A Theoretical Model for Block Propagation Analysis in Bitcoin Network

Yahya Shahsavari, Kaiwen Zhang, *Member, IEEE*, and Chamseddine Talhi

***Abstract*—Blockchains are currently gaining attention as a newly emerging technology in both academia and industry, capable of impacting a variety of domains beyond cryptocurrencies. Performance modeling can be used to provide us with a deeper understanding of the behavior and dynamics within blockchain peer-to-peer networks. Blockchain system architects can leverage network models to properly tune their system and to reduce design costs significantly. In this article, we focus on the original and well-established Bitcoin blockchain network. In particular, we propose a random graph model for performance modeling and analysis of the inventory-based protocol for block dissemination. This model addresses the impact of key blockchain parameters on the overall performance of Bitcoin. We derive some explicit and closed-form equations for block propagation delay and traffic overhead in the Bitcoin network. We also adapt our model to study the impact of deploying a relay network and investigate the effect of the relay network size on the network performance and decentralization. We implement our model using the popular network simulator OMNet++. We validate the accuracy of our theoretical model and its implementation with our dataset mined from the Bitcoin network. Our results show the tradeoff between the default number of connections per node, network bandwidth, and block size in order to compute the optimal block propagation delay over the network. Additionally, we found that bigger relay networks can jeopardize the decentralization of the Bitcoin network.**

***Index Terms*—Bitcoin, blockchain, cryptocurrency, distributed ledger technologies (DLTs), peer-to-peer network, performance modeling, relay networks.**

I. Introduction

BLOCKCHAIN is gaining attention as a newly emerging technology in both academia and industry: from cryptocurrencies [1] to decentralized applications (e.g., cloud storage [2]) and pervasive use cases (e.g., IoT [3]–[5], health care [6], [7], and legal and law enforcement [8]–[10]). Despite its potential as a disruptive technology, blockchain systems currently face major performance issues, which hinder their widespread adoption as an alternative to other data management solutions [11].

Furthermore, we observe that there is a wide variety of distributed ledger technology (DLT) platforms with vastly different architectures (permissioned or permissionless), consensus mechanisms [Proof-of-Work (PoW), Proof-of-Stake, practical Byzantine fault tolerance (PBFT), etc.], data structures (Blockchain or directed acyclic graph (DAG)), etc. [12], [13]. Each system also has a variety of configurable parameters, such as block size, block time, or number of peers.

In light of these observations, we argue that it is essential for blockchain developers to choose the right DLT system and appropriately tune it to maximize its performance given a specific application and infrastructure. To assist blockchain architects in this design process, we propose the use of analytical models in order to predict and compare the performance of various blockchain designs.

A theoretical model will help the blockchain designers in obtaining a better understanding of the underlying blockchain dynamics and characteristics, which impact the performance of the blockchain network. A theoretical model can also accelerate the development of a blockchain system by quickly discovering a theoretically optimal initial design, which is preferable to an incremental design through iterative benchmarking.

Bitcoin suffers from a lack of scalability, which may endanger the longevity of the cryptocurrency. In particular, the performance of Bitcoin is greatly impacted by P2P network parameters, such as throughput and information propagation delay.

In recent years, there have been many proposals to address the performance inefficiency of blockchain networks. From selecting nodes located in the geographical proximity of the miners as the next logical hop [14], [15] to implementing global relay networks [16]–[18] as well as efforts for optimizing the block propagation mechanism, such as Bitcoin Improvement Proposal 152 (BIP 152) [19] and Graphene [20], these solutions attempt to reduce information propagation delay in the Bitcoin network. Therefore, a performance model would allow us to calculate the anticipated benefit of such solutions.

In this article, we revise and considerably extend our previous work presented in [21], in which we focused on exploring the design space surrounding the original and most well-known blockchain system, i.e., Bitcoin [22]. We presented a model for estimating compact block propagation delay and message exchanging traffic overhead for the Bitcoin data exchange protocol using inventory vectors. The contributions of this article are as follows.

1) We model the Bitcoin overlay network using an Erdös–Rény model [23] to generate connected random graphs. We take into account both legacy block propagation protocol and BIP 152.

Manuscript received July 1, 2019; revised November 27, 2019 and March 14, 2020; accepted April 13, 2020. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and Bitfarms Co. Review of this manuscript was arranged by Department Editor K.-K. R. Choo. *(Corresponding author: Yahya Shahsavari.)*

The authors are with the Department of Software and IT engineering, École de Technologie Supérieure, University of Quebec, Montreal, QC H3C 1K3, Canada (e-mail: yahya.shahsavari.1@ens.etsmtl.ca; kaiwen.zhang@etsmtl.ca; chamseddine.talhi@etsmtl.ca).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEM.2020.2989170

2) We derive explicit mathematical equations to estimate important performance metrics, namely block propagation delay and traffic overhead. Our extended model considers the long tail in the block outreach model.
3) We identify two important Bitcoin configuration parameters, which impact performance: average number of connections per node and the block size.
4) We implement our theoretical model using the network simulator OMNet++, as a discrete event simulator.
5) We validate our model and our simulation using our dataset mined from the Bitcoin network. Our results show the sensitivity of block propagation delay with various Bitcoin parameters.
6) We estimate the probability density function (PDF) of the rate at which the participating nodes receive the propagating block.
7) We investigate the impact of deploying relay networks on the aforementioned performance metrics.

The rest of this article is organized as follows. Section II reviews the related works. Section III presents a summarized overview of Bitcoin fundamentals necessary to understand this article. In Section IV, we present our analytical model using a random graph network and capture the Bitcoin network behavior and dynamics. In Section V, we compare the results of our analytical model with simulation results as well as empirical amounts mined from the Bitcoin network in order to validate the accuracy of the model. In Section V-C, we leverage our theoretical model for performance analysis of the Bitcoin network. In Section VI, we discuss the managerial implications of this article. Finally, Section VII concludes this article.

II. Related Works

A. Performance Analysis of Bitcoin

To the best of our knowledge, there exist few works on performance modeling and analysis of blockchain networks, especially from a theoretical perspective. The majority of existing works rely on analysis based on simulation or experimental data gathered from the blockchain networks.

In [24], a theoretical model for the propagation delay based on the path length is presented. However, this model is not validated by realistic data. Moreover, this model does not describe the impact of different configuration parameters, such as block size, average number of connections per node, network bandwidth, and network size (i.e., total number of nodes) on the performance of the network.

In [25], an analytical model for the Bitcoin network based on the Jacksonian queuing network model is presented. In this work, the data (i.e., transactions and blocks) forwarding is modeled by branching processes in the network with a random distribution of node connectivity and arrival of blocks and transactions is modeled as a nonhomogeneous Poisson process. However, this work suffers in some aspects. For example, it does not validate or verify the proposed model. Moreover, the evaluation is done for a network size of 2500–5000 nodes, which is not a realistic assumption regarding the current size of the Bitcoin network. Like the previous work, this research did not address the effects of some important configuration parameters on the overall performance of the network.

Nagayama *et al.* [26] analyzed the block propagation delay in the Bitcoin network using a network simulator. The study conducts experiments comparing the compact block propagation protocol against the legacy block propagation protocol. According to the results, the block propagation delay is reduced by 90.1% and 87.6% for 50th and 90th percentile, respectively. However, this work uses unrealistic values for some of the parameters. In this article, we consider realistic values for the parameters.

Donet *et al.* [27] presented an analysis of collected data mined from Bitcoin. This work analyzes the network in terms of node geographic distribution, network stability, transaction propagation time, and block propagation time. According to this work, the Bitcoin network is homogeneously distributed all over the world, except in countries with very low population count. The work claims that although the delay in the Bitcoin network can be acceptable for regular nodes, nevertheless it is still too high for some miner nodes, which causes them to continue working on a block already found.

Decker and Wattenhofer [28] presented an analysis of Bitcoin from a networking perspective. The main observation from this paper is that network delay is the main cause of forks in Bitcoin. This work studies the delay cost based on parameters, such as block size, but does not provide an analytical approach: its results are gathered empirically from the Bitcoin network. Yasaweerasinghelage *et al.* [29] demonstrated the feasibility of using architectural performance modeling and simulation tools to predict the latency of blockchain-based systems.

VIBES is a visual simulation tool for blockchain networks [30]. This application can estimate performance measures with configurable blockchain parameters.

In [31], an event-based simulation model for Bitcoin overlay network is presented. To parameterize this model, large-scale measurement or the realm network of Bitcoin is performed. However, this work does not provide any theoretical framework for modeling the Bitcoin network.

Nasir *et al.* [32] performed a performance evaluation of two versions of Hyperledger Fabric (v0.6 and v1.0), measuring latency, execution time, and throughput, with various workloads. The research also studied the scalability of Hyperledger Fabric using Hyperledger Caliper, which is a benchmarking tool. Our work differs from this one as we develop an analytical model rather than present empirical results through benchmarking.

Thakkar *et al.* [33] presented an empirical study on the performance of Hyperledger Fabric with the goal of identifying potential performance bottlenecks. This work aims to identify the impact of different configuration parameters (e.g., block size, endorsement policy, etc.) on transaction throughput and delay. It puts the emphasis on optimizing Hyperledger Fabric v1.0 in light of its results.

The majority of research works already done in the context of performance analysis of blockchain systems are in agreement that the current Bitcoin network suffers from poor performance and thus improvement schemes are necessary in order to scale up.

Since information propagation delay is one of the most important performance measures, different proposals have been presented to improve it. Decker and Wattenhofer [28] and Stathakopoulou *et al.* [34] proposed some optimizations to Bitcoin, such as pipelining, increasing locality and using content distribution networks in order to speed up the network.

In [14], an optimization mechanism based on geographical proximity sensing clustering for fast information broadcasting is proposed. In this approach, the participating nodes are grouped into clusters based on geographical proximity. Then, strong connectivity and minimum network diameter are ensured using node attribute classification. Finally, the parallel spanning tree broadcast algorithm is used to broadcast the data among the participating nodes. The results of this paper demonstrate the lower delay of the solution compared to the current networks of Bitcoin and Ethereum.

A similar work about forming geographical clusters is presented in [15]. The difference between this work and the work mentioned above is the use of ping packets in order to detect proximity with considerably high P2P link bandwidth. However, none of these works present an analytical model, which can describe the relationship between configuration parameters (e.g., distance and bandwidth) and the information propagation delay.

Another approach to reducing the information propagation delay in blockchain networks is to deploy a central backbone of high-speed servers with high degree called a relay network.

The Bitcoin relay network [35] was the first well-known and operational relay network, which achieved a lower propagation delay by avoiding retransmission of known transactions as well as full block verification. This system was eventually upgraded as the Bitcoin FIBRE [16], which exploits cut-through routing with compact blocks and forward error correction [36] over user datagram protocol (UDP).

The Falcon relay network [17] is another relay network, which uses cut-through routing in order to increase the block propagation speed. This relay network is directly connected to 36.4% of the total hash power in Bitcoin [37].

Klarman *et al.* [18] presented a Blockchain distribution network (BDN), which increases throughput to thousands of transactions per second while reducing block propagation overhead without requiring any change to the blockchain protocol. The authors claim that BDN enables faster and gigabyte-sized block propagation, decreases inter-block times without increasing the risk of forks, reduces wasted miner effort, and improves fairness and decentralization compared to the aforementioned relay networks.

None of the aforementioned works provide a theoretical approach to performance modeling based on important configuration parameters. In contrast, this article models the Bitcoin P2P network using a random graph model in order to derive closed-form equations for two performance metrics: block propagation delay and network overhead. Moreover, we assess the impact of deploying relay networks on the performance of the system. To the best of our knowledge, this is the first work in which such a contribution is presented.

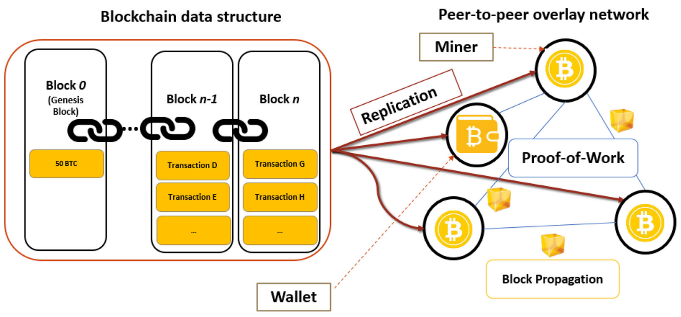


Fig. 1. Overview of the Bitcoin network.

