; xwang8@sjtu.edu.cn).

Huquan Kang and Luoyi Fu are with the Department of Computer Science, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: kinghiqian@sjtu.edu.cn; yiluofu@sjtu.edu.cn).

Chenghu Zhou is with the Institute of Geographical Sciences, and Natural Resources Research, Chinese Academy of Sciences, Beijing 100101, China (e-mail: zhouch@lreis.ac.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TNSE.2021.3112670, provided by the authors.

Digital Object Identifier 10.1109/TNSE.2021.3112670

I. Introduction

THE blockchain is a peer-2-peer (P2P) mechanism for consensus establishment in distributed environments [1]–[3]. It is currently the supporting architecture of almost all decentralized cryptocurrencies, the most famous of which is the Bitcoin, a trendy electronic payment system with a total capitalization of more than one-fourth of a trillion dollars [4], [5].

The security of Bitcoin and other Bitcoin-like cryptocurrencies is guaranteed by Proof-of-Work (POW) consensus protocol whose effectiveness relies heavily on the computing power and the communication conditions of the user devices. For one thing, POW protocol requires the miners to solve cryptographic puzzles which have high requirements for the computing power of the devices. For another, after a block is generated by a miner, it has to be broadcast for verification with high efficiency, which requires the communication between user devices to be in good condition. However, these conditions limit the application of cryptocurrencies on wireless devices, as opposed to today's tendency that mobile devices play a crucial role in people's daily lives.

As the rapid development of the 5 G technique, it has been recognized that blockchain has a great potential to be applied to mobile terminals due to the following reasons [6], [7]. (i) Mobile devices nowadays are WiFi-supported, thus are able to perform, given their limited source, self-forming, local and distributed communications in a wireless multi-hop manner in areas with poor or no Internet connection; (ii) Such wireless multi-hop networks (WMNs) formed by mobile devices impose less reliance on the central infrastructure, thus cost less and are easier for deployment, ensuring the extendability of cryptocurrency service into wider areas without Internet or costly infrastructure deployment; (iii) The application of cryptocurrencies to mobile devices is also in line with the current tendency that rich functions can be implemented by smartphones.[1] Motivated by these benefits, a derivative of real blockchain-based innovations have attempted to incorporate the wireless multi-hop architecture during their continuous development. For example, several ongoing projects are directed toward merging the blockchain technology with mesh networking [8], [9], which aims to enlarge the applicability of bitcoin services by enabling the connection among

[1] The technologies that can support the application of blockchain on the mobile devices is already mature, such as the increasing of smartphone capacity and resources.

smartphones, onboard devices and other devices without the internet. Besides, [10] solved the problem of insufficient computing power and limited storage space for mobile devices by proposing a novel idea of diet nodes, which enables low-resource devices to fully verify subchains of blocks without having to pay the onerous price of a full chain download and verification. Furthermore, LocalCoin is also proposed [11], serving as a low-cost cryptocurrency payment that works with off-the-shelf mobile devices distributed geographically around in an ad-hoc manner.

A. Problem Statement

Inspired by these blockchain-innovations adopted to mobile devices, in this paper we also look into such scenarios, and focus on the most communication demanding procedure, i.e., the consensus stage, of the blockchain framework. In the consensus stage of the traditional Internet-backed blockchain network, block verification that is done by broadcasting the block to the entire network incurs huge *latency* (i.e., the consumed time to send the block from the source node to all receiver nodes) and *communication cost* (i.e., the amount of exchanged message during this process). From statistics, in extreme cases, the transaction takes 16 hours to complete, with block propagation alone taking up an order of ten thousand seconds [12]! Such latency problem will become even tricker if considered in an WMN, a location-sensitive network where packets are transferred by multiple successive nodes before reaching their destinations.[2]

In this work, we first attempt to reduce the latency and communication cost of the block propagation by reducing the number of nodes to verify blocks. Thus, a block is verified with only a subset of all the nodes in the network. As will be elaborated later (Section 3.1), this can be made applicable based on the idea of *lightweight architecture* adopted in [10] and [11]. Instead of merely considering the number of the nodes that participant in the block verification, we take the performances of nodes into consideration by distinguishing different nodes by node weight. A higher weight means that the node can feedback faster in past verifications, thus it has a higher probability to respond faster in the present verification. The feedback speeds vary from different users due to multiple factors including the difference of the performances of mobile devices and the difference of the communication environment in which the users locate in. This assumption allows us to select nodes with high performances during the block verification, which could also reduce the latency and communication cost of the block propagation process. By this way, the block propagation process becomes a multicast process instead of a broadcast one. Meanwhile, we also attempt to search for the optimal routing from the sender node to the selected receivers to further reduce the latency and communication cost. Consequently, enhancing the block propagation in WMNs is equivalent to solving the following optimization problem: *given the block holder (source miner, sender node) and a group of potential receiver nodes scattered across the network, how to select a part of nodes in the network as receivers and design an optimal routing that ameliorates the information transmission from source to receiver nodes*? From this point of view, the efficient block propagation in WMNs bears much resemblance to the search for optimal multicast routing, which can be depicted as the minimum length multicast tree problem. Intrinsically, this is the Steiner tree problem that aims to find a tree in a given network that spans a subset of terminals with the least total weight. And thus the latency corresponds to the running time of the multicast tree construction, and the communication cost is measured by the amount of exchanged messages during the tree construction. It has long proven that the Steiner tree problem is NP-hard [13], with enormous efforts made in seeking for various approximate solutions [14], [15].

Compared to the conventional approximate Steiner tree problem, our problem is more challenging under the blockchain settings. Firstly, the distributed characteristics only allow local communication among nodes. The lack of global topological information makes it difficult to build trees with a high efficiency. Secondly, unlike traditional wireless networks where receiver nodes can be predetermined, the receiver nodes who are able to participate in the block verification process are mainly decided by their node weight, making it impossible to know a priori the exact number of receiver nodes. Most existing solutions focus on minimizing the routing and time complexity in centralized surroundings, hence are incompatible with the distributed environment. No prior efforts have been made, to the best of our knowledge, on the approximate Steiner tree problem in wireless blockchain networks.

B. Our Inspiration: TST Construction

Recently, an excellent distributed algorithm called Toward Source Tree (TST) [16] has been proposed to address the Steiner tree problem in Wireless Sensor Networks. TST functions in an environment similar to ours, therefore may be particularly instructive for problem-solving. Out of such motivation, we would like to present a brief review of TST, which achieves high efficiency and low complexity at an affordable cost. TST is a three-step solution. The source first broadcasts a message and wakes up all the neighbor nodes it chooses. Each receiver node then chooses the closest neighbor node that has a shorter distance to the source node than the receiver node itself. Up till now, a temporary tree (possibly with cycles) can be constructed among all nodes. In the end, the cycles are eliminated from the temporary tree. TST generates tree length proportional to $\sqrt{m}$ with latency $O(\sqrt{n \log n})$ (where m and n denote the number of selected receivers and nodes in the network, respectively), the best among all existing approximate solutions for large multicast groups.

[2] While there are multiple extra reasons (e.g., the slow generation of blocks, the mining speed among different miners, block sizes, etc) that lead to the large latency of the blockchain, they are out of the scope of this paper. In other words, we focus on exactly the phase where a block mined by a rounder leader (miner) is about to be sent for verification.

C. Our Solution

We build our algorithm on TST Construction. Since TST does not require global network topology, the first challenge mentioned above is circumvented. As for the second challenge, we quantify both the feedback speed of each potential receiver and the total threshold of the nodes number needed for transaction validation with weights. The tree construction is terminated when the accumulated weight of receivers surpasses the threshold. While deferring the details to Section 4, here we present a sketch of our algorithm: the tree is constructed by iteratively searching receiver nodes within expanding radii. TST is first applied in each single searching process. The total node weight within current searching radius is then compared to $\mathcal{W}$ (which is defined as the goal weight to guarantee that there are enough nodes to participate in the verification) to check whether the stopping criterion is satisfied. If yes, the tree construction is completed. Otherwise, the algorithm enlarges the searching radius and repeats the whole process.

Our main contributions are summarized as follows.

- Design: We propose a novel algorithm named as TSTB algorithm based on a distributed Steiner tree approach. The algorithm optimizes the block propagation in wireless blockchain networks by partial engagement of nodes and optimal transmission routing.
- Analysis: We theoretically analyze the performance of our proposed TSTB algorithm, and results show that it realizes efficient block propagation with latency $O(\sqrt{n\log n})$, tree length $O(2.311\sqrt{m})$, and communication cost $O(n)$, the best among all existing techniques to optimize the block propagation in wireless blockchain networks.
- Application: In addition to simulation results, we validate the effectiveness of our algorithm with real applications in the Bitcoin network. It is shown that our algorithm could reduce both latency and communication cost with the same order of tree length as the optimal Steiner tree.

II. Related Literature

Before embarking on our analysis, we would like to give a review of related researches on wireless blockchain networks, performance metrics of blockchain, research techniques and the Steiner tree problem.

A. Wireless Blockchain Networks

Blockchain has been recognized as a revolutionary technique due to its capability of establishing trust in a complete decentralized manner [17]. It has been verified that Blockchain also has great potential to be applied in WMNs [3], [6], [7], [18]. Specifically, the work flow of blockchain mainly consists of four stages, i.e., *transaction request*, *consensus*, *state replication* and *reply to the client* [19]. Among them, the *consensus* refers to that after the transaction is requested, any node who received the request will be responsible for broadcasting the transaction to the whole network, then all miners

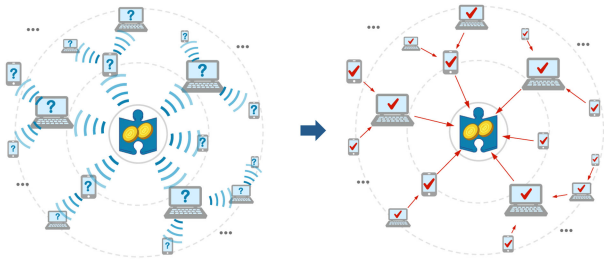


Fig. 1. An illustration on the consensus stage of blockchain.

should receive the transaction and are required to achieve an agreement to confirm the authenticity of the transaction, as illustrated in Fig. 1. From the communication perspective, the *consensus* is the most communication demanding procedure among all stages [1], [19], [20], which may be neglected in wired networks with stable communication conditions and advanced devices. However, for wireless networks, all nodes located in various areas should receive the transaction, which requires a significant number of relays, and thus incurs heavy latency and communication cost [19]. Such phenomenon greatly hinders the wide application of wireless blockchain networks.

To address this problem, we focus on the *consensus* stage and aim to propose an efficient block propagation mechanism. Specifically, we implement it from the following two aspects: (i) *Partial Participation:* we select partial nodes with high performance (e.g., good communication condition and computational capability) rather than all nodes to participate in the *consensus* stage, which is proven to be sufficient to verify the authenticity of the transaction (with detailed illustrations later); (ii) *Optimal Routing:* we design an efficient algorithm to search for the optimal routing so as to transmit the block from the sender node to the selected receiver nodes. By doing so, the latency and communication cost for wireless blockchain networks could be greatly reduced. During the abve process, we mainly consider the consensus and distributed characteristics of the blockchain. And it has been pointed out that lower communication cost and latency could also enhance other properties of the blockchain network, such as security and immutability [19].

B. Performance Metrics of Blockchain

Recently, there have been increasing efforts to investigate the performance of blockchain in terms of three main aspects, i.e., *decentralization, scalability and security* [21]. Compared with most networks, the blockchain is much more secure owing to the cryptography techniques, e.g., Hash algorithm and Elliptic Curve Cryptography. However, as the emergence of 5 G techniques, the scalability issues of more complex and larger-scale wireless scenes have arisen and greatly affect the development of blockchain [22]–[24], such as major public-chain platforms (i.e., Bitcoin and Ethereum). To improve the scalability of wireless blockchain networks, several metrics of blockchain have been analyzed, including *latency, communication cost* (of the transaction confirmation), *bootstrap time* and *throughout* [25]. Among them, latency and

TABLE I
PERFORMANCE COMPARISON OF VARIOUS RESEARCH TECHNIQUES

Properties	Network	Transaction Dimension	Latency	Cost	Path Length	Receivers
PBFT [30]	mesh network	unicast/broadcast	$O(n^2)$	$O(2n^2)$	/	Known
Raft [31]			/	$O(2n)$		
PoW [32]		broadcast	$O(n^2)$			
ELASTICO [28]	community graph	broadcast/multicast	$O(n \log k)$	$O(nc^3)$		
BC-IoT [33]	hierarchical graph	unicast/broadcast	$O(\frac{ck}{2})$	$O(c)$		
IRM-BC [29]	BA graph	unicast	$O(\frac{m \log n}{\log(\log n)})$	$O(\frac{mN_{con} \log n}{\log(\log n)})$	$O(\frac{m \log n}{\log(\log n)})$	
TST [16]	general graph	multicast	$O(\sqrt{n \log n})$	$O(n)$	$O(5.622\sqrt{m})$	
TSTB					$O(2.311\sqrt{m})$	Unknown

[1]The property *cost* refers to the communication cost of transaction confirmation, and the symbol "/" means that the work provides no theoretical analysis of the corresponding property.
[2]Parameter n denotes the number of nodes and parameter c denotes the number of communities or clusters in the blockchain network, parameter m represents the number of nodes participating in the transaction confirmation procedure, parameter N_{con} represents the default number of connections of a newly added node in BA graphs and k is a constant.

communication cost are the two most important performance metrics that have a significant impact on the user's quality of experience [26]. From statistics, the latency of Bitcoin is about several hours while Visa merely requires multiple seconds [27]. It is obvious that both the latency and communication cost have not achieve a satisfactory level in large-scale trading scenarios, which also receive the most attention among all performance metrics [26]. Our work exactly focuses on improving the performance of blockchain in WMNs in terms of latency and communication cost. Besides, many researches [28], [29] also take the path length into consideration, which refers to the total of the distances from the sender node to all receivers in the network. It is obvious that this metric could implicitly affect the latency and communication cost in blockchain networks.

