Modeling of Bitcoin's blockchain delivery network

Jelena Mišić[1], Vojislav B. Mišić[1], Xiaolin Chang[2], Saeideh G. Motlagh[1], and M. Zulfiker Ali[1]
[1]Ryerson University, Toronto, ON, Canada
[2]Beijing Key Laboratory of Security and Privacy in Intelligent Transportation
Beijing Jiaotong University, Beijing, China

Abstract—**In this work we provide a comprehensive analytical model for Bitcoin's blockchain distribution network. Components of the model are derived from recent measurements and business analysis reports. We model the data distribution algorithm using branching processes in the network with random distribution of node connectivity. Then we apply Jackson network model to the entire network in which individual nodes operate as priority M/G/1 queuing systems. Data arrival to the nodes is modeled as a non-homogeneous Poisson process where the distribution of arrival rate to the nodes is derived from the analytical model of data delivery protocol. Within performance results we present probability distributions of block and transaction distribution time, node response time, forking probabilities, network partition sizes and duration of ledger's inconsistency period.**

Index Terms—**Bitcoin, blockchain P2P network, performance analysis**

I. Introduction

Bitcoin, hereafter referred to as BTC, is a decentralized cryptocurrency system running on a number of processing nodes interconnected through a P2P network [22]. It is based on blockchain technology which implements a distributed or, rather, replicated ledger holding financial transactions [3], [4], [20], [29] packaged into blocks. Transactions document transfer of funds from payer(s) to payee(s) and have to be verified to make sure the value of all claimed inputs is equal to or larger than the sum of new outputs [25]. Transactions are grouped into blocks that are 'sealed' by a nonce in the block header which guarantees that the block hash contains the required number of leading zeros. This step requires non-trivial computing effort known as proof of work, or PoW. Each block also contains the hash of the previous block, thus creating a single linked list of blocks held at each processing node.

Newly created or 'mined' block is distributed to the network for further verification; block miners are rewarded in order to incentivize the mining efforts. Upon receiving a new block, nodes which were in process of verifying the same (or nearly the same) set of transactions will abandon their efforts, verify the newly arrived block and insert it into their ledgers, and continue transaction verification from the pool of unverified transactions. To regulate the rate of block mining, the Bitcoin protocol periodically adjusts the difficulty of computing the nonce in the block header so that new blocks are mined once every 10 minutes on the average [22].

Long delays in forwarding blocks or transactions lead to security vulnerabilities [5], [11], [12], [17], [23], [24], [27] which affect ledger consistency. One of the important vulnerabilities is forking which occurs when some nodes append one block to their ledger while the others append another one, leaving the distributed ledger in an inconsistent state. Forking can be exploited for an attack if the attacker can generate blocks at a sufficiently high rate and, thus, build its own blockchain extension which will take over blockchain head. This situation can be mitigated with fast data propagation in the network.

In this work we develop a detailed analytical model of the Bitcoin P2P network based on analysis of Bitcoin's business structure over the P2P network [20], [29] and recent measurement reports regarding connectivity, round trip time (RTT), block size and distribution of block mining power in the network [1], [6], [8], [21], [28]. Analytical modeling of the BTC network faces many challenges. For example, variability of the number of connections among the nodes affects data distribution algorithm and introduces variability of data rates coming to nodes. This introduces non-homogenous Poisson process of data arrivals at network nodes which, in turn, affects further queuing analysis of node which needs to be done in the M/G/1 manner due to unknown properties of service time [30]. Another problem is that block and transaction traffic need to be separated in different queues and handled in priority order. Solving all these issues allows analysis of the blockchain network using the Jackson network approach [19].

Our model includes node connectivity, RTT probability distribution, block size, the impact of non-uniform distribution block miners, data distribution (gossip) algorithm, queuing model at individual nodes and overall network queuing model. The model is then used to find probability distributions of the number of delivery hops, population of nodes receiving data in each hop, and block and transaction delivery times in the blockchain network. Furthermore, we derive phases of forking probability, sizes of network partitions upon a forking event, and duration of the ledger inconsistency period, all of which can help analyze consensus protocols.

The rest of the paper is organized as follows: Section II discusses related work while Section III presents analytical models of connectivity, RTT and block size based on measurements reported in literature. In Section IV, we develop a model of data distribution protocol. Model of incoming data arrival process is given in Section V. In Section VI, we present a queuing model of the distribution network and derive the total distribution time for blocks and transactions in the network. Derivation of forking probability and duration of inconsistency period is presented in Section VII followed by performance evaluation in Section VIII. Finally, Section IX concludes the paper.