B. *Performance Modeling of P2P Networks*

Beyond blockchains, peer-to-peer networks and random graphs have been studied to some extent. Kumar and Ross [38] studied the minimum achievable file distribution time in terms of basic network parameters, such as file size, number of servers, number of receiving nodes, and the upload and download capacities of participating nodes. In [39], the authors calculated the minimal time to fully disseminate a file of M parts from a server to N end users in a centralized architecture. However, none of the two above works address the problem of average delay of file dissemination in a peer-to-peer network in decentralized scenario.

One of the most relevant research to our own is [40]. This work investigates probabilistic flooding (randomly choosing next neighbor node) when the underlying network is a random graph. The distribution of diameters in Erdös–Rényi graphs is studied in [41]. The diameter is useful to calculate the global outreach time in a decentralized peer-to-peer network. However, this work does not give any explicit equation for calculating these parameters.

III. Background on Bitcoin

In this section, we briefly introduce Bitcoin and provide the main concepts required to understand this article. We study the information dissemination in Bitcoin networks, which can be broken down into two types: *transactions dissemination* and *blocks dissemination*. In this article, we develop an analytical model for block propagation in the Bitcoin network and leave transaction propagation as a future work. Transactions are the atomic units of information of any blockchain system, which are then grouped into blocks to be entered in the distributed ledger. Fig. 1 provides an overview of the different concepts related to the Bitcoin architecture.

Note that although the scope of this article (and thus this background section) is limited to Bitcoin, our theoretical model can be generalized to other blockchains, such as Ethereum [42].

A. *Blockchain Data Structure*

Fig. 2 illustrates the blockchain data structure. Blocks are linked together using the hash pointer of the previous block, starting from the root block (genesis block). In Bitcoin, the genesis block is hard-coded into the clients. These series of blocks are distributed, replicated, and stored in a ledger, which is located in

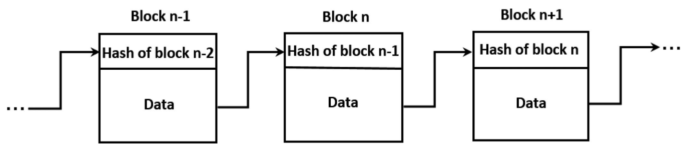


Fig. 2. Blockchain data structure.

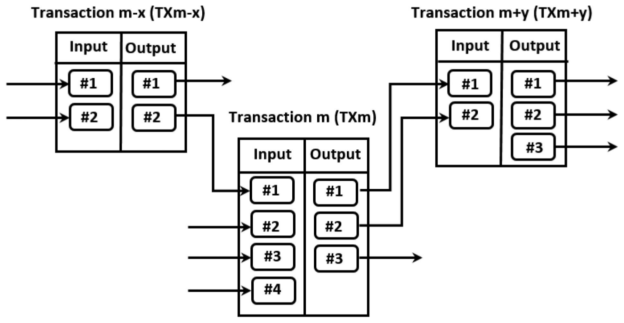


Fig. 3. Structure of a transaction.

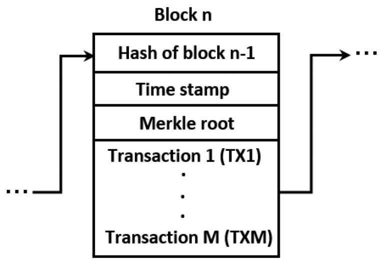


Fig. 4. Internal structure of a block.

all participating nodes. This ledger is necessary for verification of transactions embedded in each block. Participating nodes run over a peer-to-peer network and there is no intermediary node between them. Any node with access to this distributed ledger can read it and be notified about new data stored in the nodes.

B. Cryptocurrency Transactions

In Bitcoin, transactions describe the transfer of digital coins between users. Each transaction consists of several input and output records. Each input indicate funds to be spent from previous transactions and output records indicate the amount transferred to the specified addresses (which are generated from the public key of the recipient). To be valid, each transaction should be digitally signed by the private key of the sender.

In each transaction, the sum of inputs should be equal or greater than the sum of outputs. There exists also special transactions (called *coinbase*), which have no input and grant a reward to the miner who successfully added the block.

For example, transaction m in Fig. 3 takes as input #1 the output #2 from transaction $m - x$ and sends part of the funds to transaction $m + y$.

C. Blocks

As depicted in Fig. 4, each block consists of several components: hash of previous block, a timestamp, and transactions arranged to form a Merkle tree [1].

To be entered in the distributed ledger, individual transactions have to be embedded in the blocks. A subset of nodes in a Bitcoin network, called miners, gather transactions already propagated by users and group them into blocks to be added to the blockchain. Grouping the transactions in the blocks is an optimization, since a hash chain of blocks is significantly shorter than a hash chain of transactions.

To be eligible to propagate a proposed block as the next block in the chain, each miner has to show a PoW. To accomplish this, miners have to find a number called nonce. The hash of the nonce, combined with the hash of previous block, and the Merkle root of the tree containing the transactions proposed by the miner should be below a certain target threshold. In other words

$$H(\text{nonce}\|\text{H(prev\_block)}\|\text{Root}\{TX_1\|\ldots\|TX_M\}) \quad (1)$$

where $H(\text{prev\_block})$ is the hash of previous block. This means that the result of (1) should start with certain number of leading zeros. The difficulty of the mentioned puzzle is adjusted periodically.

For a given block, the miner who first solves the PoW will broadcast his block over the network. The nodes that receive the block must verify the incoming block and add it to the local copy of the blockchain, thus becoming the new blockchain head. Since Bitcoin uses a very wide peer-to-peer network, inconsistencies in this network are unavoidable. When multiple valid blocks are solved and disseminated at the same time, we say that a fork has occurred in the network, which is to likely to be resolved by the following block mined [43].

In order to reduce the block propagation delay in the Bitcoin network, the compact block proposal was introduced in BIP 152. According to this proposal, compact blocks have the same metadata as legacy blocks. The main difference is that the hash of the block transactions are disseminated instead of a full copy of the transactions.

D. Mempool

The nodes store unconfirmed transactions that are arriving from different links in a local memory pool (mempool). These transactions will remain in the mempool until they are included in the blockchain. Currently, the mempool can contain between 10^4 to 10^5 transactions, which is a churn rate of 1300–2400 transactions per block [44].

E. Overlay Network

Bitcoin operates on an unstructured peer-to-peer network. When a node joins the network, the Bitcoin protocol allows the node to collaboratively maintain peer-to-peer connections with other nodes to exchange blocks and transactions.

When a node receives a transaction or block, it verifies the transaction or block. If the transaction or block is valid, the node relays it to other nodes. Thus, the verification process is helpful to avoid denial of service attacks.

When a new node joins the network, it has no knowledge of the IP addresses of active nodes in the network. To discover this information, the new node queries a number of domain

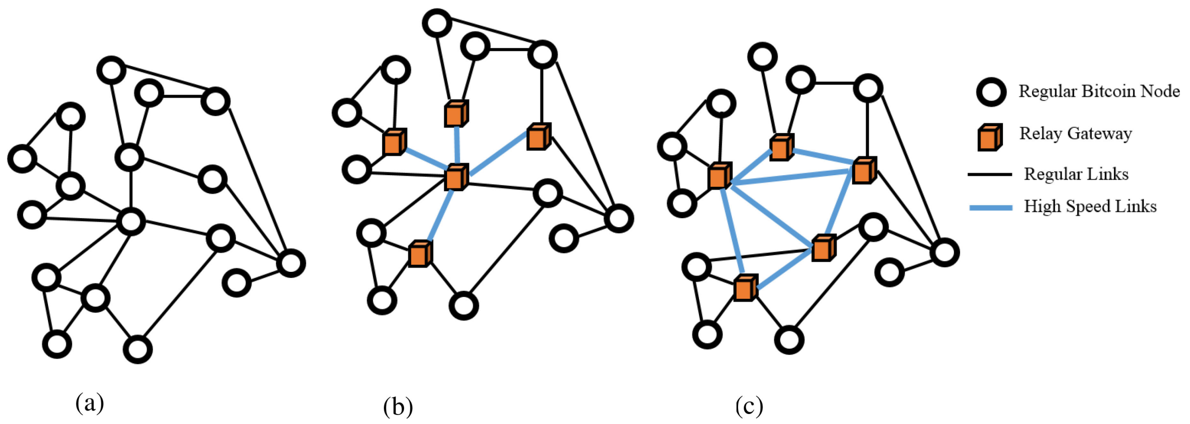


Fig. 5. Typical blockchain network with and without relay networks. (a) Regular Bitcoin overlay network. (b) Relay network with a centralized topology in a Bitcoin network. (c) Relay network with a decentralized topology in a Bitcoin network.

name system (DNS) servers (or DNS seeds), which are run by volunteer nodes in the Bitcoin community. Whenever a DNS server receives such a query, it responds with a number of DNS records with the IP addresses of bootstrap nodes that may accept the incoming requests. DNS servers can be configured to get the IP addresses of active nodes either automatically or manually.

Once connected, peers can send $addr$ (address) messages to other peers. This message consists of the IP address and port number of other nodes existing in the network. Also, newly joined nodes learn about other nodes by asking the neighboring nodes for known addresses or by listening to occasional advertisements of new addresses, which are broadcasted in the network. Nodes may leave the network or change their IP address silently. Hence, it is possible that new nodes may have to try several attempts before successfully connecting to a peer. This can impose a considerable delay to a node bootstrapping time. To avoid this problem, Bitcoin can use dynamic DNS seeds in order to get the IP addresses of nodes with a high probability to be available.

Before a node can validate transactions and blocks, it must download and validate all the blocks and transactions of the known longest version of the blockchain ledger. Furthermore, there is no need to download the genesis block, which it is already hard coded in the blockchain program. This process is called initial block download, which is done only once per new node. However, a node that was disconnected from the network for a significant amount of time may wish to run this process again.

Each participating node in the blockchain network maintains a connection pool, with connections to other peers active at all time. The minimum, maximum, and average number of connections are indicated by N_L, N_U, and M, respectively. If the number of connections is below the predefined amount of N_L, the node will randomly select connections from known neighboring nodes and will attempt to establish new connections. Also, if the node accepts incoming connections from other nodes, it can maintain these connections open. It is reported that nodes running Bitcoin have an average of 32 connections, which is far more than the default number of connections in Bitcoin, set to 8 [28].

F. Relay Networks

Generally speaking, a relay network is a set of global gateways with high velocity links, which form the backbone of the P2P network and connect to a large proportion of the network. Relay networks can propagate blocks and transactions much faster than the regular P2P network and thus increase the efficiency of information propagation. Relay networks can also be leveraged as a scalability solution for increasing the transaction throughput [18]. Analytically, relay networks can have a centralized or decentralized architecture. In a centralized architecture, the relay gateways interact with each other via a central orchestrator entity. But in decentralized architecture, the gateways are connected to each other in a P2P manner. Fig. 5 illustrates a typical blockchain network with and without relay networks. However, relay networks are accused of decreasing decentralization since they can censor certain nodes, wallets, and miners or discriminate among different nodes by filtering the information they spread.

G. Information Dissemination

In order to update the distributed ledger, transactions and blocks must be disseminated over the overlay network. To accomplish this, Bitcoin and most other blockchains use a gossip protocol [45]. In order to avoid saturating the network with redundant copies of transactions and blocks, Bitcoin employs a push-based approach to sharing data.