C. Research Techniques

As previously mentioned, latency and communication cost are two important scalability bottlenecks of the blockchain. To address this problem, a multitude of techniques have been proposed, which mainly involve in the following three aspects [23], [26]. The first one [34], [35] focus on the on-chain design, including the network and data structure of the blockchain (such as increasing the blockchain size, dividing the data into a number of fragments to separately store), various consensus strategies and the corresponding varieties (such as PBFT [30], Raft [31], PoW [32] and so on). The second one [36], [37] move some complex computational tasks to an off-chain platform, e.g., payment channel, side chain and some cross-chain solutions. And the third one [28], [29], [33] consider the optimization of the block propagation, which is a kind of relatively new solutions and are desired in future scalable networks like wireless blockchain networks. For instance, the work in [28] proposes a new distributed agreement protocol, named ELASTICO, which uniformly partitions the network into several committees so as to improve the efficiency. Ali Dorri *et al.* [33] study the blockchain in Internet of Things (BC-IoT) and propose a hierarchical architecture consisting of smart homes. The block propagation procedure is also investigated in [29], which introduces a new protocol where the block owner is validated before the block is created in $Barábsi - Albert(BA)$ graphs. And more recent works [38], [39] are proposed to improve the propagation performance with empirical results. Our work is also of this third category. Despite of this similarity to the above methods, the techniques utilized are quite different. Specifically, previous works are dedicated to optimize the block propagation process by decomposing it into many sub-processes via clustering or selecting a leader, which requires prior knowledge and manual settings of the network. In constant, our focus here lies in an efficient block propagation mechanism based on the performance of nodes without the need for human intervention.

D. Steiner Tree Problem

The nature of our problem is analogous to the Steiner tree problem, which aims to find a tree with the least total weight that interconnects a given collection of points or terminals in Euclidean space [40]. This problem has been under intensive study for long. Shortest Path Heuristic (SPH) and Kruskal Shortest Path Heuristics (KSPH) [14] are two centralized Steiner heuristics proposed by F. Bauer and A. Varma in 1995. Despite minor differences in the manner of tree expansion, both require shortest path information. A year later, they introduced two distributed algorithms respectively based on SPH and KSPH for constructing multicast trees in point-to-point networks [41]. With the multicast tree construction demanding only local participations of multicast members and nodes, the distributed algorithms cut down the convergence time and even yield better results in most cases. Recent work in [15] investigates this problem in distributed computing and proposes distributed approximation algorithms for Steiner tree construction. Saikia *et al.* [42] consider a general Steiner tree problem named prize-collecting Steiner tree based on the primal-dual approach.

The performance comparison between state-of-the-art techniques and ours regarding six main properties is provided in Table I. It can be seen that our proposed TSTB algorithm

achieves the lowest latency and communication cost for the general wireless blockchain networks among all listed techniques. To conclude, the aforementioned work differs from ours as follows: (i) none of existing optimization techniques of block propagation has theoretically taken into consideration of the geographical features, which are primordial for the multi-hop-based block propagation in WMNs and could lead to improvements in both latency and communication cost (that will be illustrated in details in the remaining parts); (ii) all the above techniques require some manual settings, which could affect the fairness in real-world WMNs. Last but not least, the research techniques most related to ours may be [16]. It proposes the Toward Source Tree (TST) algorithm to address the Steiner tree problem in a distributed manner. However, this algorithm assumes the receivers are known in prior, and thus could not be applied to the block propagation procedure, where the receivers participating in the block verification process could not be predetermined.

III. Network Model and Problem Statement

A. Block Propagation in Blockchain Network

We model the whole blockchain network as a graph $G(V, E)$, with V and E representing, respectively, the sets of users and the edges, or connections between them. Here each user may either be a transaction initiator, a miner who successfully mines a block or a client that helps validate the transactions. For the graph's topology, instead of the pure online P2P network, as assumed in most of the prior arts, we consider the network to be a broad geographical area, over which the bitcoin users are distributed. The geographical network of choice stems from the fact that blockchain networks are nowadays evolving to industrial deployments with large numbers of geographically distributed nodes [9]. Each blockchain user is a power/resource-limited device that communicates with each other via wireless environment. Particularly, for any pair of users, they are able to communicate directly if and only if the Euclidean distance between them is within a fixed transmission range r. In other words, two users that are geographically out of this range have to rely on intermediate nodes that help forward packets via multiple hops. Here we choose r to be $O(\sqrt{\frac{\log n}{n}})$ to ensure the whole network connectivity [43]. While the communication ranges may vary among different devices, we represent them uniformly as a circle area so as to increase the tractability of our mathematical analysis.

Specifically, the block propagation of interest refers to the process of communicating a new block to the blockchain network. Recall that in the consensus stage of the traditional blockchain framework, the transactions is considered valid if and only if the corresponding block is verified by all users in the network. Thus ensuring efficient block propagation is crucial to reducing the latency and communication cost in mobile systems. Thanks to the idea of lightweight node, mobile devices are able to support the blockchain by storing a small fraction of blockchain data. Particularly, Diet node [10], a recently proposed idea, verified the possibility of applying blockchain architecture to mobile devices based on the idea of lightweight node. Furthermore, different from the traditional blockchain where all nodes need to be notified for verification, here we use a distributed blockchain architecture where each block is duplicated to a sufficient number of nodes but not to every node. Thus, even though every node stores only a subset of the full blockchain, the consensus about a valid transaction history can still be reached. Localcoin [11], another recently designed blockchain innovation, proved that the distributed blockchain is able to prove the ownership and avoid the double spending based on the social hardness which is the nature of the mobile ad-hoc network. Therefore, it is completely applicable to allow a sufficient number of nodes rather than all the nodes to participate in the block verification so as to reduce the latency and communication cost.

B. Problem Statement Regarding Block Propagation

In the wireless blockchain network of interest, the aforementioned process of block propagation is intrinsically equivalent to the construction of a multicast tree over the network. The source (S) of the tree is the miner holding a block that has successfully mined, while the rest of the nodes on the tree are constituted of either the receivers that help verify the block or the intermediate relays that help forward the block to the receivers. Recall that one-hop communications only occur among nodes that are within a certain transmission range r, thus an optimal block propagation routing is to design a corresponding multicast tree that can span the block from S to all receivers with the shortest tree length (where the tree length refers to the sum of the distances from S to all receivers in the tree). Intuitively, a longer tree length implies both a larger propagation latency and a higher communication cost. Therefore, an optimal multicast tree structure is of significance in ensuring the efficient block propagation.

For the above optimal multicast tree construction, unlike traditional wireless multicasting where the receivers are known a priori, who will be the receivers for verifying the block still needs to be determined. As the blockchain architecture is a distributed system composed of users with different levels of reliability, it is expected to choose those who are more trustful for block verification. In reality, the trustfulness/reliability of a blockchain user can be determined by his historical verification behavior. To elaborate, the following assumption associates each user's trustfulness level with a corresponding weight:

Assumption 1: We assume that each node has a weight $(0 \leq W_i \leq 1)$ to indicate the feedback speed of this node when confirming the correctness of messages. This weight is determined by the historical verification latency and the current communication condition of the node, which is influenced by both the performance of the user's mobile device and the communication environment the user locates in. Therefore, the value of this weight will change over time. In two extreme cases, for instance, a node with $W_i = 1$ means that this node i can respond with nearly no latency, while a node with zero weight indicates his incapability to provide any effective judgment.

Assumption 1 suggests that users with higher weights are more promising in reducing the block propagation latency.

Algorithm 1. Requests From Receivers

Input: The location and weights of all nodes in the network
Output: Request messages

while *total weight* W *is less than* $\mathcal{W}$**do**
  wake up the nodes with in the area with radius of R
  for *every* $N_i \in \{\mathcal{R}\}$**do**
    coverage range: $r = r_c$
    new hops: $h = 0$
    new weight: $w = W_i$
    path length: $l = 0$
    reset node sequence Sq and add N_i to Sq
    forward the request message to its neighborhood
    while *no respond is received***do**
      $r = 2 \cdot r$
      $h = 0$
      $l = 0$
      $w = W_i$
      reset node sequence Sq and add N_i to Sq
      forward the request message to its neighborhood
    for *every* $N_j \in Sq$**do**
      $mark[N_j] = true$
    $H = H + h$
    $L = L + l$
    $W = W + w$
  $R = 2 \cdot R$

Thus, it is natural to firstly choose those with higher weights to connect with as the potential receivers for block verification. To specify who will be the receivers on the tree, Assumption 2 designates a weight threshold as the condition of the choice.

Assumption 2: To reduce the tree length, we only choose nodes with a weight larger than W_c as receivers. Here W_c is the critical value of weight determined by such distribution function of node weight. Usually, for different weight distributions, the value of W_c that achieves the minimum tree length differs. For example, if all the nodes in the network have the weight of 1, then the critical value W_c is also 1.

Assumption 2 implies that, instead of informing all the nodes for completing the block verification, we can possibly shorten the verification time by only informing those with weights above W_c for participation in the verification. In other words, it becomes harder for nodes with slow feedback speed to participate in the block verification. However, as the feedback speed may change with time, each node has the possibility to participate in the verification in the future.

The last assumption below specifies when the block propagation comes to an end.

Assumption 3: In order to guarantee that there are enough nodes to participate in the verification, we introduce the goal weight $\mathcal{W}$. The definition of $\mathcal{W}$ is as follows. Given a propagating block and the nodes engaged in the block verification, $\mathcal{W}$ is a critical value to determine whether the verification process of the transaction in this block could be completed with these engaged nodes. If and only if the sum of weights of these nodes chosen for verification is larger than $\mathcal{W}$, the verification process is completed and the transactions in this block are verified. Intuitively, $\mathcal{W}$ reflects the expected security level of transactions within the block. It is obvious that $\mathcal{W}$ should be at least larger than the number of users included in this block. Thus, $\mathcal{W} \geq E(w) \cdot BC$, where $E(w)$ represents the expected node weight in the network, and BC denotes the block size. The reason we choose the total weight of the nodes as the cut-off condition instead of the total number of nodes is that the difference in the feedback speed of the nodes is taken into consideration. In order to reduce the latency, we give the nodes with faster feedback speeds a greater weight so that they have greater chances to contribute more to the total weight while being selected.

TABLE II
NOTATIONS AND DEFINITIONS

Notation	Definition
n	the total number of nodes after k^{th} searching
m	the total number of receivers after k^{th} searching
N_i	node i
$\{\mathcal{T}\}$	the collection of all nodes of the multicast tree
W_i	weight of node i
$\mathcal{W}$	goal weight of the tree construction
W_c	critical weight
k	searching times when the termination function is satisfied
R_i	i^{th} searching radius when the source tries to connect with other nodes $(1 \leq i \leq k)$
r	searching radius when a receiver tries to connect with its neighboingr receiver

Therefore, given the the weight W_i of a generic user i, the **termination function** of the tree construction should satisfy

$$\sum_{N_i \in \{\mathcal{T}\} \setminus S} W_i \geq \mathcal{W}, \tag{1}$$

where $\{\mathcal{T}\}$ is the collection of all receivers of the multicast tree. Thus far, the block propagation process has inherently fallen into a Steiner tree construction problem, which has long been proven to be NP-hard. Here, to construct a potential approximate tree, we need to address two major constraints: (i) the unpredictable number of receivers chosen via their weights; (ii) the limited network knowledge to both the source and other users due to the distributed nature of blockchain systems. We present the detailed algorithm for tree construction in the next section. For ease of expression, we summarize some major notations in Table II.

IV. Tree Construction Algorithm for Block Propagation

A. Anology With TST

In this section, we are dedicated to the tree construction for achieving efficient block propagation. Considering the geographical property of the network, our design of the tree construction gets inspiration from Toward Source Tree (TST) [16], a recently proposed distributed algorithm of approximate minimum-length multicast tree in wireless sensor networks. With provable ease of implementation and low computational complexity, TST bears an affordable cost on the suboptimality of tree length, provided with the limited availability of network knowledge to each node.