II. Related work

Related work can be grouped into papers that report measurements in the BTC network, papers which report business statistics of BTC, and papers which deal with data distribution algorithms in BTC.

a) Measurement reports on BTC network: A number of reports related to measurements of various aspects of the Bitcoin network have appeared in past several years. Combinations of data regarding the BTC network size, geographical distribution of nodes, number of connections per node, and block and transaction statistics were reported in [1], [6], [8], [21], [28]. Measurements of round trip times (RTT), coupled with node distribution in BTC network with vantage point in Vienna, Austria, were reported in [1]; similar measurements from a purpose-built network were presented in [10]. It is worth noting that all reported works used measurement tools which had interfered to some extent with genuine activities in the BTC network.

b) Impact of business profile on BTC network: Another group of works revealed business profile of BTC network [20], [29] which we believe to be the primary driver for the BTC network architecture. They showed that Bitcoin is used by heterogenous businesses that differ by the frequency and value of financial transactions. According to the number of transactions, businesses may be categorized into gambling, mining, exchanges, wallets, programming&hardware services, media news, online vendors, over-the-counter trades, donations, Bitcoin services etc. According to the value of transactions, we can distinguish between exchanges, vendors, wallets, mining, and gambling [20]. Geographical distribution of businesses evolved over the past several years. For example in 2015 majority of transactions originated from US with emphasis on gambling business and followed by Western European countries with emphasis on mining. Recently, the number of BTC nodes in Western Europe has exceeded the number of nodes in US but the intensity of block mining has increased in China where around 70% of all blocks were mined [1], [10].

c) Data distribution algorithms: One of the first and most influential works in measuring performance of data distribution algorithms for BTC network was reported in [5]. Results suggest that block distribution time is exponentially distributed and that forking probability was about 1.8% under 100kbps of TCP connection throughput but with unclear values of node connectivity, RTT distribution, and network size.

Subsequent literature about data distribution paradigm does not provide a completely consistent picture. For example, [27] considers broadcast/flooding only, [5], [7] use a gossip (i.e., controlled flooding) based algorithm, and [24] considers both kinds of data distribution protocols. However, pure broadcast protocol tends to be non-scalable to large networks under ever increasing transaction traffic. Our data distribution algorithm resembles randomized rumor spreading [18], but that work assumed full connectivity and random choice of peers for each transmission by the source. In this work, the source node has peers which are chosen randomly before the TCP connections used for subsequent data distribution were established.

Regarding block arrival process, it is widely assumed that time to complete PoW of new blocks is exponentially distributed, i.e., that blocks arrive to the network according to a homogeneous Poisson process [5], [9], [13], [22], [27]. However, [2] indicates that non-homogenous Poisson process might be a better match due to periodic adjustment of the difficulty to compute the nonce in the block header. Block propagation time is generally assumed to follow exponential probability distribution [5], [27].

The importance of understanding the network layer of the Bitcoin blockchain in order to prevent security attacks is discussed in [7], [24]. Connectivity, join/leave procedures, and communication strategy were mentioned as well, although without qualitative or quantitative evaluation.

III. Node connectivity, RTT and block size

Topology of the BTC network is influenced by the type of business and, to some extent, geographical/national factors. It is well known that each BTC node can connect to 8 outbound peers (nodes) and up to 125 inbound peers. While this seems to imply that the links are unidirectional, it is just an unfortunate choice of terms, as Bitcoin technical community information explicitly states that outbound peers are 'nodes that our node goes out and finds' while inbound peers refer to 'nodes that find us through the network'[1]. This is further confirmed by the fact that both types of connections use TCP transport protocol which is bidirectional by default and that Bitcoin protocol includes message transmission in both directions, as explained in Section IV below.

We hypothesize that this limitation has a rather interesting, although perhaps unintended, consequence: namely, functional partitioning into a highly connected core and lightly connected edge. Namely, some of the BTC peers that have a small number of connections – say, up to 8 or so – function as edge servers that connect small businesses with the network core. These business edge units (i.e., intra-business nodes) are connected to the network core consisting of gateway nodes – either general-purpose or business-specific ones – with much higher connectivity that ensures fast propagation of transactions and blocks throughout the network and, consequently, serves to support efficient operation of Bitcoin distributed ledger. This is the case with block mining, exchanges, gambling, and other highly centralized businesses with high financial incentives [21], which are likely to have nodes with high connectivity which effectively result in network communities with high clustering coefficient and modularity. It is also possible (and economically justified) to have gateways set up by businesses to share their connections between inter- and intra-business interconnections. This type of sharing clearly reflects the social network aspect of the BTC network [20].

The presence of partitioning seems to be confirmed by empirical research that has found both edge nodes with connectivity in the range from 5 to 13, and gateway nodes with connectivity in the range above 14, with the highest value around 60, even though the rules allow up to 125

[1] https://en.bitcoin.it/wiki/Bitcoin_Core_0.11_(ch_4):_P2P_Network, last accessed February 4, 2019

connections [6], [20]. An even larger number of connections was reported in [21], but that observation (which no other report has confirmed) might reflect the use of concurrent TCP connections to increase the throughput, much like modern browsers do. However, the increase in throughput is limited by the available bandwidth of the node [10]; also, due to the inherent fairness of TCP, the throughput per connection will drop.