Currently, two kinds of block propagation protocols are being used in the Bitcoin network. The legacy block propagation protocol and compact block propagation protocol, which was introduced with BIP 152 [19]. The compact block protocol is currently being used by more than 98% of nodes in the Bitcoin network.[1]

Legacy Block Propagation Protocol: Sending nodes notify neighboring nodes about the availability of a transaction or block once they are verified. To accomplish this, a node that wants to

[1]See the presentation by core developer Greg Maxwell: https://people.xiph.org/~greg/gmaxwell-sf-prop-2017.pdf

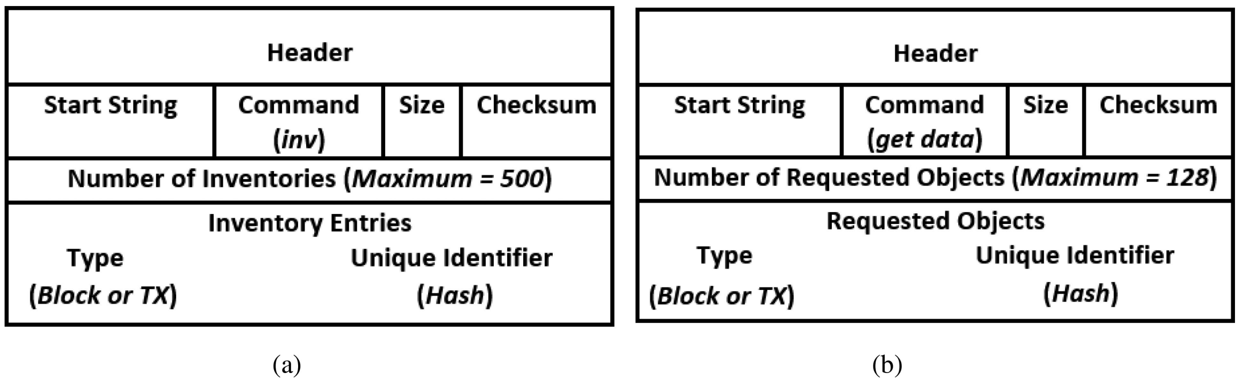


Fig. 6. Message structures of the inventory protocol. (a) inv message. (b) $get\ data$ message.

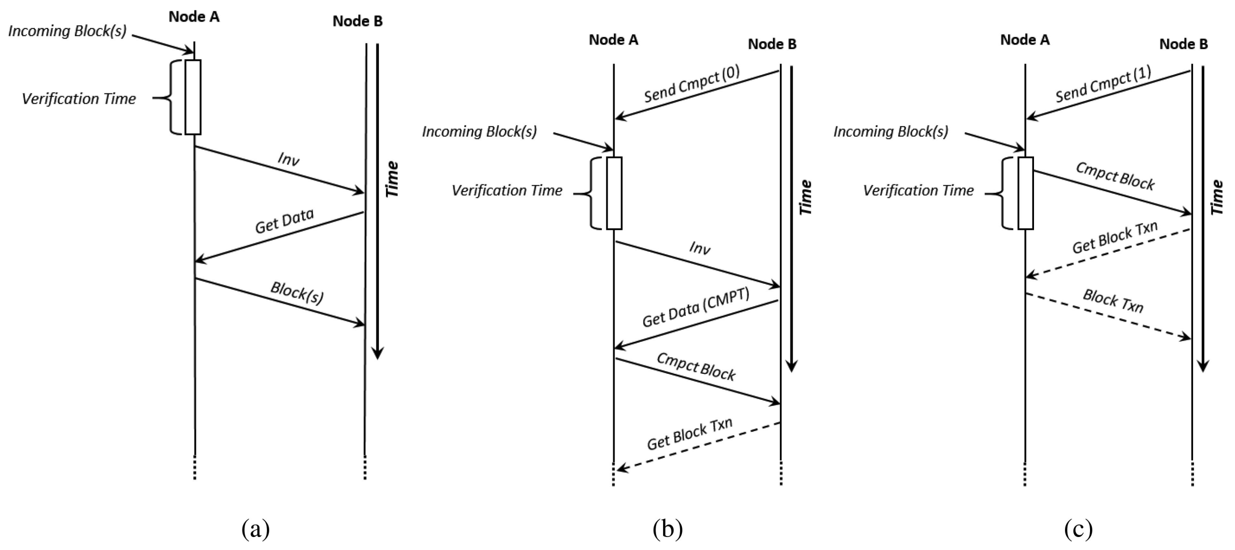


Fig. 7. Block propagation protocols in Bitcoin. (a) Legacy block propagation protocol. (b) Compact block propagation protocol [low-bandwidth mode (LBM)]. (c) Compact block propagation protocol [high-bandwidth mode (HBM)].

forward a transaction or block sends an inventory message (inv) to all nodes existing in its connection pool. The structure of the inv message is depicted in Fig. 6(a). An inv message consists of the hash of transactions or blocks that are now available and ready to be sent.

When a node receives an inv message for a block or transaction, which it does not have yet, it replies with a $getdata$ message. As depicted in Fig. 6(b), this message contains the hash of requested transactions or blocks.

Once the sender node receives the $getdata$ message, it will send the requested block or transaction to the receiver. This process is depicted in Fig. 7(a).

Compact Block Propagation Protocol: Compact block reduces the amount of bandwidth required for disseminating blocks in the Bitcoin network when the nodes are fairly synchronized and have already gathered a considerable amount of similar information (i.e., transactions) in their mempool. The main idea of this protocol is to let peers try to reconstruct entire blocks using their mempool content and a sketch of the blocks already received from the connected peers. In the case, transactions need to be sent over the network once only as they are not repeated during block propagation.

This protocol consists of two modes of operations: LBM) and HBM. In LBM [depicted in Fig. 7(b)], node B notifies node A that it intends to minimize bandwidth usage by sending a $sendcmp(0)$ message. When node A receives a new block, it fully validates it. If the block is valid, then it informs node B about the reception of the new block using an inv message. If node B has already received this block, it will ignore it. Otherwise, it will respond to the inv message using a $getdata\ (cmpct)$ message. Then, node A will send the header of the new block, hash of transactions, and transactions that B is missing (guessed by A). If B receives all of the transactions necessary for reconstructing the new block, the protocol stops. Otherwise, it will ask node A to send transactions that are still missing.

In HBM [depicted in Fig. 7(c)], node B notifies node A that it needs to receive blocks as soon as possible by sending a $sendcmp\,(1)$ message. When a new block arrives, node A starts to perform some basic validation (e.g., checking the block header). Then, it will send the header of the new block, the hash of transactions, and missing transactions to node B. Then, if the received information is adequate, node B will try to reconstruct the block. Otherwise, it will request information about missing transactions by sending a $getblocktxn$ message to node A. Node A will respond to this message by a $blocktxn$ message.

In both LBM and HBM modes, a node A follows a simple approach to guess missing transactions in node B: It checks the recently arrived block and sees which transactions were in the block but not in its mempool. These transactions are the transactions that will be missing in the neighboring peers with high probability (w.h.p.).

IV. Analytical Model

In this section, we propose a random graph model for performance modeling and analysis of the Bitcoin network. To accomplish this, we use a graph model introduced by Erdös and Rényi. This graph has properties suitable for modeling peer-to-peer overlay networks used by blockchain systems.

In this work, we define an overlay network as a graph $G(V, L)$, where V is the set of vertices and L is the set of links between nodes. For example, if there is a link between node i and node j, then $(i, j) \in L$.

Furthermore, we represent a random graph using $G_p(N)$, where N is the total number of nodes and p is the independent probability that there exist a link between any two selected nodes in the peer-to-peer overlay network. In this work, we assume that N is significantly large, as it is in Bitcoin network ($N \approx 10\,000$).

A. Random Graph Construction

To construct our random graph representation of the overlay network, we first start with a single node, which we call the initiator node, denoted by n_0. Then, a second node n_1 enters the system and establishes a link with n_0 with probability p. Naturally, this means that the probability of not having a link is $1 - p$. After this, a third node n_2 enters and can potentially connect n_0 or n_1, both with probability p. This process continues until node n_{N-1} enters the system and potentially establish links to the $N - 1$ existing nodes under the same probability p.

However, the random graph construction is an artificial mechanism to generate a topology of a fixed size, and does not reflect how the P2P system evolves in reality. Another way to interpret the random graph construction is as follows: first generate the total number of nodes (e.g., 10 000 nodes) as vertices in the graphs. Then, generate the set of all possible edges between any pair of vertices (called F). Finally, choose a subset of edges to be included in E such that $\frac{|E|}{|F|} = p$. In this way, we can obtain a random graph with desired number of nodes and the correct link probability p. In other words, our model only calculates performance values for the steady state (when the graph is fully constructed), and not the transient state of nodes joining.

It is obvious that for $p = 0$, we have an empty graph (all nodes are isolated with no edges), and that for $p = 1$, we have a complete graph with degree of $N - 1$ for each node. In general, our random graph model has several characteristics: if $p > \frac{1}{N}$, then a giant component exists w.h.p; and for $p \geq \frac{\log(N)}{N}$, all nodes become a part of the giant component and $G_p(N)$ becomes a connected graph.

These connectedness properties are vital for a blockchain network to operate properly. With a sufficient p value, each node has likely more than one path to reach any other node, using different outgoing links. In such a network, removing one link does not cause a network partitioning as the entire system stays connected. Network partitions are important to avoid in a blockchain network, since they guarantee that blockchain forks cannot resolve as long as the partitions stay isolated, thus compromising the integrity of the data on the ledger.

B. Achieving the Connectedness Properties

We now show how we can satisfy the abovementioned connectedness properties (i.e., obtaining a sufficiently high value of p) by appropriately configuring blockchain parameters.

Consider a connected blockchain network consisting of a set $\mathcal{N} = \{n_0, n_1, ..., n_{N-1}\}$ of N participating nodes. For convenience, assume that all links between nodes have the same point-to-point bandwidth of B. Each node maintains on average M open connections to other nodes. Since the network is connected, the average number of links between nodes can be calculated as follows:

$$\overline{L} = \frac{pN(N-1)}{2}. \tag{2}$$

Additionally, in a connected random graph, the average degree of each node equals to $p(N - 1)$ (i.e., $M = p(N - 1)$). Hence

$$p = \frac{M}{N-1}. \tag{3}$$

The above equation enables us to approximate p when we have the average number of default connections for each node. Additionally, according to (3), given the total number of participating nodes in a random graph, we can calculate the minimum degree or number of connections per node in order to have a connected graph w.h.p. as follows:

$$M > \frac{N-1}{N}\log(N). \tag{4}$$

In other words, to form a connected graph w.h.p., it is sufficient that

$$M \geq \left\lceil \frac{N-1}{N}\log(N) \right\rceil. \tag{5}$$

Fig. 8 illustrates the relationship between the total number of participating nodes in the graph and the minimum number of connections required to form a connected graph. According to this figure, since the Bitcoin network has approximately 10 000 participating nodes, if each individual node maintains at least 5 connections to other nodes, the network should be connected w.h.p.

C. Legacy Protocol Block Dissemination Analysis

In this section, we present a theoretical model for block propagation delay in the legacy propagation protocol. Although this protocol is deprecated for the current Bitcoin network,

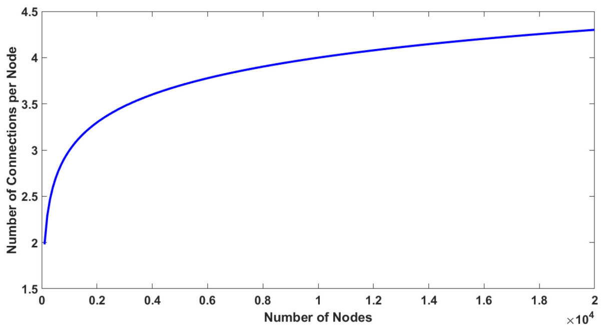


Fig. 8. Minimum number of connections for connectedness.

we include this analysis for the sake of completeness and to demonstrate that this model serves as a foundation that can be generalized for other blockchain protocols.

Assume a P2P overlay network that consists of N participating nodes. Suppose one miner node mines a block at time $t = 0$. We denote this initial node by n_0. According to Section III-G, node n_0 sends an inv message to the set $\mathcal{W}_1 = \{n_1^1, n_2^1, \ldots, n_M^1\}$ of neighboring nodes in its connection pool. We assume that the sending node sends the inv message to its neighbors in succession with a very small delay ϵ between each message. Since this is the first time that the inv message is being sent for this block, none of the $\mathcal{W}_1$ neighboring nodes have the block and will respond the inv message with the $getdata$ message. Then, the node n_0 will send the complete block to the requesting neighbors. We call this first step of the block dissemination process Wave 1, as illustrated in Fig. 9(a).