Algorithm 2. Requests Forwarding

Input: Request messages of all receivers and the location of all nodes
Output: Respond messages
for *every* N_i *received request message* **do**
  $dist = ||location\ of\ N_i - location\ of\ sender||$
  if $dist < r$ *then* **then**
    add N_i to Sq
    $H = H + 1$
    $w = w + W_i$
    newDist $=$ || location of N_i $-$ location of previous hop||
    $l = l + newDist$
    if $mark[N_i]$ *is true* **then**
      nodeSourceDist $=$ || location of N_i $-$ location of S||
      senderSourceDist $=$ || location of sender $-$ location of S||
      if $nodeSourceDist < senderSourceDist$ **then**
        N_i send respond message back to sender

To facilitate the understanding of our later tree construction, let us first briefly review the main idea of TST, which lies in the searching of relay at the side of each receiver, who, at each round, chooses a node with the shorter Euclidean distance than itself to connect to the source. This inverse searching facilities the decentralization where receivers from different locations can launch their respective searching at the same time. While incorporating this toward source searching manner, our tree construction differs from TST in the sense that the number of receivers is not predetermined, but instead depends on the termination function prescribed in (1). Consequently, it also remains unknown in advance how many users will be spanned when the tree construction is over. To tackle the issue, our tree construction process, as will be unfolded in detail shortly, can be seen as a repetition of such searching process for multiple times, with the searching at each time adopting a similar manner to that of TST. The first searching round originates from the source in a circular area with a certain radius, and each subsequent round of searching is executed in a larger ring area with an expanding searching radius than the previous one. Suppose that the searching has undergone k times when there are sufficient nodes satisfying the termination function. For ease of later description, we categorize the searching into two types, i.e., the *source search process*, distinguished from another searching called the *receiver search process*, which will be introduced in the sequel.

B. Three Phases of the Tree Construction

With the above analogy in mind, we are now ready to present our proposed tree construction, which we name as Toward Source Tree in Blockchain (TSTB). Basically, TSTB is composed of the following three phases.

Phase 1: Searching and Identifying Receivers in One Source Search Process. This phase is called *source search*. The source broadcasts its coordinate and the critical weight W_c in an area with radius R_i, the radius of the i^{th} source search. Every node received the message then broadcasts it into its neighbors within radius r until the receiving nodes exceed the bound R_i. Then every node in the area, upon receiving this message, will immediately confirm whether its weight is larger than W_c. If so, it is chosen as a receiver. This phase is shown in Fig. 2(a), with green nodes representing the receivers.

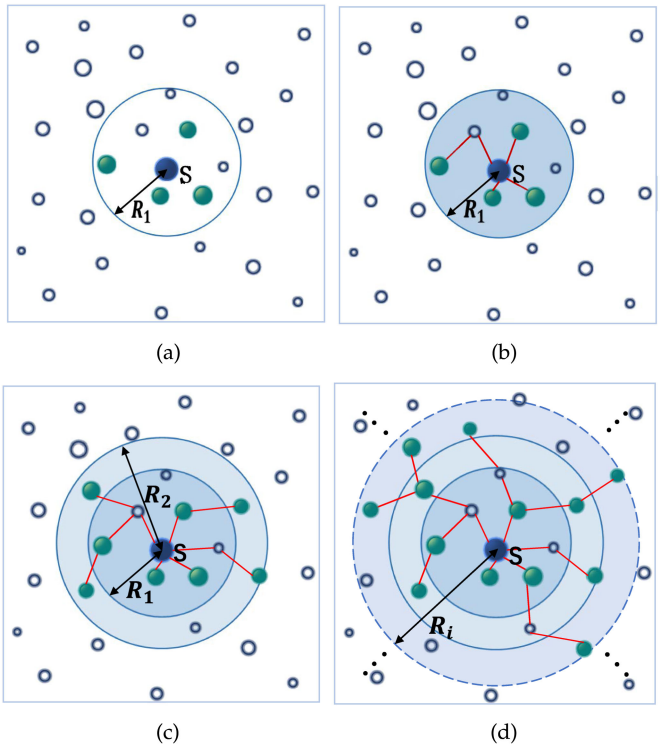


Fig. 2. Steps of the proposed tree construction algorithm TSTB. (a) Broadcast message in an area of radius R_1 and wake up receivers. (b) Receivers connect with neighboring receivers. (c) Expand search radius and repeat the first two phases. (d) The process ends until the search is repeated k times and the termination function is satisfied.

Phase 2: Building Multicast Tree in One Source Search Process. In this phase, each receiver chooses another receiver that has a shorter Euclidean distance to the source than itself to connect with. If no such receiver can be found, the receiver connects with the source directly. Then a *temporary tree*, which is an intermediate state of TSTB, is formed with all the receivers contained in this area. By finding the minimum-length path between each directly connected node pairs in the temporary tree, the construction of multicast tree in this area is completed.

The receivers contact with each other by sending three types of messages. Firstly, a *request message* is sent by each receiver in an area with radius r, which is initially set as $r_c = O(\sqrt{\frac{\log n}{n}})$ (r_c is the initial value of r as r may grow bigger if necessary). Request message is a set of {*sender ID, sender location, sender weight, coverage range* r_c*, node sequence, total hop H, path length L*}. It aims to ask for connection establishment with other neighboring receivers. Note that the location of all nodes could be derived by existing localization techniques based on WiFi like [44], and represented with the

coordinate in the same coordinate system. The pseudocode of this process is provided in Algorithm 1. For ease of description, we use node set $\{\mathcal{R}\}$ to represent the collection of all the receivers within the i^{th} source search area, and N_i representing node i.

Secondly, nodes who received request message in this area then compare their locations with the sender's location. Those nodes who are closer to the source then add themselves into the node sequence and also broadcast the request message into their neighborhood until another receiver or a node that is already marked receives this request message. We mark a node when it is chosen as a relay and becomes a part of the multicast tree to avoid the formation of cycles. Then the receiver or the marked node responds a *respond message* which contains {*sender id, respondent id, node sequence, total hop H, path length L*} back to the sender. If the sender does not receive any respond message in a certain time delay, it will expand the searching radius to $2r$ until the searching radius reaches the upper limit that the device can support.[3] We name this process as *receiver search*. Algorithm 2 summarizes how nodes deal with the request message in the above way.

Finally, when a sender receives more than one respond message in one search, it sends a *confirm message* to the closest receiver who sent respond message to it through the sequence of nodes (we name it the *minimum length path*) recorded in this respond message. In this way, the sender and this closest receiver are considered to be connected. After all the nodes have sent the *confirm message*, the construction of the multicast tree in the area with R_i is finished. This phase is shown in Fig. 2 (b), with red lines representing the established connections.

Phase 3: Weight Calculation and Searching Area Expansion. Recall again that the number of receivers for the block cannot be predetermined, but jointly depends on the goal weight $\mathcal{W}$ and the distribution function of node weight. Therefore, Phase 3 repeats Phases 1 and 2 k times with radius of source search double each time ($R_i = 2^{i-1}R_1, 0 < i \leq k$) until the termination function in (1) is satisfied.

In this phase, we calculate the sum of weights of all the receivers in multicast group and compare it with $\mathcal{W}$. If the sum is larger, the construction of the multicast tree is finished. If not, however, we double the search radius and repeat the first two phases until the k^{th} search satisfies (1), as illustrated in Fig. 2. In this figure, the size of each node means its authority (weight). The blue node in the center represents the sources (miners) ready for block broadcasting while the green ones are the receivers of the block. The empty nodes can serve as either the intermediate nodes or relays. Nodes connected into the tree are reflected by the red lines. Note that receivers must have larger sizes (weights) than relay nodes. Nevertheless, after the search area is expanded, the construction has a difference. Because the tree in the circular area with the radius R_{i-1} has been constructed, the newly searched nodes located in the circular area with a radius between R_{i-1} and R_i have to connect to any node on the tree that has been constructed within the circular area with the radius of R_{i-1}. Then every newly searched node (nodes in the area between R_{i-1} and R_i) chooses the closest receiver within R_{i-1} as its "source" and repeat the first two phases. Thus, no matter how big the value of $\mathcal{W}$ is, our algorithm always satisfies the termination function. (In the extremely large case of W, our algorithm will terminate after searching all nodes in the network.)

[3] In an area with high user density, however, this phenomenon does not usually occur. Since a node can be woken up in Phase 1, it must be located within the communication range of at least one node.

C. Rationality of the Tree Construction

When the source searching process is repeated, we face a problem of how to connect the newly searched nodes with the tree already constructed. In Phase 3, we point out that the newly searched nodes choose the closest receiver in the last search area to connect with. To ensure the rigor of our algorithm, the rationality of our connecting solution is examined in Lemma 1.

Before we proceed, we have a few words on notations: throughout the paper we assume that every user (including the miner) which is seen as a node in the network is located in a polar coordinates. The coordinate of the miner is at the origin $S(0,0)$, while each of the other nodes have a coordinate of (r,θ), with r representing the distance from the node to the source and θ being the directional angle. Thus, we use $A(r,\theta)$ to represent the coordinate information of a generic node A. Also, we let $L(A,B)$ denote the length of the minimum length path between nodes A and B, while $||AB||$ represents the Euclidean distance between them.

Lemma 1: For the nodes within the ring area with the radii between R_{i-1} and $R_i (i \leq 1)$, choosing the closest receiver within the area with the radius of R_i to connect with is the best solution, which returns the smallest upper-bound of the tree length.

Proof: Let us consider a node A in the specified ring area. Assume that it has two optional receivers V_1 and V_2, while the Euclidean distance satisfies $||AV_1|| \leq ||AV_2||$. Then, to prove Lemma 1 is equivalent to proving

$$\max\{L(A,V_1)\} \leq \max\{L(A,V_2)\}.$$

We prove the above condition by induction. In doing so, we need to consider the case that for $\forall A(a,0) \neq S(0,0)$, the tree length of constructing a tree to the source $S(0,0)$ should satisfy $\max\{L(A,S)\} \leq 2a$ (*Assumption a*). Notice that any relay chosen by node A must belong to the collection $\{(r,\theta)|r \leq a, r \leq 2a\sin\theta\}$, because, according to TSTB, the relay must be closer to the source than itself. Now, by induction on the number of relays, we will prove that *Assumption a* always stands for any number of relays.

Step 1: We first consider the case where there is only one relay $B_1(r_1,\theta_1)$ between nodes A and S (the source). It means that

$$\begin{aligned} L(A,S)_1 &= ||AB_1|| + ||B_1S|| \\ &= \sqrt{a^2 + {r_1}^2 - 2ar_1\sin\theta_1} + r_1 \\ &\leq \sqrt{a^2 + {r_1}^2 - ar_1} + r_1 \\ &\leq 2a. \end{aligned}$$

Thus, *Assumption a* holds when the number of relay is one.

Step 2: Then, we assume that *Assumption* a still holds when there are k relays existing between nodes A and O. Equivalently, it means that $L(A,S)_k \leq 2a$. Based on the hypothesis, we turn to the case where there are $k+1$ relays. We assume $B_{k+1}(r_{k+1}, \theta_{k+1})$ is the closest relay to node A. Then we want to prove *Assumption* a also stands. Looking into $L(A,S)_{k+1}$, we get

$$\begin{aligned} L(A,S)_{k+1} &= L(B_{k+1},S) + ||AB_{k+1}|| \\ &\leq 2r_{k+1} + \sqrt{a^2 + {r_{k+1}}^2 - 2ar_{k+1}\sin\theta_{k+1}} \\ &\leq 2a. \end{aligned}$$

Consequently, by virtue of induction [45] we can know that the upper bound of the tree length from node $A(a,0)$ to node $S(0,0)$ is $2a$ no matter how many intermediate relays are in between. Then because the closer the two nodes are, the smaller the value of a is. It is obvious that connecting with the closest receiver can absolutely lead to the smallest upper bound of tree length. This completes our proof. ■

Lemma 1 tells that we are always able to construct a multicast tree regardless of how many times the source searching process is repeated. Thus, no matter how big the value of $\mathcal{W}$ is, our TSTB algorithm can always satisfy the termination function (the largest $\mathcal{W}$ means all the blockchain users are spanned into the tree.).

Remark: In our algorithm, we give every node a *mark* to represent that it belongs to the collection $\{\mathcal{T}\}$ that contains nodes on the tree. Every node stops expanding the searching radius r as soon as it has connected with a marked node, Thus there is only $n-1$ edges constructed, making sure that the multicast tree constructed by TSTB form a tree topology.

V. Performance Analysis

Recall that tree length is an important metric that implicitly affects the latency and communication cost of the transaction confirmation process. In this section, we first analyze the expected tree length of the proposed TSTB algorithm (Theorem 1), based on which we present the performance evaluation on the other two explicit metrics, i.e., latency and communication cost over the blockchain network (Theorem 2 and Theorem 3).

A. Performance Analysis on Tree Length

This subsection analyzes the tree length of the constructed multicast tree with the proposed TSTB algorithm. Particularly, our tree length analysis consists of the following four parts: (i) we first calculate the expected value of the temporary tree length under the uniform distribution of node locations (Lemmas 2 and 3); (ii) we generalize the above results of the temporary tree length to the situation with a general function $f(r,\theta)$ of nodes' geographical distribution (Lemma 4); (iii) we quantify the relationship between the tree length and the critical weight W_c (Lemma 5); (iv) we finally derive the tree length of the constructed multicast tree, which is actually approximately equal to the temporary tree when the network size is large enough (Theorem 1).