Upon completion of Wave 1, all involved nodes during this initial wave (i.e., nodes which received the block during the first wave) validate the block and send an inv message to the neighboring nodes in each of their own connection pool. We call this Wave 2, as illustrated in Fig. 9(b). For convenience, we assume that nodes do not know or remember the initial node n_0, but they do remember the node which they received the block from and will not send an inv message back to the sender.

We observe that some of the nodes receiving an inv message during Wave2 may have already received the block in Wave1 and will not respond the inv message with a $getdata$ message. Therefore, the forwarding probability, which is the probability that a node replies affirmatively to the inv message with a $getdata$ message immediately after ending the Wave1, can be calculated as follows:

$$p_{f2} = \alpha \frac{N - 1 - |\mathcal{W}_1|}{N - 1} \tag{6}$$

where $0 \le \alpha \le 1$ is the reachability factor of the nodes in the network. We use this factor to take into account the connections lost during the three-way message exchange mentioned in Section III-G or due to message loss (e.g., congestion and link outage). For instance, if $\alpha = 0.95$, this means that 95% of the messages on average will reach the destinations during the three-way inventory message exchange.

Accordingly, the set of receiver nodes, which send back a $getdata$ message, can be represented as follows: $\mathcal{W}_2 = \{n_1^2, n_2^2, \ldots, n_{|\mathcal{W}_2|}^2\}$, where $|\mathcal{W}_2| = \lceil p_{f1} p_{f2} M^2 \rceil$.

Each subsequent wave will follow the same pattern as Wave2. Fig. 9(c) shows an example for some Wave$k$, where only the nodes that received the block during the previous Wave$k-1$ will send inv messages during the current wave.

In general, for each wave, we define forwarding probability p_{fi} as follows:

$$p_{fi} = \alpha \frac{N - 1 - \sum_{j=0}^{i-1} |\mathcal{W}_j|}{N - 1} \quad (i \ge 1 \text{ and } |\mathcal{W}_0| = 0) \tag{7}$$

where $|\mathcal{W}_j| = \lceil M^j \prod_{k=1}^{j} p_{fk} \rceil$. The numerator of the above equation gives the number of nodes that have not received the block at the beginning of Wave i. This means that during Wave i, the nodes that receive the inv message will respond it with a $getdata$ message with probability p_{fi}. Accordingly, the inv message will be timed out with probability $1 - p_{f_i}$. Note that $p_{f1} = 1$ since all the nodes that are connected to the initial node n_0 will receive the block during the first wave.

We define parameter C_i as the partial coverage, which means the coverage of nodes during the Wave i (i.e., the number of nodes that receive the block during Wave i) as

$$C_i = |\mathcal{W}_i| = \left\lceil M^i \prod_{j=1}^{i} p_{fj} \right\rceil. \tag{8}$$

Additionally, we define cumulative coverage as follows:

$$C_i^T = \sum_{j=1}^{i} |\mathcal{W}_j| = \sum_{j=1}^{i} \left\lceil M^j \prod_{j=1}^{i} p_{fj} \right\rceil \tag{9}$$

where C_i^T it the total number of nodes that have received the block at the end of $Wave\ i$.

The set of receiver nodes which send back $getdata$ message after receiving inv message during Wave i can be represented as $\mathcal{W}_i = \{n_1^i, n_2^i, \ldots, n_{|\mathcal{W}_i|}^i\}$. According to (7), p_{fi} can be calculated as follows:

$$p_{fi} = \alpha \frac{(N-1) - \sum_{j=1}^{i-1} M^j \prod_{k=1}^{j} p_{fk}}{N - 1} \quad (1 < i \le K) \tag{10}$$

where K is the total number of waves needed for a block to be distributed among all nodes in a connected peer-to-peer overlay network. Note that the above equation is valid as long as $C_i^T \le N - 1$. For the case where $C_i^T > N - 1$, we need to adapt and reform (10) as follows:

$$p_{fi} = \beta\alpha \frac{(N-1) - \sum_{j=1}^{i-1} M^j \prod_{k=1}^{j} p_{fk}}{N - 1} \quad (1 < i \le K) \tag{11}$$

where β is the adaptation factor. In fact, (11) can be used for calculating all forwarding probabilities where β is expressed as follows:

$$\beta = \begin{cases} 1 & C_i^T \le N - 1 \\ \frac{N-1-C_{i-1}^T}{C_i} & C_i^T > N - 1 \end{cases}. \tag{12}$$

Equation (11) is recursive and enables us to calculate the number of waves required for global outreach of a block ($x\%$

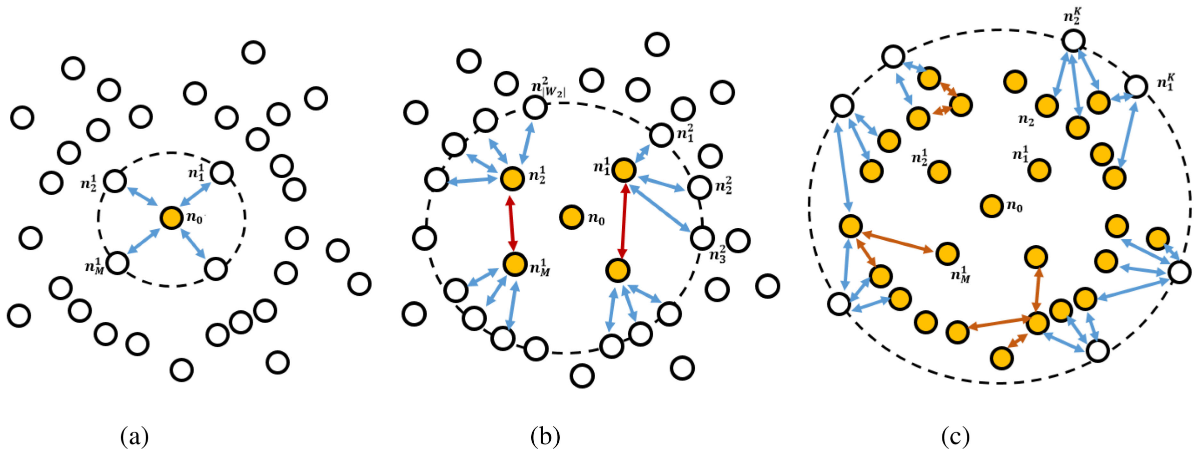


Fig. 9. Different waves of block dissemination in the proposed blockchain network. Colored nodes are the nodes that have already received the block. Blue arrows show successful block transfers. Red arrows show timed out inv messages. (a) Wave 1. (b) Wave 2. (c) Wave k.

block propagation)

$$\sum_{i=1}^{K} M^i \prod_{j=1}^{i-1} p_{fj} = x(N-1). \tag{13}$$

After the calculated number of waves have occurred, if no conflicting blocks were found during the entire dissemination process of the current block, we can guarantee that forks resulting from network delays or partitions cannot occur for this block. Note that malicious attacks (such as 51% attacks) to intentionally introduce forks in the blockchain are out of the scope of this model.

Algorithm 1 shows how we can obtain the number of required waves for fully disseminating a block, as well as the forwarding probability in each wave (11). This algorithm computes forwarding probabilities in a recursive manner.

D. Performance Metrics Analysis of the Legacy Protocol

Consequently, key features and performance measures can be derived in terms of block dissemination delay and traffic overhead. We calculate block dissemination delay as follows:

$$D = K(D_v + X_I + Y_I + D_g + X_G + Y_G + D_b + X_B + Y_B) \tag{14}$$

where D_v is the block validation time, X_I is the transmission delay, and Y_I is the signal propagation delay of inv messages. D_g is the time interval taken by a node to process an inv message (i.e., lookup if the content of the message are present locally or not) before replying to the inv message. Additionally, X_G and Y_G are the transmission delay and propagation delay of the $getdata$ message. Similarly, D_b is the time interval taken to process a $getdata$ message before replying with the requested block. Similar to previous ones, X_B and Y_B are the transmission delay and average propagation delay of the sent block, respectively. For convenience, we assume that $Y_I = Y_G = Y_B$.

Algorithm 1: Calculating the number of waves

```
Input  : The values of $N$, $M$
Output: $K$ and Matrix $\mathcal{P}[p_{f1}...p_{f_K}]$
1  $\mathcal{P}[1] \leftarrow 1$
2  $K \leftarrow 2$    // We assume $M < N-1$ and hence $K \geq 2$
3  $\mathcal{C}[1] \leftarrow M$
4  $Ctrl \leftarrow 1$    // Controller
5  while $Ctlr = 1$ do
6      for $i = 2$ to $K$ do
7          $\mathcal{P}[i]=\frac{(N-1)-\sum_{j=1}^{i-1} M^j \prod_{k=1}^{j} \mathcal{P}[k]}{N-1}$
8          $\mathcal{C}[i]= M^i \prod_{j=1}^{i} \mathcal{P}[j]$
9          if $\sum_{j=1}^{i} \mathcal{C}[j] \leq N-1$ then
10             $K = K+1$
11         else
12             $\mathcal{P}[i]=\frac{N-1-\sum_{j=1}^{i-1} \mathcal{C}[j]}{\mathcal{C}[j]} \times \frac{(N-1)-\sum_{j=1}^{i-1} M^j \prod_{k=1}^{j} \mathcal{P}[k]}{N-1}$
13             if $\mathcal{P}[i] \geq 0$ then
14                 $K = K+1$
15                 $\mathcal{C}[i]= M^i \prod_{j=1}^{i} \mathcal{P}[j]$
16             else
17                 $Ctrl \leftarrow 0$
18             end
19         end
20     end
21 end
22 return $K$ and $\mathcal{P}$
```

Given the bandwidth B for each link of the random graph, we can rewrite (14) as follows:

$$D = K\left(D_v + \frac{S_i}{B} + Y_I + D_g + \frac{S_g}{B} + Y_G + D_b + \frac{S_b}{B} + Y_B\right) \tag{15}$$

where S_i, S_g, and S_b are the sizes of inv message, $getdata$ message, and the transmitted block, respectively.

Moreover, the network traffic overhead for Wave i can be calculated as follows:

$$H_i = \frac{(1-p_{fi})M^i \prod_{j=1}^{i-1} p_{fj}}{N-1} \quad (1 \le i \le K). \tag{16}$$

The overall packet exchanging traffic overhead is the sum of the traffic overheads in each wave given by

$$\overline{H} = \frac{1}{N-1} \sum_{i=1}^{K} \left[(1-p_{fi})M^i \prod_{j=1}^{i-1} p_{fj} \right]. \tag{17}$$

Note that the packet exchanging traffic overhead reveals the burden of sending redundant inv messages in later waves to nodes that have already previously received the block. Since Bitcoin is an unstructured and decentralized P2P network that relies on gossiping, there is no coordination that allows nodes to efficiently decide which neighbors are likely to have received the block already without at least contacting them with an inv message.

Also note that for this model, we focus principally on the dissemination of a single block at a time. An inv message can also be used to exchange multiple blocks at the same time. However, we argue that the single block use case, as discussed here, is much more common in Bitcoin due to the long block time of 10 min, which diminishes the chance of requiring multiple blocks to be disseminated in the same inv message.

E. Compact Block Protocol Block Propagation Analysis

In this section, we present an analytical model for block propagation delay using the compact block propagation protocol of Bitcoin. As already shown in Fig. 7, the compact block protocol operates in two modes of operation: low bandwidth (LBM) and high bandwidth (HBM). Each node in the network establishes regular connections to some peers in its connection pool using LBM and tells a subset of the remaining nodes to push newly arrived blocks (i.e., by sending inv message). Thus, LBM operates similarly to the legacy protocol. The main differences are the size of full blocks compared to the compact ones and the two additional messages exchanged in case of block reconstruction failure ($getblocktxn$ and $blocktxn$). In this section, we assume that the block reconstruction failure rate is very low and negligible (see footnote 1). Given a P2P overlay network with N participating nodes, suppose each node has established on average M LBM connections and m HBM connections (i.e., m peer nodes will push compact blocks using a $sendcmp$ message). We assume that all links between nodes have the same P2P bandwidth of B. Since block dissemination using LBM is similar to the block dissemination in the legacy protocol, we can use the concept of waves for LBM as we did in Section IV-C. But for HBM, we adapt our model by introducing the concepts of long waves and short waves. Each long wave involves three message transfer as required to deliver the compact block as shown in Fig. 10. Additionally, each short wave involves just a compact block transmission. Therefore, each long wave time duration is essentially three times the duration of a short wave as depicted in Fig. 10. In other words, one block transfer in LBM is equal to three block transfers in HBM, since the size of the transferred packets are comparatively small and have a negligible transmission delay compared to the signal propagation delay. Since we have assumed an equal signal propagation delay for the all links, this assumption is reasonable.