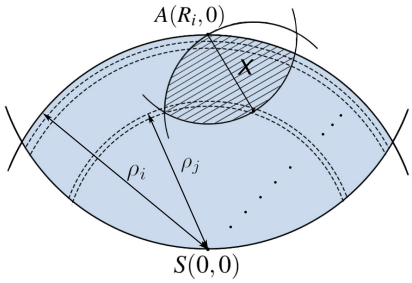


Fig. 3. The average length of a receiver with coordinate $(\rho_i, 0)$ connecting with its neighbour receiver.

1) Temporary Tree Length in Uniform Distribution: We start with our analysis of the expected tree length of the temporary tree when nodes are uniformly distributed, with the result summarized in Lemma 2. This part of calculation lays the foundation of the general situation.

Lemma 2: If nodes are uniformly distributed in a circle with area of $\pi R^2 = 1$, the expected length of the temporary tree spanning m receivers is upper bounded by $C\sqrt{m}$, where $C = 2.311$.

Proof: Prior work [46] analyzes the lower bound of the number of relay nodes that are engaged in the transmission process from one source to m receivers, where the network is a square of unit value and the transmission range is fixed. Based on its method, now we explore the results in a circle network under our proposed TSTB algorithm. We assume that the radius of the area is R. The length of the temporary tree in this area is discussed below. And we assume the number of receivers in this area is m. We divide the circle into m rings with equal area. Denoting the i^{th} ring's radius as ρ_i $(1 \leq i \leq m)$, and obviously $R_m = R$. Then we have

$$\pi\rho_i^2 = \frac{\pi R^2}{m} i \Rightarrow \rho_i = \sqrt{\frac{i}{m}} R.$$

Since the nodes are uniformly distributed in a circular area, the direction angle θ has nothing to do with the average tree length. Hence the probability that a node is in the i^{th} ring is $\frac{1}{m}$.

Fig. 3 shows how a node with coordinate $(\rho_i, 0)$ chooses its neighboring receiver. As we can see from Fig. 3, the neighboring receiver with the coordinate (ρ_j, θ) $(j < i)$ which could be chosen as the next hop of receiver $A(\rho_i, 0)$ must be located in the blue area. Thus, the coordinates of the next hops (the blue area) should satisfy

$$\begin{cases} \rho_j \leq \rho_i, & \frac{\pi}{6} \leq \theta \leq \frac{5\pi}{6}; \\ \rho_j \leq 2\rho_i \sin\theta, & 0 \leq \theta \leq \frac{\pi}{6}, \frac{5\pi}{6} \leq \theta \leq \pi. \end{cases}$$

So we calculate the expected tree length by summing all the expected length between node A and the neighboring node in the blue area. We assume that a receiver is chosen as the next hop if and only if there is no other receivers in the shadow

area in Fig. 3. Let p_{j1} be the probability that a receiver in ring $j(j \leq i)$ is chosen as the next hop, and S_{shadow} is the area of the shadow while S_{total} is the area of the whole circle area with radius of R, then

$$p_j = \Pr(\text{the shadow area is empty}) = \left(1 - \frac{S_{\text{shadow}}}{S_{\text{total}}}\right)^{m-1}.$$

From [47], S_{shadow} satisfies $(\frac{\pi}{3} - \frac{\sqrt{3}}{4})X^2 \leq S_{shadow} \leq (\frac{2\pi}{3} - \frac{\sqrt{3}}{2})X^2$ in Fig. 3. With the total area being πR^2, p_{j1} satisfies

$$p_{j1} \leq \left(1 - \frac{aX^2}{\pi R^2}\right)^{m-1}, \tag{2}$$

where $a = \frac{\pi}{3} - \frac{\sqrt{3}}{4}$. Since the blue area, as can be seen from Fig. 3, is symmetric about the straight line $r \sin\theta = \frac{\rho_j}{2}$, X and ρ_j are equivalent when calculating the expected tree length. Therefore, for simplicity of calculation, we use ρ_j instead of X in the expression of p_{j1}. Thus, (2) can be further reduced to

$$p_{j1} \leq \left(1 - \frac{a{\rho_j}^2}{\pi R^2}\right)^{m-1}.$$

Then we denote p_{j2} as the probability that there is at least one node exists in the ring area j with outer radius ρ_j. Obviously,

$$\begin{aligned} p_{j2} &= 1 - p(\text{all } n-2 \text{ nodes exist outside ring } j) \\ &= 1 - \left(1 - \frac{S_{\text{ring j}}}{S_{\text{total}}}\right)^{m-2} \\ &\leq 1 - \left(1 - \frac{1}{2\ m}\right)^{m-2}. \end{aligned}$$

With p_{j1} and p_{j2}, the expected length results from summing over the distances between every possible relay and $(R_i, 0)$ given all possible situations of R_i, i.e.,

$$\begin{aligned} E(L_v) &= m \cdot \frac{1}{m} \sum_{i=1}^{m} \sum_{j=1}^{i} p_{j1} \cdot p_{j2} \cdot \rho_j \\ &\leq \sum_{i=1}^{m} \sum_{j=1}^{i} \left[1 - \left(1 - \frac{1}{2\ m}\right)^{m-2}\right] \cdot \left(1 - \frac{a{\rho_j}^2}{\pi R^2}\right)^{m-1} \cdot \rho_j. \end{aligned}$$

When m is sufficiently large, by virtue of the formula $\lim_{x\to\infty}(1 - 1/x)^x = e^{-1}$ [48], $E(L_v)$ can be further derived as

$$\begin{aligned} E(L_v) &\leq \sum_{i=1}^{m} \sum_{j=1}^{i} \left(1 - e^{\frac{m-1}{2\ m}}\right) \cdot e^{-\frac{aj(m-1)}{\pi m}} \cdot \sqrt{\frac{j}{m}} R \\ &\leq \frac{(1 - e^{-\frac{1}{2}})R}{\sqrt{m}} \sum_{i=1}^{m} \sum_{j=1}^{i} e^{-\frac{aj}{\pi}} \cdot \sqrt{j}. \end{aligned} \tag{3}$$

Inequality (3) suggests that the upper bound of the expected tree length is determined by the double summation $\sum_{i=1}^{m} \sum_{j=1}^{i} e^{-\frac{aj}{\pi}} \cdot \sqrt{j}$, which is provably upper bounded by $(\frac{\pi}{2a\sqrt{a}} + \frac{\pi}{4a} \cdot \sqrt{\frac{\pi}{2a}} \cdot e^{-\frac{\pi}{2a}}) \cdot m$. We defer the corresponding detailed proof to our supplementary file. Substituting this result into Inequality (3), we can rewrite $E(L_v)$ as

$$\begin{aligned} E(L_v) &= \frac{(1 - e^{-\frac{1}{2}})R}{\sqrt{m}} \sum_{i=1}^{m} \sum_{j=1}^{i} e^{-\frac{aj}{\pi}} \cdot \sqrt{j} \\ &< \frac{(1 - e^{-\frac{1}{2}})R}{\sqrt{m}} \sum_{i=1}^{m} \left(\frac{\pi^2}{2a\sqrt{a}} + \frac{\pi}{4a} \cdot \sqrt{\frac{\pi}{2a}} \cdot e^{-\frac{\pi}{2a}}\right) \\ &= \frac{(1 - e^{-\frac{1}{2}})Rm}{\sqrt{m}} \left(\frac{\pi^2}{2a\sqrt{a}} + \frac{\pi}{4a} \cdot \sqrt{\frac{\pi}{2a}} \cdot e^{-\frac{\pi}{2a}}\right) \\ &< C \cdot R\sqrt{m}, \end{aligned} \tag{4}$$

where $C = 4.096$ is a constant. When we normalize the area by letting $R = \frac{1}{\sqrt{\pi}}$, the expected tree length in the first search process is upper bounded by $E(L_v) \leq 2.311\sqrt{m}$. ■

Remark: The temporary tree of TSTB returns the expected tree length proportional to $\sqrt{m}$, the same as that of TST. However, the proportionality factor is reduced from 5.622 to 2.311. In fact, this is not to say that our algorithm has a better performance, but the upper bound we derive is tighter. (As we will show in later simulation results, this upper bound can be even smaller.)

Recall that according to Phase 3 in TSTB, we have to repeat this search process k times until the termination function is satisfied. Then Lemma 3 suggests that no matter how many times we repeat the search process, the conclusion above remains unaffected.

Lemma 3: The expected tree length of temporary tree in the area with radius R_k is equal to the expected tree length, if we firstly construct a temporary tree in area with radius R_{k-1}, followed by the construction of a temporary tree from the circular zone between circle R_{k-1} and R_k to circle R_{k-1}.

Proof: We assume $L(\mathcal{X}, \mathcal{Y})$ is the tree length of the temporary tree constructed by method $\mu(\mathcal{X}, \mathcal{Y})$ of tree construction based on TSTB algorithm. Here $\mathcal{X}$ is the set of all receivers within radius R_{k-1}, $\mathcal{Y}$ is the set of all receivers within the ring area between radii R_{k-1} and R_k, and $\mathcal{Z} = \mathcal{X} \cup \mathcal{Y}$. Let $dist(\mu(\mathcal{X}, \mathcal{Y}))$ be the tree length of the tree construction method $\mu(\mathcal{X}, \mathcal{Y})$. Therefore, to prove Lemma 3, it is equivalent to proving the following equation:

$$L(\mathcal{Z}, S) = L(\mathcal{X}, \mathcal{Y}) + L(\mathcal{Y}, S).$$

We introduce three different methods, denoted as μ_1, μ_2 and μ_3, respectively. Suppose that

$$\begin{aligned} \mu_1 &= \arg L(\mathcal{X}, \mathcal{Y}) \quad \Rightarrow L(\mathcal{X}, \mathcal{Y}) = dist(\mu_1(\mathcal{X}, \mathcal{Y})); \\ \mu_2 &= \arg L(\mathcal{Z}, S) \quad \Rightarrow L(\mathcal{Z}, S) = dist(\mu_2(\mathcal{Z}, S)); \\ \mu_3 &= \arg L(\mathcal{Y}, S) \quad \Rightarrow L(\mathcal{Y}, S) = dist(\mu_3(\mathcal{Y}, S)). \end{aligned}$$

Since

$$\begin{aligned} dist[\mu_2(\mathcal{Z}, S)] &= dist[\mu_2(\mathcal{X}, \mathcal{Y})] + dist[\mu_2(\mathcal{Y}, S)] \\ &\geq dist[\mu_1(\mathcal{X}, \mathcal{Y})] + dist[\mu_3(\mathcal{Y}, S)], \\ dist[\mu_2(\mathcal{Z}, S)] &\leq dist[\mu_1(\mathcal{X}, \mathcal{Y})] + dist[\mu_3(\mathcal{Y}, S)], \end{aligned}$$

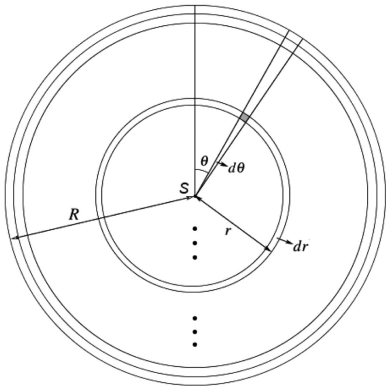


Fig. 4. Using the thought of calculus to calculate the tree length in general distribution of nodes' locations.

therefore, it follows that

$$dist[\mu_2(\mathcal{Z}, S)] = dist[\mu_1(\mathcal{X}, \mathcal{Y})] + dist[\mu_3(\mathcal{Y}, S)].$$

Equivalently, it means $L(\mathcal{Z}, S) = L(\mathcal{X}, \mathcal{Y}) + L(\mathcal{Y}, S)$.

Therefore, the expected tree length while building a tree under multiple search processes within a radius R is actually the same as building a tree directly within this area. That is to say, if the area is a unit circle, Phase 3 will not increase the expected tree length at all! As a result, the expected tree length of the whole process is also upper bounded by $E(L_v) \leq 2.311\sqrt{m}$. ■

2) Temporary Tree Length in General Distribution: With the calculation of the expected tree length in uniform nodal geographical distribution, we then calculate the tree length in general situation, as depicted by Lemma 4.

Lemma 4: Assume that all the nodes are independently distributed with a density function $f(r, \theta)$ in the circular area of $\pi R^2 = 1$. Then the expected length of the temporary tree spanning m receivers is upper bounded by $c\sqrt{m}\int_0^{2\pi} d\theta \int_0^R [r \cdot \sqrt{f(r,\theta)}]dr$, where $c = 2.311$.

Proof: Under the polar coordination, we divide the uniform circle into small enough pieces, each with an area of $r \cdot dr \cdot d\theta$, as illustrated in Fig. 4. As a result, in each small piece we can treat the nodes as being uniformly distributed. Then we can calculate the expected value of the tree length by taking the integral with respect to r and θ over the whole region. Here we assume the density function of node location as $f(r, \theta)$, which is a function of radius r and the direction angle θ. Thus, we have

$$\begin{aligned} E(L_v) &= \iint \left[\frac{d\theta}{2\pi} \cdot C \cdot r \cdot \sqrt{mf(r,\theta)\pi r^2} \right. \\ &\quad \left. - \frac{d\theta}{2\pi} \cdot C \cdot (r - dr) \cdot \sqrt{mf(r,\theta)\pi(r-dr)^2} \right] \\ &= \frac{C}{\sqrt{\pi}} \iint [r^2 - (r-dr)^2] \cdot \sqrt{mf(r,\theta)} \cdot dr \cdot d\theta \\ &= c\sqrt{m} \int_0^{2\pi} d\theta \int_0^R [r \cdot \sqrt{f(r,\theta)}]dr, \end{aligned}$$

where $c = 2.311$ which is equal to the conclusion in uniform distribution. ■

We can see from the calculation, the expected tree length is related to the distribution function $f(r, \theta)$. This is reasonable because for some extreme distributions, the tree length may therefore increase significantly. If the distribution of the users can be detected by some means before the construction of the multicast tree, we can estimate the expected length of the tree through the above calculations.

3) Relations With Critical Weight W_c: Recall that each blockchain user is rendered a weight, indicating its feedback speed in block verification. Since the number of receivers is dependent on the weight, the distribution of nodes' weight is supposed to have impact on the tree length. Obviously, if there is a larger fraction of nodes with high weights in the blockchain network, the expected tree length will be reduced because the termination condition can be achieved without too many rounds of searching. However, in the extreme case where the weight of every user in the network is zero, the expected tree length will tend to the infinity. Therefore, it is necessary to explore the determination of the critical weight W_c, and find out how it affects the tree length.

Lemma 5: Assume that the distribution of node weight in the network is based on the density function $f(w)$, where $0 \leq w \leq 1$. Then there must be a $W_{c_0} = 0$ or $W_{c_0} = arg\{W_c : 2W_c \int_{W_c}^1 f(w)dw - \int_{W_c}^1 wf(w)dw = 0\}$ to minimize the expected temporary tree length $E(L_v)$. Specially, if the weight is uniformly distributed, then $W_{c0} = \frac{1}{3}$.

Proof: Obviously, the expected tree length is positive for $\forall W_c \in [0, 1]$. There must be an optimal $W_c \in [0, 1]$ so that the tree length reaches a minimum. Now we prove that the optimal values W_{c_0} satisfy the conclusion drawn in the lemma. The proof is divided into two basic parts.

Part 1: The relationship between goal weight $\mathcal{W}$ and network size n.

Assume that after k^{th} search, the termination function in (1) is satisfied, thus the sum of the weight of nodes in the multicast tree is bigger than the goal weight $\mathcal{W}$. Since the $(k-1)^{th}$ search must satisfy that the summation of all weights in the multicast tree by then is no larger than $\mathcal{W}$. And the relationship between R_k and R_{k-1} is $R_k = 2R_{k-1}$, which means at most the summation of weights grows four times in the last expansion. Therefore, we get the upper bound of the summation of weights:

$$\sum_{N_i \in \{R\}} W_i \leq 4\mathcal{W}.$$

The sum of the weights $\sum_{N_i \in \{R\}} W_i$ of all receivers is equal to the number of receivers $m = n\int_{W_c}^1 f(w)dw$ multiplied by the mean of an receiver weight. So, we have

$$\begin{aligned} \sum_{N_i \in \{R\}} W_i &= n \int_{W_c}^1 f(w)dw \cdot \frac{\int_{W_c}^1 wf(w)dw}{\int_{W_c}^1 f(w)dw} \\ &= n \int_{W_c}^1 wf(w)dw, \end{aligned}$$

where n is the number of all nodes in the searching area which is large enough. Therefore, the total weight of the multicast tree should be in the order of $O(n)$. Then, in order to facilitate the determination of W_c, we assume that

$$\sum_{N_i \in \{R\}} W_i = n \int_{W_c}^{1} w f(w) dw = 4\mathcal{W}.$$

Rearranging the equation, we get

$$n = \frac{4\mathcal{W}}{\int_{W_c}^{1} w f(w) dw}.$$

Part 2: Determination of the optical critical weight W_c.

Next, we express the expected tree length as a function of W_c in order to find the value of W_c that leads to the smallest tree length. We assume that the total number of nodes n when the tree construction terminates is proportional to the square of the radius of the area R, i.e., $R = O(\sqrt{n})$. Recall the conclusion in Lemma 2, the expected tree length $E(L_v)$ of the temporary tree can be expressed as

$$E(L_v) \leq C \cdot R \cdot \sqrt{m} = O\left(\sqrt{n^2 \int_{W_c}^{1} f(w) dw}\right).$$

We introduce another function $g(W_c)$, expressed by

$$g(W_c) \quad = n^2 \textstyle\int_{W_c}^{1} f(w) dw = \dfrac{\mathcal{W}^2 \int_{W_c}^{1} f(w) dw}{\left(\int_{W_c}^{1} w f(w) dw\right)^2}.$$

Thus, we have

$$g'(W_c) = \mathcal{W}^2 \cdot f(W_c) \int_{W_c}^{1} w f(w) dw \cdot \frac{2W_c \int_{W_c}^{1} f(w) dw - \int_{W_c}^{1} w f(w) dw}{\left(\int_{W_c}^{1} w f(w) dw\right)^4}.$$

Since W_c is a variable within the range of [0, 1], and the density function $f(w)$ is always positive, the monotonicity of the function $g(W_c)$ depends entirely on the positive and negative properties of the function $h(W_c)$ given by

$$h(W_c) = 2W_c \int_{W_c}^{1} f(w) dw - \int_{W_c}^{1} w f(w) dw. \tag{5}$$

Therefore, the ideal value of W_c that minimizes the expected value of the tree length must be within (0, 1) and $arg\{W_c : h(W_c) = 0\}$. Obviously, the expected tree length approaches infinity when $W_c = 1$. So the optimal values of W_c represented by W_{c_0} must satisfy $W_{c_0} = 0$ or $W_{c_0} = arg\{W_c : 2W_c \int_{W_c}^{1} f(w) dw - \int_{W_c}^{1} w f(w) dw = 0\}$, which is determined by the density function of the node weight in the network $f(w)$.

Specially, if the node weight is uniformly distributed, which means $f(w) = 1$. From Eqn. (5) we know $h(W_c) = -\frac{3}{2}(W_c - \frac{1}{3})(W_c - 1)$. So the optimal value $W_{c_0} = 0$ or $\frac{1}{3}$. However, $g(W_c = 0) = 2\mathcal{W}^2$ that is bigger than $g(W_c = \frac{1}{3}) = 1.898\mathcal{W}^2$. Thus the optimal value of W_c is $\frac{1}{3}$ when the node weight is uniformly distributed in the range of [0, 1]. ■

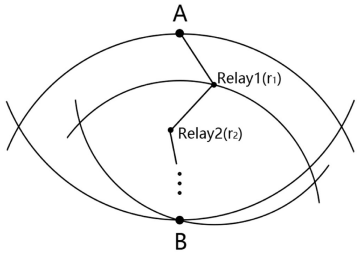


Fig. 5. The Construction of the Minimum Length Path between A and B.

4) Multicast Tree Length: In Lemmas 2 and 4, we analyze the expected tree length of the temporary tree when the nodes are uniformly distributed and generally distributed. However, we have to find some relays between every two connected receivers so as to reduce the transmission distance between nodes. In this way, every two directly connected receivers in the temporary tree are connected with a minimum length path and the construction of the multicast tree is finished. Now we are about to study the relationship between the minimum length path and the Euclidean distance between two nodes, and summarize our findings in Lemma 6.

Lemma 6: If n nodes are uniformly distributed in a circular area with a radius of R with the Euclidean distance between $\forall$ nodes A and B being $||AB||$, then the minimum length path $L(A, B)$ constructed by TSTB satisfies $L(A, B) \rightarrow ||AB||$ when n is sufficiently large.

Proof: According to the work in [43], the searching range of the receiver is set to be $o(\sqrt{\frac{\log n}{n}})$ so as to ensure that the network is connected.

Therefore, the average nodes in the searching range is $O(\log n)$. When n approaches infinity, the nodes within the searching range is also sufficiently large. Fig. 5 illustrates the minimum length path constructed based on TSTB.To prove that when $n \rightarrow \infty$, $L(A, B) \rightarrow ||AB||$, we assume there are t relays in the minimum length path between nodes A and B. From Fig. 5, we have $L(A, B) = ||Ar_1|| + \sum_{i=i}^{t-1} ||r_i r_{i+1}|| + ||r_t B||$. Thus the expected length of the minimum length path is

$$\begin{aligned} E(L(A,B)) &= E\Big(||Ar_1|| + \sum_{i=i}^{t-1} ||r_i r_{i+1}|| + ||r_t B||\Big) \\ &= E\Big(||Ar_1||\Big) + \sum_{i=1}^{t-1} E(||r_i r_{i+1}||) + E(||r_t B||) \\ &\overset{(a)}{\leq} \frac{C}{\sqrt{m}} \cdot ||AB|| + \sum_{i=i}^{t} \frac{C}{\sqrt{m}} \cdot ||r_i B|| \\ &\overset{(b)}{\leq} \frac{C}{\sqrt{m}} ||AB|| \left[\frac{1 - \left(1 - \frac{C}{\sqrt{m}}\right)^{t+1}}{1 - \left(1 - \frac{C}{\sqrt{m}}\right)}\right] = ||AB||, \end{aligned}$$

where Inequlity (a) is based on (4) in Lemma 2, we know that the expected length of the first step from node A to node B, is upper-bounded by

$$\frac{(1-e^{-\frac{1}{2}})R}{\sqrt{m}}\sum_{j=1}^{i} e^{-\frac{aj}{\pi}}\cdot\sqrt{j} \le \frac{C}{\sqrt{m}}\cdot||AB||.$$

Inequlity (b) above holds due to the fact that

$$||r_iB|| \le ||r_{i-1}B|| - E(||Ar_i||) = ||AB||\left(1-\frac{C}{\sqrt{m}}\right)^i.$$

In this way we can express $E(L(A,B))$ as a sum of a geometric series with proportional ratio $(1-\frac{C}{\sqrt{m}})$.

It is noteworthy that when $n\to\infty$, $t\to\infty$, the above results also establish. Thus, we can conclude that $L(A,B)\to||AB||$ when n is sufficiently large. ■

Remark: We can also calculate the expected length of the longest branch of our multicast tree. Obviously, the expected longest branch is the minimum length path between the source and the outermost leaf node. Thus, the expected length of the longest branch in a circle area with radius of R convergence to R when n is sufficiently large.

Theorem 1: Assume that all nodes are independently distributed in the network and the distribution function of node location is $f(r,\theta)$. The goal weight of the tree construction is $\mathcal{W}$. When $\mathcal{W}$ is large enough (equal to n is large enough), the expected tree length $E(L)$ is upperbounded by $O(\sqrt{\mathcal{W}})$.

Proof: In Lemma 4, we prove that in general distribution, the expected tree length of the temporary tree is upper bounded by $C\sqrt{m}\int_0^{2\pi}d\theta\int_0^R[r\cdot\sqrt{f(r,\theta)}]dr$, where $C=2.311$. And in Lemma 6, we prove that the expected length of the multicast tree is equal to that of temporary tree. Jointly considering the two facts, we have

$$E(L)\le 2.311\sqrt{m}\int_0^{2\pi}d\theta\int_0^R[r\cdot\sqrt{f(r,\theta)}]dr.$$

Therefore, in order to prove our theorem, we have to find the relationship between m and $\mathcal{W}$. As discussed in Lemma 5, we have $m=n\sum_{W_c}^1 f(w)dw$ and $n\sum_{W_c}^1 wf(w)dw\le 4\mathcal{W}$. Then we can prove

$$m\le\frac{4\mathcal{W}\cdot\sum_{W_c}^1 f(w)dw}{\sum_{W_c}^1 wf(w)dw}=4\mathcal{W}\cdot\psi(W_c)^2,$$

where

$$\psi(W_c)=\sqrt{\frac{\sum_{W_c}^1 f(w)dw}{\sum_{W_c}^1 wf(w)dw}}.$$

Thus we can further derive $E(L)$ as

$$\begin{aligned}E(L)&\le 4.622\cdot\psi(W_c)\cdot\sqrt{\mathcal{W}}\cdot\int_0^{2\pi}d\theta\int_0^R[r\cdot\sqrt{f(r,\theta)}]dr\\&=O(\sqrt{\mathcal{W}}).\end{aligned}$$

This completes our proof of the lemma. ■

B. Performance Analysis on Latency

Based on the tree length presented before, we now analyze the latency of the transaction confirmation process with our proposed TSTB algorithm. As previously mentioned, the latency of the transaction confirmation procedure corresponds to the running time of the multicast tree construction in our algorithm. In the following part, we discuss the running time of our algorithm, as given in Theorem 2.

Theorem 2: Assume n nodes are independently distributed in a circular area, the time complexity of constructing a multicast tree based on our algorithm is $O(\sqrt{n\log n})$.

Proof: There are basically four phases of our algorithm, we consider the time complexity of each phase separately. The total time complexity should be the sum of the running time of all four phases.