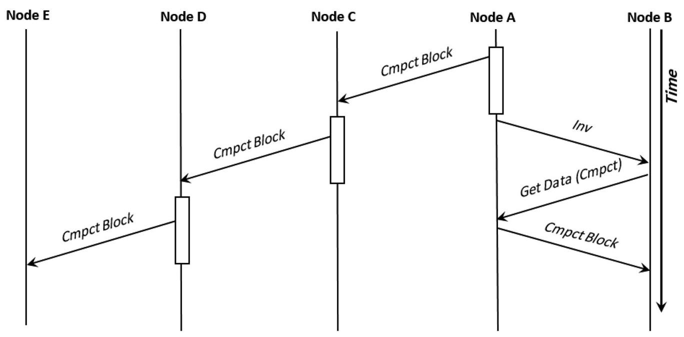


Fig. 10. Example to show the block propagation time in different modes of operation: node A has established an LBM connection to node B as well as an HBM connection to node C. In the same way, node C has established an HBM connection to node D, and node D has established an HBM connection to node E. Nodes B and E receive the block at almost the same time.

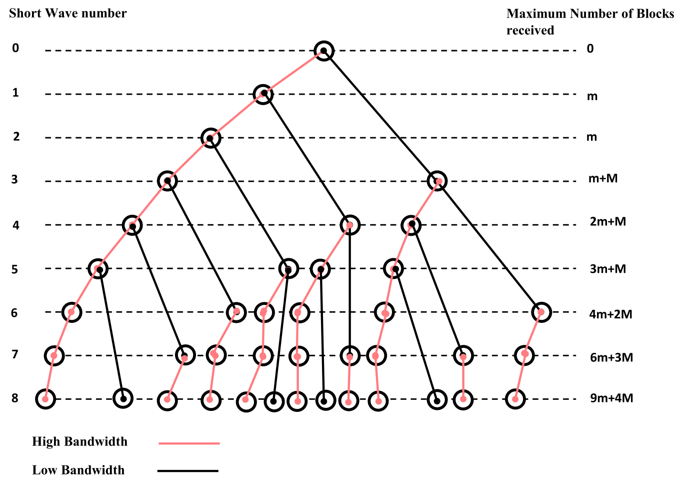


Fig. 11. Long waves and short waves in block propagation. Each long wave is equivalent to three short waves. Red links are the links operated in HBM and black ones are the links operated in LBM.

Suppose n_0 is the initiator node that intends to disseminate a newly mined block. It will send the compact block to M peers in LBM and to m nodes in HBM. At the end of the first short wave, m nodes will receive the compact block and immediately will start to sending the compact block to m of their already selected peers in HBM and to M of the remaining peers in LBM. According to Fig. 11, at the end of the second short wave, at most $2m$ nodes will have the new block. We call the number of nodes that receive a block during a short wave i as the *wave coverage* and denote it by C_i. Also, we use the term *total coverage* for the total number of nodes that have received the block at the end of short wave i and denote it by N_i. At the end of the third short wave, the first set of the nodes that are connected to the initiator node in LBM will receive the compact block (M nodes). Additionally, m more nodes operating in HBM will receive the block at the end of the third short wave (see Fig. 11). Therefore, the coverage of the third wave and total coverage of the third wave will be at most $m + M$ and $3m + M$ nodes, respectively. According to the above discussion, coverage of each wave is calculated as follows:

$$C_i = X_i m + Y_i M \tag{18}$$

where X_i and Y_i are nonnegative integers and can be calculated as follows:

$$\begin{aligned} &X_1 = 1 \\ &X_i = X_{i-1} + Y_{i-1} \quad i \geq 2 \end{aligned} \tag{19}$$

and

$$Y_i = \begin{cases} 0 & 1 \leq i \leq 2 \\ 1 & i = 3 \\ X_{i-3} + Y_{i-3} & i \geq 4 \end{cases}. \tag{20}$$

According to the above discussion, $C_1 = N_1 = m$. Thus, we can estimate the total coverage at the end of the second short wave as follows:

$$N_2 = N_1 + \left(1 - \frac{N_1}{N-1}\alpha\right) m \tag{21}$$

where $\frac{N_1}{N-1}$ is the fraction of the nodes that have received the block so far. Accordingly, $1 - \frac{N_1}{N-1}$ yields the probability that receiving nodes have not already received the block and therefore will accept it. Consequently

$$N_k = N_{k-1} + \left[\left(1 - \frac{N_{k-1}}{N-1}\alpha\beta\right)(X_k m + Y_k M)\right] \tag{22}$$

where N_k is the total coverage at the end of the short wave k. Finding the maximum amount of N_k that satisfies the condition $N_k \leq N - 1$ will give an estimation of the block propagation delay based on the number of short waves

$$\beta = \begin{cases} 1 & N_k \leq N-1 \\ \frac{N-1-N_{k-1}}{N_k} & N_k > N-1 \end{cases}. \tag{23}$$

The number of short waves required for global outreach of the compact block can be estimated as follows:

$$K = \{\min k | N_k \geq N - 1\}. \tag{24}$$

F. Performance Metrics Analysis of BIP 152

The compact block dissemination delay can be estimated as follows:

$$D = \left\lfloor \frac{K}{3} \right\rfloor (D_p + X_I + Y_I + X_G + Y_G + X_B + Y_B) + \left(K - 3\left\lfloor \frac{K}{3} \right\rfloor\right)(X_I + Y_I) \tag{25}$$

where D_p is the processing delay of the compact block and consists of block validation time and internal processing delay in nodes.

Algorithm 2: Calculating the number of short waves

```
   Input  : The values of $N$, $M$, $m$
   Output: $K$ and Matrix $\mathcal{N}[N_1,..., N_K]$
1  $\mathcal{N}[1] \leftarrow m$
2  $\mathcal{X}[1] \leftarrow 1$
3  $\mathcal{Y}[1] \leftarrow 0$, $\mathcal{Y}[2] \leftarrow 0$, $\mathcal{Y}[3] \leftarrow 1$
4  $\mathcal{C}[1] \leftarrow m$ // We define this matrix for the coverage of
     each short wave
5  while $\mathcal{N}[i] \leq 0.9N$ do
6      $\mathcal{X}$[i+1]= $\mathcal{X}$[i]+$\mathcal{Y}$[i]
7      $\mathcal{Y}$[i+3]= $\mathcal{X}$[i]+$\mathcal{Y}$[i]
8      $\mathcal{C}$[i+1]= $m\mathcal{X}$[i+1]+$M\mathcal{Y}$[i+1]
9      $\mathcal{N}$[i+1]= $\mathcal{N}$[i]+ $\mathcal{C}$[i+1]($\frac{\mathcal{Y}[i+1]}{N-1}$)
10     $i = K$
11 end
12 return $K$ and $\mathcal{N}$
```

G. Assumptions

Our analytical solution models block propagation in several waves. This means that all nodes receive and release blocks from/to their neighbors at the same time. Additionally, it assumes the latency numbers for each pair of nodes. In reality, the latency between peers depends on different factors, such as geographical distance between peers and the bandwidth of the links. Moreover, block validation time varies in different nodes depending on their computational power. Although the network is overall asynchronous, we argue that the propagation of a single new block can be approximated as a synchronous process. Furthermore, Bitcoin has only been proven to be correct in a partially asynchronous system [46], which supports our assumption. Note that in a real system, different configuration parameters are tightly tangled together such that a theoretical model of the system becomes intractable. Hence, our model aims to simplify to a certain degree while still providing approximate, yet accurate results. Thus, we mostly rely on the average values of the metrics. Despite of these assumptions, the results of our model closely match the results of the data mined from the real network of Bitcoin.

V. EVALUATION

In this section, we first validate our theoretical model by comparing the theoretical results with simulation results as well as the values of the empirical data mined from the real Bitcoin network.

We then conduct a performance analysis to assess the impact of various parameters on the performance of the Bitcoin network. We study the impact of the average number of connections per node, the network size and the network bandwidth, and the reachability factor on the performance of the Bitcoin network. We also measure the network traffic overhead for different number of connections per node. Finally, we investigate the impact of deploying a relay network on the system performance and study the overall performance of the network when modifying other parameters.

A. Settings and Implementation

Implementation: We implemented a discrete event-based simulation using Omnet++ [47]. We developed a C# code to automatically get the data from the provided application program interface. We also used MATLAB for theoretical analysis.

Dataset: We compare numerical values of our theoretical model and simulation results to real empirical measurements provided in [48]. In this dataset, 90% denotes that 90% of inv messages for a block were observed within the given time from the first 1000 nodes. We extracted the data of almost 15 000 Bitcoin blocks from block number 507016 through 522429 in the main chain. These block numbers correspond to blocks were generated since February 1, 2018 until June 12, 2018. These blocks comprise different number of transactions embedded in them and hence have different sizes.

Parameters: The block processing time (i.e., block validation time plus other internal processing delays) depends on the block size. After observing the block propagation delay data reported in [48], we infer that the average block propagation delay increases sublinearly with the increase of block size. Since the impact of increasing the compact block size on the propagation delay (X_B or compact block transmission delay) is negligible, we can conclude that the processing delay increases sublinearly with increasing the block size. Hence, we estimated an average time of 4 ms for processing of a block (around 1 ms for block validation) with a size of 0.1 MB (1800-KB size of the compact block) and extrapolated it for other blocks with different sizes.

B. Model Validation Results

Although different nodes in Bitcoin use different links with various bandwidths, it has been shown in [37] that the median provisioned bandwidth of nodes in Bitcoin network was 33 and 56 Mb/s for early 2016 and early 2017, respectively. The latter is thus 1.7 times the first one. Accordingly, the average bandwidth for nodes that use IPv4, IPv6, and Tor addresses are reported as 73.1, 86.5, and 4.7 Mb/s, respectively. We also observed that almost 83%, 13%, and 4% of nodes in the Bitcoin network during the time interval related to our data set used IPv4, IPv6, and Tor addresses, respectively. Using a weighted mean and multiplying it by 1.7 in order to scale to early 2018, we estimated provisioned bandwidth as 123 Mb/s in early 2018. However, due to the small size of compact block, this parameter becomes less important and the compact block transmission delay can be neglected.

In [28], it has been stated that a $bitcoind$ node that accepts incoming connections has an average of 32 connections, which is much more than the default number of 8 connections for each node. In this experiment, we thus considered $M = 32$ for theoretical analysis and simulation.

To estimate average signal propagation delay in the Bitcoin network, we consider blocks with the smallest size in the dataset ($S_b < 1$ kB) and obtained 20 ms as the average value for signal propagation delay.

We also consider the same size of 37 B for inv and $getdata$ messages according to [49]. Also, in the theoretical analysis and simulation, we assume that there is no packet loss, and, consequently, no packet retransmission. In all of the experiments, we consider a network size of 10 000 nodes, which is near the Bitcoin network size.

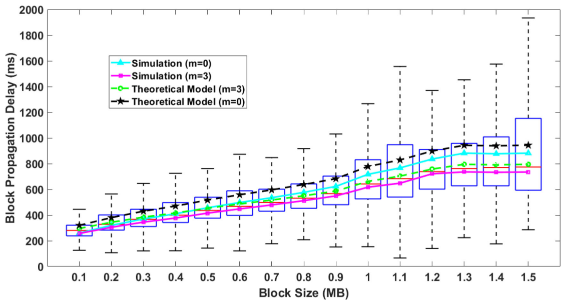


Fig. 12. 90% block propagation delay versus block size.