In the first phase, the source broadcasts its location into an area with radius R_i and increases it to R_{i+1} each time. We assume that there are n_i nodes within the area with radius R_i, and the time complexity of each searching process is upper bounded by $O(\log n_i)$. Thus the running time of phase 1 for i^{th} search is

$$E(t_1)_i\le O(\log n_i).$$

The dominant time cost of phase 2 is the process of finding neighboring receivers. The searching radius increases with the searching times. For j^{th} search, the radius is $r_{c_j}=2^{j-1}r_c$, and at most $O(2^i n{r_c}^2)$ relays are needed for messages forwarding. When the source searches for the i^{th} time, the worst case is the searching radius $r_c=R_i-R_{i-1}$. Thus, the running time of phase 2 for i^{th} search is upper bounded by

$$\begin{aligned}E(t_2)_i&\le O\left(\sum_{j=1}^{\lceil\log\frac{R_i-R_{i-1}}{r_c}\rceil}2^{j-1}n_i{r_c}^2\right)\\&\le O(n_ir_c)\\&=O(\sqrt{n_i\log n_i}).\end{aligned}$$

In phase 3, we repeat phase 1 and 2 for k times. Noticing that $n_i=\frac{n}{2^{2(k-i)}}$, thus the influence of this phase is calculated as follows:

$$\begin{aligned}E(t_1)&\le O\left(\sum_{i=1}^{k}\log n_i\right)\\&=O\left(\sum_{i=1}^{k}\log\left(\frac{n}{4^{k-i}}\right)\right)\\&=O\big(k\cdot\log n-2\ k^2\big)\\&\le O(k\cdot\log n),\end{aligned}$$

$$\begin{aligned}E(t_2) &\le O\left(\sum_{i=1}^{k}\sqrt{n_i \log n_i}\right)\\ &= O\left(\sum_{i=1}^{k}\sqrt{\frac{n}{4^{k-i}}\log\frac{n}{4^{k-i}}}\right)\\ &= O\left(\sum_{i=1}^{k}\sqrt{4^{-(k-i)}\cdot n\log n - 4^{-(k-i)}\cdot n\cdot 2(k-i)}\right)\\ &\le O\left(\sqrt{n\log n}\cdot\sum_{i=1}^{k}2^{-(k-i)}\right)\\ &= O(\sqrt{n\log n}).\end{aligned}$$

The total running time is the sum of all time cost in three phases. Therefore, we have

$$E(t) = E(t_1) + E(t_2) \le O(\sqrt{n\log n}),$$

which completes our proof. ■

As we can see from Theorem 2, the total running time is also in the same order of TST, which is one of the best among all the algorithms. That is to say, the repeat of searching processes brings about greater time complexity. Meanwhile, the latency of the corresponding transaction confirmation could also be obtained as the same magnitude of $O(\sqrt{n\log n})$.

C. Performance Analysis on Communication Cost

Another performance metric of interest is the communication cost of the transaction confirmation procedure. As mentioned earlier, the communication cost is measured by the amount of exchanged messages during the tree construction, as shown in Theorem 3.

Theorem 3: Assume n nodes are independently distributed in a circle area, the expected exchanged message amount of constructing a multicast tree based on TSTB is $O(n)$.

Proof: As discussed in Section 4, there are four types of messages exchanged in our TSTB algorithm. We use $M_i(1 \le i \le 4)$ to represent the quantity of each type of message sent during the construction.

Type 1: In the first phase, Message 1 (denoted by M_1) are sent by the source to broadcast its location and wake up receivers. Obviously, the quantity of M_1 for each searching process in an area with radius of R_i and nodes number of n_i. Since the searching radius is doubled each time, we have

$$E(M_1) = \sum_{i=1}^{k} n_i = \sum_{i=1}^{k}\frac{n}{2^{i-1}} = 2n(1-2^{-k}) = O(n).$$

Type 2: In the second phase, three types of messages are sent to construct the temporary tree. Message 2 is the *request message* sent by each receiver, and the quantity of this message is not influenced by the source searching process. Thus we can calculate the expected quantity of the request message in the whole area with radius of R_k.

For the l^{th} searching of a receiver, it broadcasts messages within an area with radius of $2^l r_c$. Thus there are $\frac{\pi(2^l r_c)^2}{\pi {R_k}^2} n$ nodes in the transmission range, which means for each receiver theret are $\frac{(2^l r_c)^2}{{R_k}^2} n$ request messages to be sent for the l^{th} receiver searching. Then we use the same analysis method in Lemma 2 to divide the whole area into m rings with the equal area. And assume a receiver is located in ring i, p_{j1} is the probability that a node in ring $j(j \le i)$ is chosen as a relay and p_{j2} is the probability that there is at least one node exists in ring j. Because when a receiver find another receiver in its searching area,t the connection is build. We have $l = \lceil \log \frac{\rho_j}{r_c} \rceil = \lceil \log \sqrt{\frac{j}{m}}\frac{R_k}{r_c} \rceil$. In this way, we can calculate the expected quantity of request message as

$$\begin{aligned}E(M_2) &= m\cdot\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{i} p_{j1}\cdot p_{j2}\cdot \sum_{l=0}^{\lceil \log\sqrt{\frac{j}{m}}\frac{R_k}{r_c}\rceil}\frac{(2^l r_c)^2}{{R_k}^2} n\\ &\le (1-e^{-\frac{1}{2}})\sum_{i=1}^{m}\sum_{j=1}^{i} e^{-\frac{aj}{\pi}}\cdot 4\lceil \log\sqrt{\frac{j}{m}}\frac{R_k}{r_c}\rceil \frac{n}{m} j\\ &= O\left(\sum_{i=1}^{m}\sum_{j=1}^{i} e^{-\frac{aj}{\pi}}\cdot\frac{n}{m} j\right)\\ &= O\left(\frac{n}{m}\sum_{i=1}^{m}\sum_{j=1}^{i}\left(e^{-\frac{aj}{2\pi}}\cdot\sqrt{j}\right)^2\right)\\ &\overset{(a)}{=} O(n).\end{aligned}$$

From Eqn. (3), we can easily prove that $\sum_{j=1}^{i} e^{-\frac{aj}{\pi}}\cdot\frac{n}{m} j$ is upper bounded by a constant. Therefore, we can easily prove that Eqn. (a) holds.

Type 3: The third type of message in phase 2 is the *respond message*. The respond messages are sent from receivers to their previous hops. Thus the expected quantity of the respond messages are upper bounded by the total hops in the multicast tree. Obviously, the expected distance between two nodes in the network is upper bounded by $\sqrt{\frac{1}{n}}$, i.e.,

$$E(M_3) \le O\left(\frac{E(L_v)}{\sqrt{\frac{1}{n}}}\right) = O\left(\frac{\sqrt{m}}{\sqrt{\frac{1}{n}}}\right) \le O(n).$$

Type 4: The last type of message is the *confirm message*. There is no doubt that the quantity of this message cannot exceed the quantity of respond message. Thus, we have

$$E(M_4) \le E(M_3) \le O(n).$$

Combining the above four cases, we can get the total message complexity of TSTB as

$$E(M) = \sum_{i=1}^{4} E(M_i) = O(n),$$

which completes our proof. ■

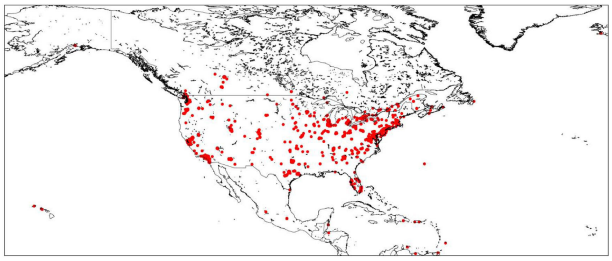


Fig. 6. Overview of the geographical distribution of bitcoins (represented by red dots) in North America.

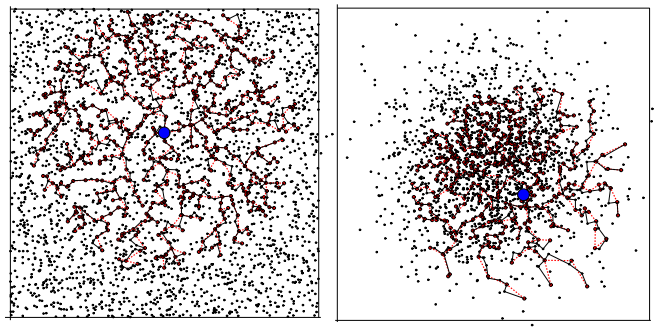


Fig. 7. Simulation of the blockchain multicast tree construction in (a) uniformly distributed network and (b) Gaussian distributed network.

VI. Experiments

To empirically validate the effectiveness of our proposed TSTB algorithm, we conduct experimental measurements based on both simulations and real applications to the Bitcoin network in terms of three metrics, i.e., tree length, latency and communication cost.

A. Experimental Setup

We introduce our experimental setup from the following four aspects in this subsection:

1) Simulation Setup: In synthetic datasets, we simulate a network where both nodes' location distribution and weight distribution are randomly generated. For each type of the distributions, we simulate using both uniform and Gaussian distributions to create different scenarios to perform our experiments. To elaborate, under a specified distribution function of the node location, we have $f(r,\theta)_{uni}=\frac{1}{\pi {R_k}^2}$ and $f(r,\theta)_{nor}=\frac{1}{\sqrt{2\pi}}\exp^{-\frac{r^2}{2\sigma^2}}$ $(0\leq r\leq R_k)$; as for the distribution function of the node weight, we have $f(w)_{uni}=1$ and $f(w)_{nor}=\frac{2}{\sqrt{2\pi}}\exp^{-\frac{(w-1)^2}{2\sigma^2}}$ $(0\leq w\leq 1)$, where the subscripts "uni" and "nor" represent the uniform distribution and normal (or Gaussian) distribution, respectively. Thus, there are totally four different situations, i.e., uniformly distributed nodes with uniformly distributed weights (abbreviated by Weight Uni & Location Uni), normal distributed nodes with uniformly distributed weights (abbreviated by Weight Uni & Location Nor), uniformly distributed nodes with normal distributed weights (abbreviated by Weight Nor & Location Uni), and normal distributed nodes with normal distributed weights (abbreviated by Weight Nor & Location Nor). In each scenario, we carry out comprehensive experiments with various number of receivers ranging from 0 to 5000 and varying critical weights W_c to explore their effects on the metrics of interests.

To eliminate the randomness as mush as possible, all the above simulation results are obtained by averaging the results of 100 times' running.

2) Real Dataset Preparation: The dataset, which we call **bitnodes**, are crawled from the website bitnodes.earn.com, which is currently being developed to estimate the size of the Bitcoin network by finding all the reachable bitcoin users worldwide. The website records 9706 bitcoin users (miners), of which 8146 are valid because the IP addresses of the rest nodes are not visible. In our experiment, we only target the bitcoin users located in North America area, where the most bitcoin users worldwide are concentrated. Totally, there are 2512 users located around this area, with an overview of their location distribution pattern visualized in Fig. 6.

With the help of an api website (*IP-API.com*), we can get the latitude and longitude coordinates of nodes from their IP address, which can be obtained from the dataset. In this way, we can reorganize the real locations of these nodes in the form of coordinates.

Then we use the amount of transactions processed by each user to represent his/her weight. This is reasonable because the higher amount the user has, the larger probability that he/she is more reliable or trustful in block verification.

3) Algorithms Compared: In our experiments on both synthetic and real datasets, we compare the performance of TSTB with the optimal Steiner tree algorithm and TST. The optimal Steiner tree is algorithmically computed via [49]. We do not compare TSTB with other approximate Steiner tree solutions like SPH, KSPH, ADH and DA in these evaluations, for they were shown to perform worse than TST [16].

4) Testing Environment: We experiment on 3 Linux machines, each with a 32 core 2.6 GHz CPU and 128 GB RAM. All the codes are run in C++. To simulate the distributed tree construction behavior of TSTB and TST, we use Docker to create a testbed of multiple containers each having 64 MB of RAM. Each container corresponds to a node, and every $n/96$ nodes share a Linux machine.

B. Evaluation on Tree Length and Communication Cost Under Synthetic Datasets

Now we report our empirical evaluation on the tree length and the communication cost under the influences of both the distribution of node location and the distribution of node weight. For a general reproduction of our simulation, Fig. 7 visualizes the simulated processes of the tree construction of TSTB in two different location distributions. Then performance comparisons are plotted in Figs. 8, 9 and 10, which we will explain in details in the sequel.

1) The effects of m, distributions of node locations and node weights on the tree length: We fix the network size with

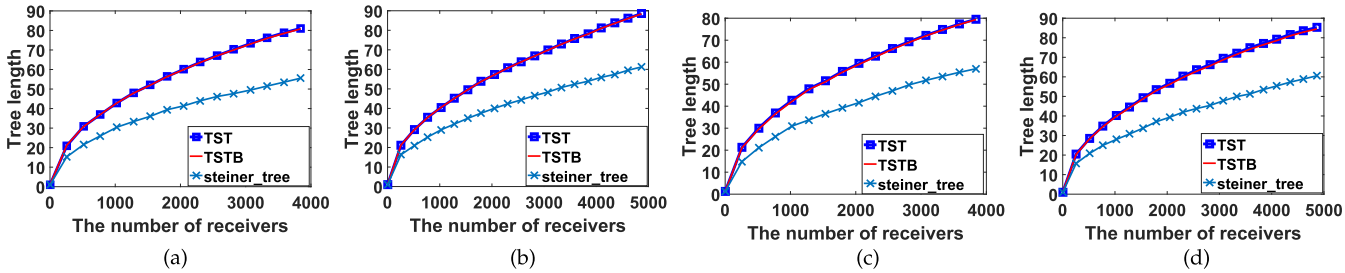


Fig. 8. The relationship between tree length and the number of receivers under various distributions of node weights and node locations. (a) Weight Uni&Location Uni. (b) Weight Uni&Location Nor. (c) Weight Nor&Location Uni. (d) Weight Nor&Location Nor.