To generate an Erdös–Renyi random graph with the two parameters of N and p, we used the network description (NED) language in Omnet++. To fix the average number of connections per each node, we counted the number of inv messages in the simulation log file using awk and divided them by N. This approach is sound because each node sends inv messages to M neighboring nodes in his connection pool.

The results of simulation and theoretical analysis are depicted in Fig. 12. In this figure, the horizontal axis indicates the block size where each point on this axis represents a range of block sizes with length of 0.1 MB. For instance, block size = 0.2 represents the interval between 0.1 and 0.2 MB. The box plot shows the propagation delay of blocks in the real Bitcoin network versus block sizes. We removed outliers to clean up the graph. Bitcoin Core with BIP 152 recommends peers to establish their last three connections in HBM. Therefore, we did the experiments with this value ($m = 3$). However, we have also tested other values, such as the case with no HBM connections ($m = 0$). For $m = 1, 2$, results fall between the results of the mentioned experiments. As it can be seen, the results of simulation and theoretical values are almost identical with high accuracy with respect to empirical results. By calculating the slope of the graph (see Fig. 12) at different points, the results show that block propagation delay increases sub-linearly with respect to block size. For the smaller blocks, the size of the quartiles are comparatively smaller. The main reason is that in small block sizes, the delay stems from the signal propagation delay. But for big blocks, the delay originates from the block processing time.

C. Performance Analysis of the Bitcoin Network

To study the effect of the default number of connections on block propagation delay, we conduct an experiment using our theoretical model for different values of M ($M = 8, 16, 32, 64$). Additionally, we assume that no connection is set in HBM ($m = 0$). $M = 8$ refers to the default number of connections in the original protocol, whereas $M = 32$ refers to the average number of connections per node observed in practice. $M = 64$ and $M = 16$ can be considered as proposals for doubling or halving the average number of connections per node in the current Bitcoin network. Like in the previous experiment, we use $N = 10\,000$.

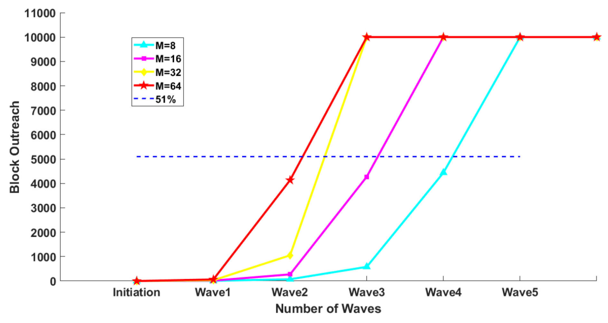


Fig. 13. Block outreach versus number of waves (90%).

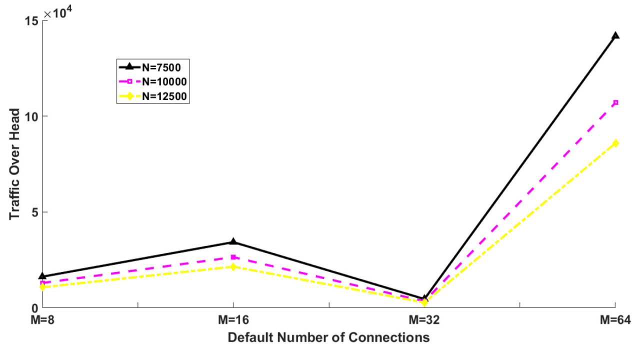


Fig. 14. Network traffic overhead (90%).

Fig. 13 depicts the results of this experiment. The network with $M = 8$ is significantly slower than others and takes five waves to fully disseminate the block. While the network with $M = 64$ is considerably faster than networks with $M = 16$ and $M = 8$, there is no significant differences between the networks with $M = 64$ and $M = 32$, since both fully disseminate the block in two waves. This means that increasing the average number of connections in the Bitcoin network from 8 to 32 significantly increases the block propagation speed. In all settings evaluated, the block does not reach a majority of nodes (5100 nodes) before the last wave. Dividing the values obtained in this experiment by the total number of nodes in the network will yield similar curves for the cumulative distribution function of the block outreach.

To study the impact of the average number of connections per node on the traffic overhead, we carry out another set of experiments for $N = 7500$, 10 000, and 12 500. The results of this experiment are depicted in Fig. 14. As can be seen in the figure, the optimal traffic overhead is for $M = 32$. This means that this configuration achieves a minimal number of redundant inv messages (inv messages that timed out). This can be explained by the fact that in a network with a lower M, each node sends fewer messages to neighbors and, according to (10), a bigger forwarding probability is achieved. On the other hand, an increased number of connections creates a faster network with a lower number of waves, which leads to an increase in total traffic overhead. Therefore, there is a tradeoff between block propagation delay and network overhead.

According to Fig. 14, the other choice for low traffic overhead can be $M = 8$. But according to Fig. 13, this will create a slow

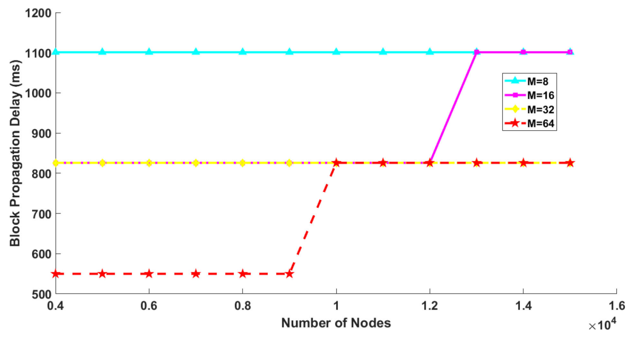


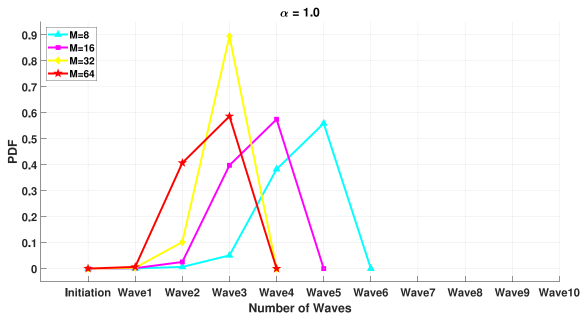
Fig. 15. 90% block propagation delay versus network size.

network. Thus, we claim that $M = 32$ is the best choice for the current Bitcoin network.

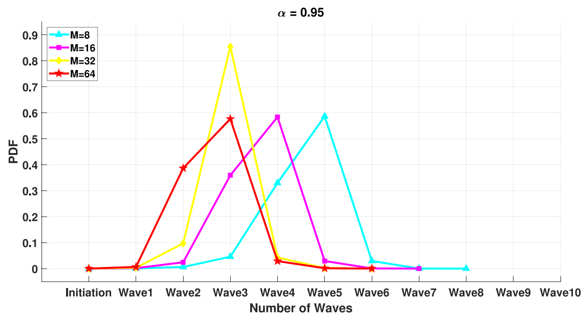
Another point of interest is the impact of network size or the number of participating nodes on the average block propagation delay. To study this, we conduct another experiment and calculate the average block propagation delay (90%) for different number of nodes (intervals of 1000 nodes) and repeat it for different number of connections. The results are depicted in Fig. 15. Increasing the number of connections will not always lead to a faster network. For the current size of Bitcoin, we expect roughly the same propagation delay when $M = 16, 32, 64$, but with different costs of traffic overhead. For smaller sizes, $M = 64$ is significantly faster, whereas there is no considerable change for the networks with $m = 16$ and $M = 32$. For sizes bigger than that of the current network, $M = 32$ and $M = 64$ will have the same impact on the block propagation delay with different amounts of traffic overhead. It is unlikely to have less than 5000 or more than 13 000 nodes in the Bitcoin network. Therefore, we limit the evaluated intervals to 4000–5000.

It is useful to study the Bitcoin network when it operates in nonideal conditions. To model such situation, we decrease the reachability factor α, which corresponds to a decrease in forwarding probability. As mentioned in Section IV-C, a decrease in forwarding probability implies that some of the blocks have not reached their destinations or are rejected by the candidate receiver nodes.

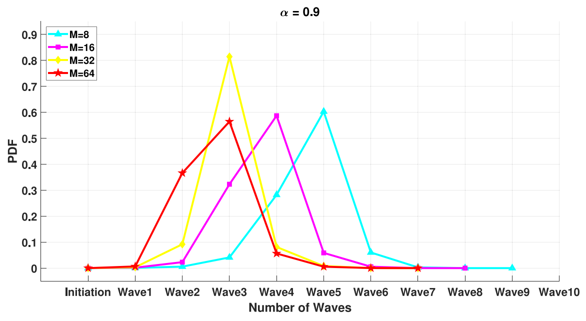
Dividing the coverage of different block propagation waves (i.e., the number of nodes that receive the block during a wave) by the network size (i.e., the total number of participating nodes) will yield the PDF of the rate at which the nodes receive the block. Fig. 16(a) plots this function for the ideal conditions (i.e., $\alpha = 1.0$). We repeated this experiment for different number of connections per node. For all values of M, the block is fully propagated over the network without any long tail in the PDF. If we slightly decrease α, a long tail appears on the PDF curve as depicted in Fig. 16(b). This means that more waves are needed for 100% block propagation. Nevertheless, more than 95% of the nodes still receive the block as they would in an ideal network. Lowering α will lead to longer tails, which means a slower 100% propagation delay [see Fig. 16(c) and (d)]. Moreover, as plotted in Fig. 16(d), when $\alpha = 0.85$, less than 90% of nodes will receive the block during the time required for 100% propagation under ideal conditions.



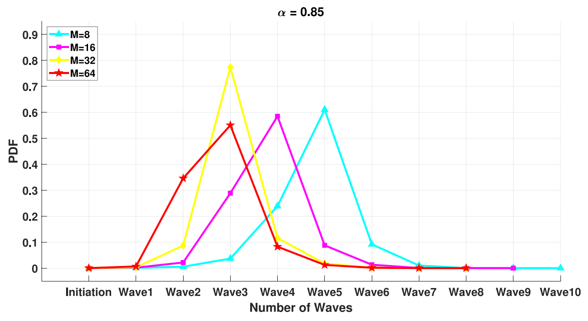
(a)



(b)



(c)



(d)

Fig. 16. PDF for waves at which Bitcoin nodes receive the block. (a) $\alpha = 1.0$. (b) $\alpha = 0.95$. (c) $\alpha = 0.9$. (d) $\alpha = 0.85$.

D. *Sensitivity Analysis of the Relay Network*

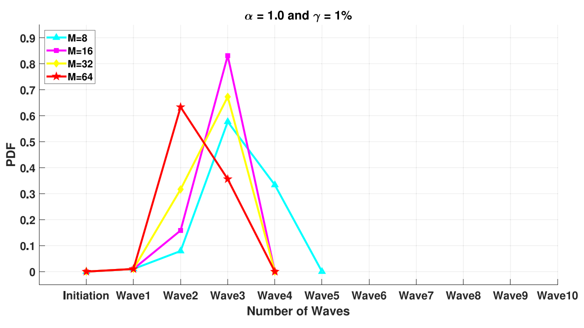
When a relay network is present, interested miners can maintain a link to one of the gateways and send any mined block directly to the relay network. The relay network will then relay copies of the block to the rest of the relay network using high speed links, thus increase the block propagation speed.

Consider that an ideal relay network is deployed in the network, where gateways are homogeneous with infinite bandwidth (see Fig. 5) with very low latency. To model this network, we adapt our basic analytical model aforementioned in Section IV-C as follows. Since a node that maintains a high speed link to the relay network does not need to exchange inventory messages with the relay gateways, it can immediately send the newly mined block to the gateway. Additionally, since the relay network does not need to validate the blocks multiple times, we assume that the internal block propagation between relay gateways is almost instantaneous: the latency from the moment the miner sends the block to the relay gateway until the time at which the relay network starts to propagate the block over the network is almost zero. We define the parameter *relay network size* as the percentage of the nodes in the network that are directly connected to the relay network and denote it by γ. For instance, if 100 nodes of the Bitcoin network are connected to the relay network, then $\gamma = 1\%$ (for a total size of 10 000 nodes). Consequently, we can consider the relay network as an initiator node with $\gamma(N-1)$ connections to other neighboring nodes. We will treat with all nongateway nodes as described in Section V-A.

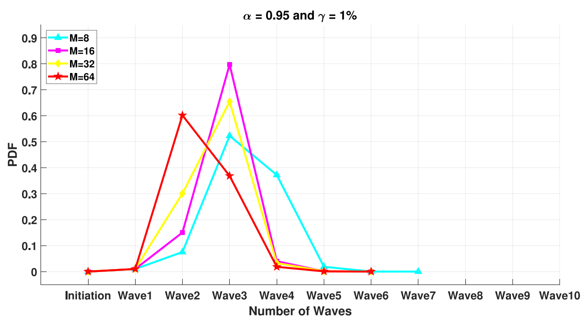
We carry out a set of experiments in order to study the impact of the relay network size on the performance of the Bitcoin network with different configurations for the average number of connections per node as well as reachability. In this set of experiments, we employ three values for the relay network size ($\gamma = 1\%,\ 2.5\%,$ and 5%). Simultaneously, we change the value of α and decrease it gradually to study the effect of message loss on the network performance. We repeat these experiments for different number of connections per node. The results are plotted in Figs. 17–19.

As depicted in Fig. 17(a), when $\gamma = 1\%$, almost all of the nodes will receive the block in three (when $M = 16,\ 32,\ 64$) or four waves (when $M = 8$). Compared to the configuration when the relay network is not deployed, there is not much difference for 100% block propagation when $M = 64,\ 32$, but for the cases where $M = 16,\ 8$, the relay network accelerates the 100% block propagation for one wave [see Fig. 16(a)]. According to the results depicted in Fig. 17(b), when reachability is decreased, the long tail of PDF manifests itself even in the presence of the relay network. More reduction in reachability leads to longer tails as shown in Fig. 17(c) and (d).

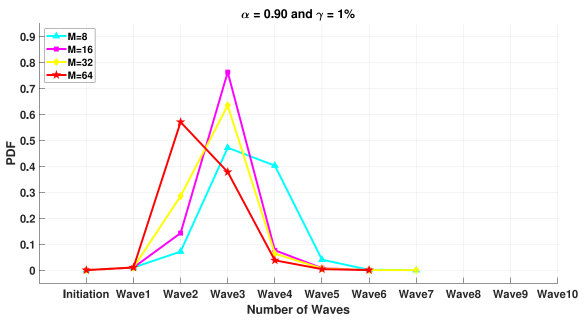
To assess the impact of γ, we conduct another experiment where γ is increased to 2.5%. The results are depicted in Fig. 18. Fig. 18(a) shows the results for the ideal conditions where $\gamma = 2.5\%$. It takes two waves for 100% block propagation when $M = 32$ and 64. Also, 100% block propagation needs three propagation waves when $M = 8$ and 16. According to these observations, we can claim that the network has a noticeable improvement over the case where $\gamma = 1\%$. Note that although



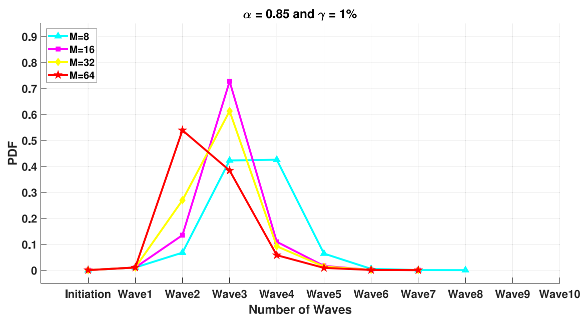
(a)



(b)

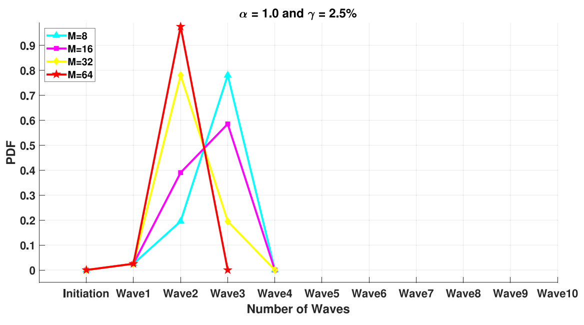


(c)

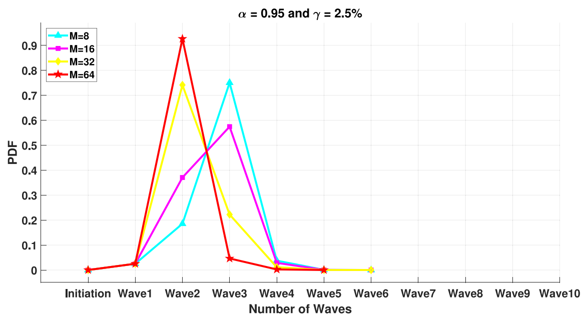


(d)

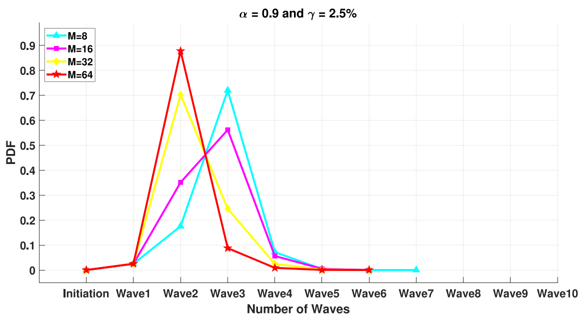
Fig. 17. PDF for waves at which Bitcoin nodes receive the block with a 1% relay network. (a) $\alpha = 1.0$ and $\gamma = 1\%$. (b) $\alpha = 0.95$ and $\gamma = 1\%$. (c) $\alpha = 0.9$ and $\gamma = 1\%$. (d) $\alpha = 0.85$ and $\gamma = 1\%$.



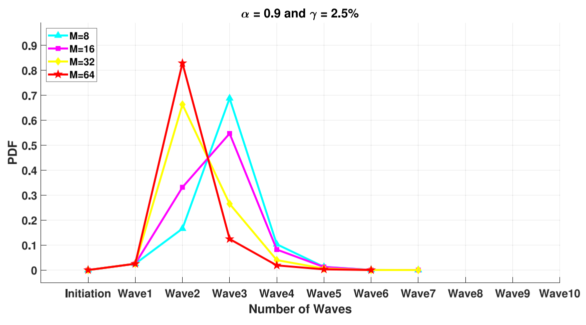
(a)



(b)



(c)



(d)

Fig. 18. PDF for waves at which Bitcoin nodes receive the block with a 2.5% relay network. (a) $\alpha = 1.0$ and $\gamma = 2.5\%$. (b) $\alpha = 0.95$ and $\gamma = 2.5\%$. (c) $\alpha = 0.9$ and $\gamma = 2.5\%$. (d) $\alpha = 0.85$ and $\gamma = 2.5\%$.

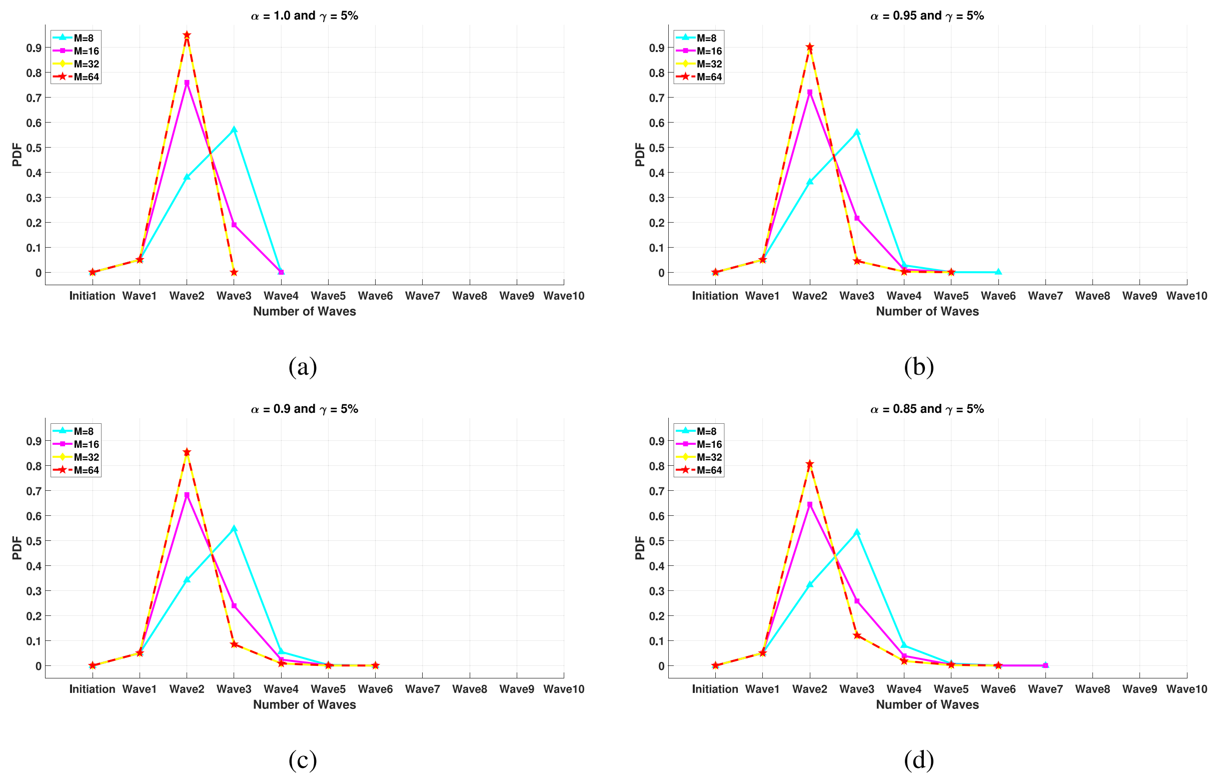


Fig. 19. PDF for waves at which Bitcoin nodes receive the block with a 5% relay network. (a) $\alpha = 1.0$ and $\gamma = 5\%$. (b) $\alpha = 0.95$ and $\gamma = 5\%$. (c) $\alpha = 0.9$ and $\gamma = 5\%$. (d) $\alpha = 0.85$ and $\gamma = 5\%$.

for $M = 16$, the 100% block propagation takes place in three waves, but when $\gamma = 2.5\%$, the block coverage is a bit more than 45% in the second wave when it is around 15% when $\gamma = 1\%$. Also, in Fig. 18(b)–(d), we observe that decreasing the reachability causes the long tails to appear on the PDF again. Another effect of decreasing the reachability factor is that the block coverage decreases in all waves.

If we keep increasing the relay network size, the network will become faster as expected. For $\gamma = 5\%$ and $M = 16$, 32, and 64, the maximum coverage of the block takes place in the second wave as can be seen in Fig. 19. The interesting point is that by increasing the relay network size, the behavior of the network with different values of M become similar. This imply that with a bigger relay network, the default configuration of the nonrelay nodes becomes less important, which may endanger the decentralization in the Bitcoin network.

VI. Managerial Implications

Performance models can provide us an extensive insight of blockchain network dynamics and behaviors. The analytical model proposed in this work contains an approach for predicting and evaluating the performance of Bitcoin-based blockchains (i.e., hard forks of Bitcoin) with controllable input parameters, such as block size, number of HBM and LBM connections, network size, and the network speed.

From the perspective of a system architect, our performance model can help the blockchain developers achieve a good first-cut design, which will meet the application requirements, alleviating the need for trial-and-error tuning of parameters, thus saving on capital expenses. Furthermore, our performance model can be used to conduct *What-if* analyses and anticipate the impact of proposed changes, such as doubling the block size of Bitcoin. The model can therefore inform the manager when taking operational decisions.

From the perspective of a cryptocurrency investment manager, our performance model can be used to compare existing Bitcoin-based cryptocurrencies (also called *altcoins*) by quickly calculating important metrics (such as block propagation delay) using collected measurements from the network. These metrics are indicators of the stability and security of the network, which in turn can be used to assess the viability and volatility of a given cryptocurrency.