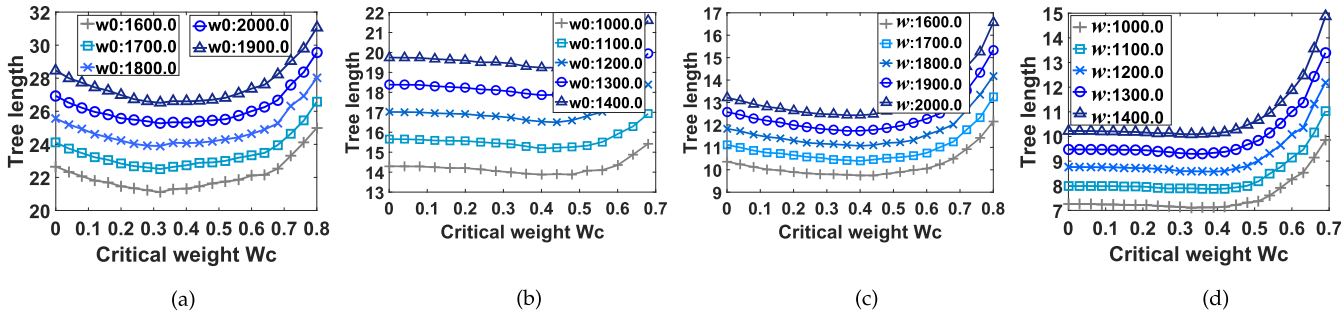


Fig. 9. The relationship between tree length and critical weights with different goal weights under various distributions of node weights and locations. (a) Weight Uni&Location Uni. (b) Weight Uni&Location Nor. (c) Weight Nor&Location Uni. (d) Weight Nor&Location Nor.

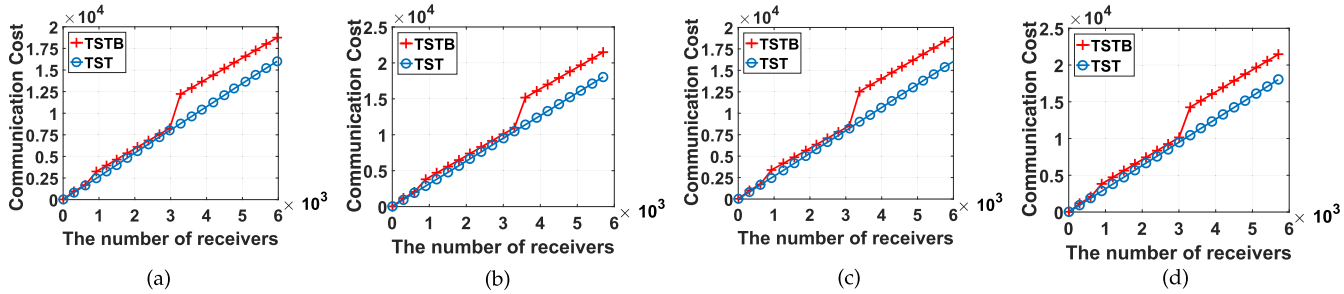


Fig. 10. The relationship between communication cost and the number of receivers under various distributions of node weights and locations. (a) Weight Uni&Location Uni. (b) Weight Uni&Location Nor. (c) Weight Nor&Location Uni. (d) Weight Nor&Location Nor.

$n = 10000$ and vary the numbers of receivers m from 0 to 5000. As can be seen in Fig. 8, the tree length is proportional to $\sqrt{m}$, which agrees with the theoretical claim in Lemma 2. Meanwhile, the coefficient obtained empirically is smaller than that in our theoretical derivation (2.311 in theory). This suggests that the empirical performance of tree length is in fact better than expected.

Also, we observe that the curves of our TSTB algorithm and TST algorithm are almost coincident. This phenomenon confirms that the repeated searching processes do not increase the tree length, which is in line with the conclusion of Lemma 3. On the other hand, although the tree length of the optimal Steiner tree is smaller than ours, they are in the same order, only with a ratio of 1.5 in the difference of tree lengths on average.

Furthermore, comparing Figs. 8(a)&(c) or (b)&(d), we find that in the same location distribution, there is no significant effect of different weight distributions on the curves. This is because the distribution of weights mainly affects the selection of W_c. However, if we fix the number of receivers, then it is equivalent to having excluded this effect. Also notice that in Figs. 8(a)&(b) or (c)&(d), when the number of receivers is small, the tree length under the Gaussian distribution is slightly lower than that under the uniform distribution. The reason is that in Gaussian distribution, the majority of the nodes are concentrated toward the source (miner), which means for the same m, the tree length required to build from the source is smaller in this situation. However, the effect is weaken as m becomes larger, since the density of nodes that are located far away from the source is bigger in uniform distribution.

2) The relationship between the tree length and the critical weight W_c: We change the value of W_c from 0 to 0.8 and change the goal weight $\mathcal{W}$ at the same time to examine how

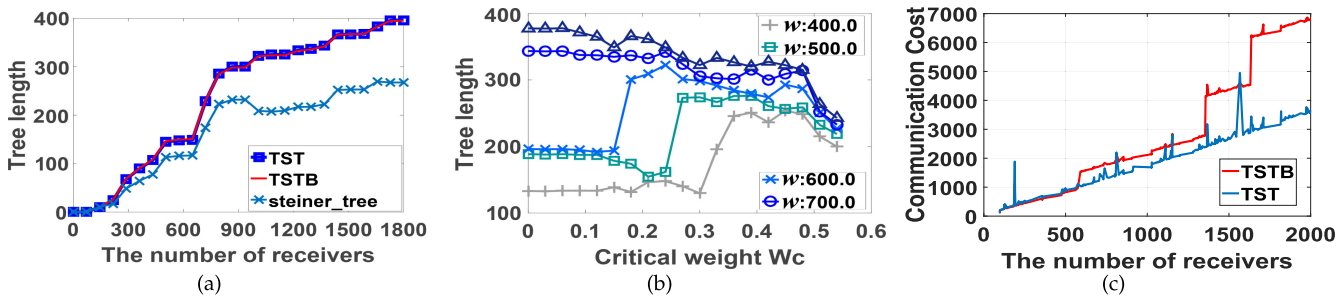


Fig. 11. Experiment results based on real data with (a) the relationship between tree length and the number of receivers, (b) the relationship between tree length and critical weight, (c) the relationship between communication cost and the number of receivers.

the tree length will change accordingly, with the results graphically shown in Fig. 9. In Lemma 5, we prove that there must exist an optimal critical weight, under which the tree length is minimized. This optimal critical weight is jointly dependent on the distributions of node location and node weight. As shown in Figs. 9(a)&(b), when the node weight follows uniform distribution, the optimal W_c is approximately 0.33, which matches the theoretical finding in Lemma 5. And as W_c tends to 1, the tree length increases without exception. This is because when W_c becomes bigger, the number of nodes selected as receivers per unit area is reduced. As a result, source searching process is repeated more times which definitely leads to a larger tree length.

When it comes to the influence of the goal weight $\mathcal{W}$, obviously the bigger $\mathcal{W}$ is, the bigger the tree length becomes. Similarly, by comparing Figs. 9(a)&(b) or (c)&(d), we can draw the conclusion that when $\mathcal{W}$ is small, the tree length in Gaussian distribution is smaller than that in uniform distribution.

3) The communication cost under different distributions: In this part, we change the network size n from 0 to 5000 and let W_c remain to be 0.33. We compare the communication cost, i. e, the amount of exchanged messages during tree construction under TSTB and TST, as reported in Fig. 10. We can see that regardless of the difference of distributions, the amount of exchanged messages is always proportional to the network size n, which confirms the result in Theorem 3. However, we notice that compared with TST, our algorithm exhibits two main features: (i) The curves experience sharp increases with the network size n periodically. This results from the repeated source search processes. At some specific values of n, TSTB just goes through a new source search process. This is because of the repeated source search processes that lead to more inevitable exchanged messages in our algorithm; (ii) In each source search process, the slope of curve under TSTB is smaller than that of TST. This is because TSTB has no circle elimination process which is however needed in TST. As a result, it saves both the running time and the communication cost.

C. Evaluation on Tree Length and Communication Cost Under Real Bitcoin Datasets

Upon simulations on synthetic datasets, we continue to perform experiments on real Bitcoin dataset. In analogy with the simulation part, we evaluate the tree length as well as communication cost of different tree construction methods, as shown in Fig. 11. Due to the geographical inhomogeneity (most of the bitcoin nodes are concentrated in the east of USA with relatively sparse nodes in the western part), the corresponding results also exhibit significant irregularity in the curve shapes. It is worth noting that the values shown in the y-axis in Figs. 11(a) and (b) are obtained by using 1km as the unit to measure the tree length.

From Fig. 11(a), we can see that the tree length is no longer strictly proportional to $\sqrt{m}$. This can be interpreted by Lemma 4 that the expected tree length in general distribution is closely related to the choice of location distribution function. Since real bitcoin network inherently has very complicated node location distribution, it largely affects the shape of the curves. However, the results still confirm that the tree length of TSTB is almost the same as that of TST, sometimes even better, and both of them are a little larger than that of the optimal Steiner tree. Another observation is that the curves exhibit sudden increases at some points such as $m = 190$ and $m = 300$. This is because the algorithm completes the exploration of all nodes in a nodes concentrated area when $m = 190$ and 300 (approximately), and if m still increases, the searching scope needs to expand but with less nodes being found, which leads to a sudden increase in tree length. Furthermore, the tree length of Steiner tree sometimes experiences decreases, which is caused by the characteristics of the construction of Steiner tree algorithm. Thus, when the number of nodes increases, the Steiner tree may change the structure of its internal construction and lead to the decrease in tree length.

Fig. 11(b) plots the relationship of the tree length and the critical weight. Because the distribution function of node weight in real bitcoin network is also uneven, the value of the optimal critical weight is changeable.

The message complexity shown in Fig. 11(c) generally confirms to the results of the simulation. We can still identify the sharp increases of TSTB caused by the repeated source search processes. However, influenced by the variation of node density in local areas, the curves experience some fluctuations. Furthermore, the curve of TST fluctuates stronger than our TSTB. This is one of the benefits of repeated searching, which makes the construction more stable and will not cause a big increase in communication cost.

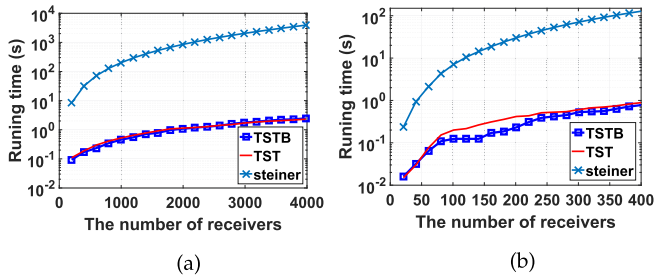


Fig. 12. Running time of different tree construction algorithms with varying numbers of receivers. (a) Synthetic datasets. (b) Real bitcoin datasets.

D. Evaluation on Latency Under Synthetic and Real Bitcoin Datasets

Last but not least, we evaluate the latency, another important metric of the transaction confirmation, under different approaches of tree construction in both synthetic and real datasets. We report the comparison results under TSTB, TST and Steiner tree in Fig. 12, with time unit set as one second. As previously mentioned, the latency is measured by the running time of tree construction. Particularly, for the running time results in synthetic dataset (Fig. 12(a)), the results are obtained based on the averaging of different node location and weight distributions. Due to the huge time differences between distributed algorithms (TSTB and TST) and centralized algorithms (Steiner tree), we use a base-10 log scale for the y-axis that reflects the running time for the ease of comparison.

Seen from Fig. 12, the latency of TSTB algorithm is in the same order of that of TST, both of which are significantly smaller than the running time of the optimal Steiner tree algorithm. Particularly, in the experimental case where we set the largest number of receivers, the exact running time of TSTB, TST and Steiner tree turns out to be 2.467 seconds, 2.319 seconds and 3947.27 seconds in synthetic datasets (with 4000 receivers) and 0.796 seconds, 0.905 seconds and 129.69 seconds in real ones (with 400 receivers). The significant difference, again, agrees with the NP-hardness of constructing the optimal Steiner tree as well as the efficiency of distributed algorithms like TSTB and TST. And both the simulation and the real data experiments confirm our theoretical claim that the repeated source search processes in TSTB incurs no extra time consuming. The comparison of running time consolidates our expectation that TSTB is indeed of help in reducing the block propagation latency.