VII. Conclusion

In this article, we proposed an analytical model for modeling the delay and traffic overhead in a Bitcoin network based on an Erdös–Renyi random graph and derived key features of performance measures of the Bitcoin network. We validated our analytical model through simulation and compared to real data measured from the Bitcoin network. We also investigated the effect of the default number of connections on the performance of the Bitcoin network. Although the throughput of Bitcoin can be increased by choosing a bigger size for blocks, this can cause a significant increase in the block propagation time. The delay can be reduced by increasing the average default number of connection per node, but this has a drawback of increased traffic overhead in the network.

We used our model to estimate the PDF of the times at which a portion of the Bitcoin nodes received a propagated block at the end of each wave. We adapted our model to analyze the Bitcoin network in the presence of the relay networks. We observe that bigger relay networks (i.e., relay networks with more miners connected to) can significantly improve block propagation delay, and make the network less sensitive to the parameters of nonrelay nodes, which can endanger the decentralization of the network.

REFERENCES

[1] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ, USA: Princeton Univ. Press, 2016.

[2] S. Wilkinson *et al.*, “Storj a peer-to-peer cloud storage network,” 2016.

[3] A. Dorri, S. S. Kanhere, and R. Jurdak, “Blockchain in internet of things: Challenges and solutions,” 2016, *arXiv:1608.05187*.

[4] H. F. Atlam, A. Alenezi, M. O. Alassafi, and G. Wills, “Blockchain with internet of things: Benefits, challenges, and future directions,” *Int. J. Intell. Syst. Appl.*, vol. 10, no. 6, pp. 40–48, 2018.

[5] A. Banafa, “Iot and blockchain convergence: Benefits and challenges,” *IEEE Internet Things*, 2017.

[6] W. J. Gordon and C. Catalini, “Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability,” *Comput. Struct. Biotechnol. J.*, vol. 16, pp. 224–230, 2018.

[7] K. Rabah, “Challenges & opportunities for blockchain powered healthcare systems: A review,” *Mara Res. J. Med. Health Sci.*, vol. 1, no. 1, pp. 45–52, 2017.

[8] M. Sajjad *et al.*, “Raspberry pi assisted face recognition framework for enhanced law-enforcement services in smart cities,” *Future Gener. Comput. Syst.*, vo. 108, pp. 995–1007, Jul. 2020.

[9] K. E. Levy, “Book-smart, not street-smart: blockchain-based smart contracts and the social workings of law,” *Engaging Sci., Technol. Soc.*, vol. 3, pp. 1–15, 2017.

[10] C. M. Christopher, “The bridging model: Exploring the roles of trust and enforcement in banking, bitcoin, and the blockchain,” *Nevada Law J*, vol. 17, pp. 139–180, 2016.

[11] M. Vukolić, “The quest for scalable blockchain fabric: Proof-of-Work vs. BFT replication,” in *Proc. Int. Workshop Open Problems Netw. Secur.*, 2015, pp. 112–125.

[12] K. Zhang and H.-A. Jacobsen, “Towards dependable, scalable, and pervasive distributed ledgers with blockchains,” in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2018, pp. 1337–1346.

[13] X. Xu *et al.*, “A taxonomy of blockchain-based systems for architecture design,” in *Proc. IEEE Int. Conf. Softw. Architecture*, 2017, pp. 243–252.

[14] W. Hao *et al.*, “Blockp2p: Enabling fast blockchain broadcast with scalable peer-to-peer network topology,” in *Proc. Int. Conf. Green, Pervasive, Cloud Comput.*, 2019, pp. 223–237.

[15] G. Owenson, G. Owenson and M. Adda, “Proximity awareness approach to enhance propagation delay on the bitcoin peer-to-peer network,” in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst.*, 2017, pp. 2411–2416.

[16] M. Coralo, “Bitcoin FIBRE network.” [Online]. Available: http://bitcoinfibre.org/public-network.html, Accessed: Jun. 16, 2019.

[17] S. Basu, I. Eyal, and E. G. Sirer, “Falcon,” [Online]. Available: https://www.falcon-net.org, Accessed: May 26, 2019.

[18] U. Klarman, S. Basu, A. Kuzmanovic, and E. G. Sirer, “bloXroute: A scalable trustless blockchain distribution network,” Bloxroute Labs, Whitepaper, ver. 1.0, Mar. 2018

[19] M. Corallo, “Bip 152: Compact block relay,” 2016. [Online]. Available: https://github. com/bitcoin/bips/blob/master/bip-0152.mediawiki

[20] A. P. Ozisik, G. Andresen, G. Bissias, A. Houmansadr, and B. Levine, “Graphene: A new protocol for block propagation using set reconciliation,” in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*. New York, NY, USA: Springer, 2017, pp. 420–428.

[21] Y. Shahsavari, K. Zhang and K. Zhang “Performance modeling and analysis of the bitcoin inventory protocol,” in *Proc. IEEE Int. Conf. Decentralized Appl. Infrastruct. (DAPPCON)*, 2019, pp. 79–88.

[22] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2009. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[23] P. Erdős, and A. Rényi, “On random graphs i.” *Publicationes Mathematicae (Debrecen)*, vol. 6, pp. 290–297, 1959 1959.

[24] T. Neudecker, P. Andelfinger, and H. Hartenstein, “Timing analysis for inferring the topology of the bitcoin peer-to-peer network,” in *Proc. Int. IEEE Conf. Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People, Smart World Congr.*, 2016, pp. 358–367.

[25] J. Misic, V. B. Misic, X. Chang, S. G. Motlagh, and Z. M. Ali, “Modeling of Bitcoin’s blockchain delivery network,” *IEEE Trans. Netw. Sci. Eng.*, 2019.

[26] R. Nagayama, K. Shudo, and R. Banno, “Simulation of the bitcoin network considering compact block relay and internet improvements,” 2019, *arXiv:1912.05208*.

[27] J. A. D. Donet, C. Pérez-Sola, and J. Herrera-Joancomartí, “The bitcoin p2p network,” in *Proc. Int. Conf. Financ. Cryptography Data Secur.*, 2014, pp. 87–102.

[28] C. Decker and R. Wattenhofer, “Information propagation in the bitcoin network,” in *Proc. IEEE Int. Conf. Peer-to-Peer Comput.*, 2013, pp. 1–10.

[29] R. Yasaweerasinghelage, M. Staples, and I. Weber, “Predicting latency of blockchain-based systems using architectural modelling and simulation,” in *Proc. IEEE Int. Conf. Softw. Archit.*, 2017, pp. 253–256.

[30] L. Stoykov, K. Zhang, and H.-A. Jacobsen, “Vibes: fast blockchain simulations for large-scale peer-to-peer networks: Demo,” in *Proc. 18th ACM/IFIP/USENIX Middleware Conf., Posters Demos*, 2017, pp. 19–20.

[31] M. Fadhil, G. Owenson, and M. Adda, “A bitcoin model for evaluation of clustering to improve propagation delay in bitcoin network,” in *Proc. IEEE Int. Conf. Comput. Sci. Eng. IEEE Int. Conf. Embedded Ubiquitous Comput. 15th Int. Symp. Distrib. Comput. Appl. Bus. Eng.*, 2016, pp. 468–475.

[32] Q. Nasir, I. A. Qasse, M. A. Talib, and A. B. Nassif, “Performance analysis of hyperledger fabric platforms,” *Secur. Commun. Netw.*, vol. 2018, 2018, Art. no. 3976093.

[33] P. Thakkar, S. Nathan, and B. Vishwanathan, “Performance benchmarking and optimizing hyperledger fabric blockchain platform,” in *Proc. IEEE 26th Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst. (MASCOTS)*, 2018, pp.264–276.

[34] C. Stathakopoulou, C. Decker, and R. Wattenhofer, “A faster bitcoin network,” Semester thesis, Distrib. Comput. Group Comput. Eng. Networks Lab., ETH, Zürich, Zurich, Switzerland, 2015.

[35] M. Corallo, “Bitcoin Relay Network.” [Online]. Available: http://bitcoinrelaynetwork.org, Accessed: Jun. 16, 2019.

[36] “RFC6363,” [Online]. Available: https://tools.ietf.org/html/rfc6363, Accessed: May 10, 2019.

[37] A. E. Gencer, S. Basu, I. Eyal, R. van Renesse, and E. G. Sirer, “Decentralization in bitcoin and ethereum networks,” in *Proc. Int. Conf. Financial Cryptography Data Secur.*, 2018, pp. 439–457.

[38] R. Kumar and K. W. Ross, “Optimal peer-assisted file distribution: Single and multi-class problems,” in *Proc. IEEE Workshop Hot Topics Web Syst. Technol.*, 2006.

[39] J. Mundinger, R. Weber, and G. Weiss, “Analysis of peer-to-peer file dissemination,” *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 34, no. 3, pp. 12–14, 2006.

[40] K. Oikonomou and I. Stavrakakis, “Performance analysis of probabilistic flooding using random graphs,” in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw.*, 2007, pp. 1–6.

[41] A. K. Hartmann and M. Mézard, “Distribution of diameters for erdős-rényi random graphs,” *Phys. Rev. E*, vol. 97, no. 3, 2018, Art. no. 032128.

[42] V. Buterin *et al.*, “Ethereum White Paper, 2014,” 2013. [Online]. Available: https://github.com/ethereum/wiki/wiki/White-Paper

[43] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, “Bitcoin-ng: A scalable blockchain protocol,” in *Proc. Usenix Conf. Netw. Syst. Des. Implementation*, 2016, pp. 45–59.

[44] M. A. Imtiaz, D. Starobinski, A. Trachtenberg, and N. Younis, “Churn in the bitcoin network: Characterization and impact,” in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency*, 2019, pp. 431–439.

[45] M.-J. Lin and K. Marzullo, “Directional gossip: Gossip in a wide area network,” in *Proc. Eur. Dependable Comput. Conf.*, 1999, pp. 364–379.

[46] R. Pass, L. Seeman, and A. Shelat, “Analysis of the blockchain protocol in asynchronous networks,” in *Proc. Annu. Int. Conf. Theory Appl. Cryptogr. Techn.*, 2017, pp. 643–673.

[47] “Omnet++ simulator,” [Online]. Available: http://www.omnetpp.org

[48] “Bitnodes API v1.0,” [Online]. Available: https://bitnodes.earn.com, accessed: Nov. 1, 2018.

[49] A. Biryukov, D. Khovratovich, and I. Pustogarov, “Deanonymisation of clients in bitcoin p2p network,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur*, 2014, pp. 15–29.

Yahya Shahsavari received the B.Sc. degree in electrical engineering from the University of Zanjan, Zanjan, Iran, in 2008, and the M.Sc. degree in information and communication technology from the Iran University of Science and Technology, Tehran, Iran, in 2016. He is currently working toward the Ph.D. degree with the FUSÉE Laboratory (a pioneer laboratory dedicated to blockchain systems), École de Technologie Supérieure, University of Quebec, Montreal, QC, Canada.

His current main research interests include network performance modeling (particularly blockchain networks) and traffic engineering.

Kaiwen Zhang (Member, IEEE) received the B.Sc. and M.Sc. degrees from McGill University, Montreal, QC, Canada, in 2008 and 2010, respectively, and the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 2015, all in computer science.

He is currently a Professor with the Department of Software and IT Engineering, École de Technologie Supérieure, University of Quebec, Montreal, QC, Canada. He was an Alexander von Humboldt Postdoctoral Fellow in computer science with the Technical University of Munich, Munich, Germany. His current research interests include blockchain technologies, publish/subscribe systems, massive multiplayer online games, and performance modeling.

Chamseddine Talhi received the Ph.D. degree in computer science from Laval University, Quebec, QC, Canada, in 2007.

He is currently an Associate Professor with the Department of Software Engineering and IT, École de Technologie Supérieure, University of Quebec, Montreal, QC, Canada. He is leading a research group that investigates efficient security mechanisms for smartphone, IoT, and edge and cloud infrastructures. His current research interests include cloud native telco services, DevOps security, and federated learning for mobile cloud and IoT.