VII. Conclusion and Future Work

In this paper, we design and analyze an algorithm to build multicast trees in wireless blockchain, in hope of optimizing the block propagation process while verifying the authenticity of the transaction. We analyze the efficiency of our algorithm with three performance metrics, i.e., the tree length, latency and communication cost. We theoretically demonstrate that the tree length is upper bounded by $O(\sqrt{n})$, which is in the same order as the optimal Steiner tree. We prove that the latency of transaction confirmation (dominated by the running time of the tree construction) is $O(\sqrt{n\log n})$, also the best among all solutions by now. The communication cost turns out to be $O(n)$ during the tree construction. Finally, the efficiency of our proposed algorithm is experimentally validated with both simulations results and real applications in the Bitcoin network.

However, there are also some constraints in the proposed algorithm. For example, we assume that all the nodes use the wireless access in the blockchain network, which could not exactly characterize some real cases, where most nodes may be stationary in the network. Therefore, it is a desirable work to model the blockchain network with partial wireless nodes, and investigate a new routing mechanism under this modeling. Furthermore, in such cases, since the frequency of a user publishing transactions and blocks is relatively low, the user is easy to be forgotten by the network, which may lead to the topology of the network rapidly changing over time. And thus the constructed tree may be time-limited. Under this circumstance, another interesting direction is to design an adaptive algorithm to realize the tree construction, which could adapt itself to the time-varying topology of the wireless blockchain network via dynamic programming or adaptive greedy algorithms. Last but not least, it is also worthwhile to apply our works in some real wireless scenarios like iot networks (e.g., smart home system) where smart-home devices can equip blockchain clients, cellular and WiFi/WLAN networks where personal computers and other mobile devices could be regarded as blockchain clients, and so on.

References

[1] M. S. Ferdous, M. J. M. Chowdhury, M. A. Hoque, and A. Colman, "Blockchain consensus algorithms: A survey," 2020, *arXiv:2001.07091*. [Online]. Available: https://arxiv.org/abs/2001.07091

[2] Y. Xu, C. Zhang, Q. Zeng, G. Wang, J. Ren, and Y. Zhang, "Blockchain-enabled accountability mechanism against information leakage in vertical industry services," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1202–1213, Apr.–Jun. 2021.

[3] H.-N. Dai, Z. Zheng, and Z. Y. Zheng, "Blockchain for Internet of Things: A survey," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8076–8094, Oct. 2019.

[4] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Bus. Rev.*, 2008, Art. no. 21260.

[5] S. G. Motlagh, J. Mišić, and V. B. Mišić, "The impact of selfish mining on bitcoin network performance," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 724–735, Jan.–Mar. 2021.

[6] L. Da Xu, Y. Lu, and L. Li, "Embedding blockchain technology into IoT for security: A survey," *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10452–10473, Jul. 2021.

[7] C. Nartey *et al.*, "On blockchain and IoT integration platforms: Current implementation challenges and future perspectives," *Wireless Commun. Mobile Comput.*, vol. 2021, 2021, Art. no. 667248225, doi: 10.1155/2021/6672482.

[8] "SmartMesh HyperMesh Ecosystem Whitepaper (Version 3.0)". [Online]. Available: http://smartmesh.io/SmartMeshWhitePaperEN.pdf.

[9] "BlockMesh-White_Paper-1". [Online]. Available: https://blockmesh.io/pdf/BlockMesh-White_Paper-1.pdf.

[10] D. Frey, M. X. Makkes, P.-L. Roman, F. Taïani, and S. Voulgaris, "Dietcoin: Shortcutting the bitcoin verification process for your smartphone," 2018, *arXiv:1803.10494*. [Online]. Available: https://arxiv.org/abs/1803.10494.

[11] D. Chatzopoulos, S. Gujar, B. Faltings, and P. Hui, "Localcoin: An ad-hoc payment scheme for areas with high connectivity: Poster," in *Proc. 17th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2016, pp. 365–366.

[12] S. Buchko, "How Long do Bitcoin Transactions Take?," 2017. [Online]. Available: https://coincentral.com/how-long-do-bitcoin-transfers-take

[13] R. M. Karp, "Reducibility among combinatorial problems," *Complexity of Computer Computations*. Boston, MA, USA: Springer, 1972, pp. 85–103.

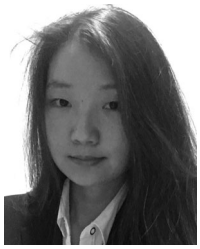
[14] F. Bauer and A. Varma, "Degree-constrained multicasting in point-to-point networks," in *Proc. IEEE INFOCOM*, 1995, pp. 369–376.
[15] P. Saikia and S. Karmakar, "Distributed approximation algorithms for steiner tree in the congested clique," *Int. J. Found. Comput. Sci.*, vol. 31, no. 7, pp. 941–968, 2020.
[16] H. Gong, L. Fu, X. Fu, L. Zhao, K. Wang, and X. Wang, "Distributed multicast tree construction in wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 63, no. 1, pp. 280–296, Jan. 2017.
[17] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 289–300, Mar.–Apr. 2020.
[18] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *Proc. IEEE BigData Congr.*, 2017, pp. 557–564.
[19] L. Zhang, H. Xu, O. Onireti, M. A. Imran, and B. Cao, "How much communication resource is needed to run a wireless blockchain network?," 2021, *arXiv:2101.10852*. [Online]. Available: https://arxiv.org/abs/2101.10852
[20] B. Cao *et al.*, "When Internet of Things meets blockchain: Challenges in distributed consensus," *IEEE Netw.*, vol. 33, no. 6, pp. 133–139, Nov./Dec. 2019.
[21] G. D. Monte *et al.*, "Scaling blockchains without giving up decentralization and security: A solution to the blockchain scalability trilemma," in *Proc. ACM 3rd Workshop Cryptocurrencies Blockchains Distrib. Syst.*, 2020, pp. 71–76.
[22] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: A survey," *Int. J. Web Grid Serv.*, vol. 14, no. 4, pp. 352–375, 2018.
[23] D. Yang, C. Long, H. Xu, and S. Peng, "A review on scalability of blockchain," in *Proc. 2nd Int. Conf. Blockchain Technol.*, 2020, pp. 1–6.
[24] H. Kohad, S. Kumar, and A. Ambhaikar, "Scalability issues of blockchain technology," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 3, pp. 2385–2391, 2020.
[25] K. Croman *et al.*, "On scaling decentralized blockchains," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2016, pp. 106–125.
[26] Q. Zhou, H. Huang, Z. Zheng, and J. Bian, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16 440–16 455, 2020.
[27] C. Egger *et al.*, "Atomic multi-channel updates with constant collateral in bitcoin-compatible payment-channel networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 801–815.
[28] L. Luu, V. Narayanan, C. Zheng, K. Baweja, S. Gilbert, and P. Saxena, "A secure sharding protocol for open blockchains," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 17–30.
[29] O. Ersoy *et al.*, "Information propagation on permissionless blockchains," vol. 1712, 2017, arXiv:1712.07564. [Online]. Available: https://allquantor.at/blockchainbib/pdf/ersoy2017information.pdf
[30] M. Castro *et al.*, "Practical byzantine fault tolerance," in *Proc. OSDI*, vol. 99, no. 1999, 1999, pp. 173–186.
[31] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX ATC*, 2014, pp. 305–319.
[32] S. Nakamoto, "White paper: Bitcoin: A peer-to-peer electronic cash system," *Email Posted Listserv*, vol. 9, pp. 1–9, 2008.
[33] A. Dorri *et al.*, "Blockchain in Internet of Things: Challenges and solutions," 2016, *arXiv:1608.05187*. [Online]. Available: https://arxiv.org/abs/1608.05187
[34] X. Dai, J. Xiao, W. Yang, C. Wang, and H. Jin, "Jidar: A jigsaw-like data reduction approach without trust assumptions for bitcoin system," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 1317–1326.
[35] J. Wang *et al.*, "Monoxide: Scale out blockchains with asynchronous consensus zones," in *Proc. 16th USENIX Symp. Netw. Syst. Des. Implementation*, 2019, pp. 95–112.
[36] A. Miller, I. Bentov, S. Bakshi, R. Kumaresan, and P. McCorry, "Sprites and state channels: Payment networks that go faster than lightning," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2019, pp. 508–526.
[37] H. Kalodner, S. Goldfeder, X. Chen, S. M. Weinberg, and E. W. Felten, "Arbitrum: Scalable, private smart contracts," in *Proc. USENIX Secur. Symp.*, 2018, pp. 1353–1370.
[38] E. Rohrer *et al.*, "Kadcast: A structured approach to broadcast in blockchain networks," in *Proc. 1st ACM Conf. Adv. Financial Technol.*, 2019, pp. 199–213.
[39] Y. Shahsavari, K. Zhang, and C. Talhi, "A theoretical model for block propagation analysis in bitcoin network," *IEEE Trans. Eng. Manag.*, to be published, doi: 10.1109/TEM.2020.2989170.
[40] C. Ras, K. Swanepoel, and D. A. Thomas, "Approximate Euclidean steiner trees," *J. Optim. Theory Appl.*, vol. 172, no. 3, pp. 845–873, 2017.
[41] F. Bauer and A. Varma, "Distributed algorithms for multicast path setup in data networks," *IEEE/ACM Trans. Netw.*, vol. 4, no. 2, pp. 181–191, Apr. 1996.
[42] P. Saikia *et al.*, "Primal-dual based distributed approximation algorithm for prize-collecting steiner tree," *Discrete Math., Algorithms Appl.*, vol. 13, no. 2, 2021, Art no. 2150008.
[43] P. Gupta and P. R. Kumar, "Critical power for asymptotic connectivity in wireless networks," *Stochastic Analysis, Control, Optimization and Applications*. Berlin, Germany, 1999, pp. 547–566.
[44] X. Tong, H. Li, X. Tian, and X. Wang, "Wi-fi localization enabling self-calibration," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 904–917, Apr. 2021.
[45] E. E. Smith, *Concepts and Induction*. MA, USA, 1989.
[46] Z. Wang, H. R. Sadjadpour, and J. Garcia-Luna-Aceves, "A unifying perspective on the capacity of wireless ad hoc networks," in *Proc. IEEE INFOCOM*, 2008, pp. 211–215.
[47] D.-Z. Du *et al.*, "A proof of the gilbert-pollak conjecture on the steiner ratio," *Algorithmica*, vol. 7, no. 1, pp. 121–135, 1992.
[48] J. Stewart, D. K. Clegg, and S. Watson, *Calculus: Early Transcendentals*. MA, USA, 2020.
[49] X. Wang, "Exact algorithms for the steiner tree problem," Ph.D. dissertation, Univ. Twente, Enschede, The Netherlands, 2008.

Teng Long received the Ph.D. degree in computer software and theory from the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China. She is currently a Lecturer with the School of Information Engineering, China University of Geosciences, Beijing, China. Her research interests include formal methods, distributed system verification, and program analysis.

Shan Qu received the B.E. degree from the School of Mathematics and Statistics, Xidian University, Xi'an, China, in 2017. She is currently working toward the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. Her research interests include social networking, big data, and machine learning.

Qi Li received the B.E. degree in information and communication engineering from Xidian University, Xi'an, China, in 2019. He is currently working toward the Ph.D. degree with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include big data and machine learning.

Huquan Kang received the B.E. degree in 2021 in software engineering from Shanghai Jiao Tong University, Shanghai, China, where he is currently working toward the Ph.D. degree of Zhiyuan Honors Program in computer science and technology. His research interests include knowledge structure and measurement. He was awarded as the Outstanding Winner in MCM/ICM 2020 and Outstanding Graduate of Shanghai Jiao Tong University in 2021.



Luoyi Fu received the B.E. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2009 and the Ph.D. degree in computer science and engineering in the same university in 2015. She is currently an Assistant Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University. Her research interests include social networking and big data, scaling laws analysis in wireless networks, connectivity analysis, and random graphs. She is a Member of the Technical Program Committees of several conferences, including ACM MobiHoc 2018–2021 and IEEE INFOCOM 2018-2021.

Xinbing Wang (Senior Member, IEEE) received the B.S. degree (Hons.) from the Department of Automation, Shanghai Jiao Tong University, Shanghai, China, in 1998, the M.S. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2001, and the Ph.D. degree major from the Department of electrical and Computer Engineering, minor from the Department of Mathematics, North Carolina State University, Raleigh, NC, USA, in 2006. He is currently a Professor with the Department of Electronic Engineering, Shanghai Jiao Tong University. Dr. Wang is an Associate Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING and IEEE TRANSACTIONS ON MOBILE COMPUTING, and a Member of the Technical Program Committees of several conferences, including ACM MobiCom 2012, 2018–2019, ACM MobiHoc 2012–2014, and IEEE INFOCOM 2009–2017.

Chenghu Zhou received the B.S. degree in geography from Nanjing University, Nanjing, China, in 1984, and the M.S. and Ph.D. degrees in geographic information system from the Chinese Academy of Sciences (CAS), Beijing, China, in 1987 and 1992, respectively. He is currently an Academician with CAS, where he is also a Research Professor with the Institute of Geographical Sciences and Natural Resources Research, and a Professor with the School of Geography and Ocean Science, Nanjing University. His research interests include spatial and temporal data mining, geographic modeling, hydrology and water resources, and geographic information systems and remote sensing applications.