A. Modeling node connectivity

Let us now present the analytical sub-models that will be used as building blocks of data delivery algorithm in the BTC network.

Number of TCP connections of gateway BTC nodes follows a scale-free long-tail distribution with parameter α in the range 2 to 2.4, depending on the type of business type and country [20]. Node degrees in the long-tail part of the distribution range between 14 and 60. This distribution can be characterized with a probability generating function (PGF) of

$$Lt(z) = L \sum_{k=14}^{60} \frac{1}{k^{\alpha}} z^k \tag{1}$$

and with the scaling factor $L = 1/\sum_{k=14}^{60} \frac{1}{k^{\alpha}}$. The other group of nodes have degrees in the range between 5 and 13 connections with a mean of about 8 [6], [21], [28]. For non-gateway nodes, number of connections for each node follows a truncated binomial probability distribution $Cn(z) = \sum_{k=5}^{13} p_k z^k$ which ranges between 5 and 13 connections with a mean of 8, i.e., $Cn(1) = 1$ and $Cn'(1) = 8$. Following [6], [20], [21], [29] we assume that final connectivity distribution can be obtained as

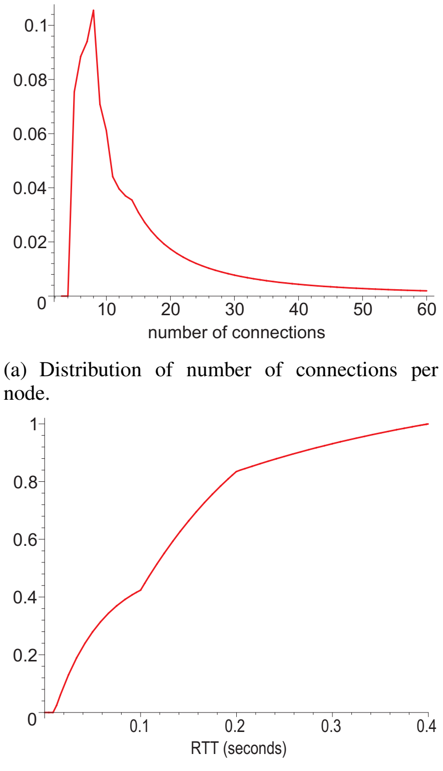
$$Mx(z) = k_m \cdot Lt(z) + (1 - k_m)Cn(z) = \sum_{i=m_{min}}^{i=m_{max}} mx_i z^i \tag{2}$$

where $m_{min} = 5$ and $m_{max} = 60$. Value of weight coefficient k_m can be in the range 0.3 to 0.7, with a note that it affects network diameter. Resulting probability density function (pdf) under $\alpha = 2$ and $k_m = 0.4$, shown in Fig. 1(a), closely resembles measured data from [6].

B. Modeling Round Trip Time (RTT)

According to [1], around 50% of IP addresses reside in EU, 30% reside in North America and 20% reside in Asia; similar results were reported in [10]. As our vantage point is in North American continent, we can create total probability distribution for RTT by combining per-continent segments with exponential distribution, as follows:

1) within North America, RTT has a range of 0 to 100ms with rate $\mu_{NA} = 1/0.05$ so that its pdf in that region is $\mu_{NA} e^{-\mu_{NA}(x-0.01)} H(0.1-x)H(x-0.01)$, where $H(x)$ denotes piecewise Heaviside function defined as 1 when $x \geq 0$ and 0 when $x < 0$.
2) RTT between North America and EU has the range between 100 and 200ms, with a rate of $\mu_{EU} = 1/0.15$, so that its pdf in that region can be expressed as $\mu_{EU} e^{-\mu_{EU}(x-0.1)} H(0.2-x)H(x-0.1)$.



(a) Distribution of number of connections per node.

(b) CDF for the RTT.

Fig. 1. Regarding node connectivity.

3) RTT between North America and Asia has the range between 200 and 400ms with a rate of $\mu_{AS} = 1/0.3$, and its pdf in that region can be expressed as $\mu_{AS} e^{-\mu_{AS}(x-0.2)} H(0.4-x)H(x-0.2)$.

Thus the joint pdf of RTT has three distinct components:

$$\begin{aligned} f_{RTT}(x) = C_f \Big(& 0.3\mu_{NA} e^{-\mu_{NA}(x-0.01)} H(0.1-x)H(x-0.01) \\ & + 0.5\mu_{EU} e^{-\mu_{EU}(x-0.1)} 0.2H(0.2-x)H(x-0.1) \\ & + 0.2\mu_{AS} e^{-\mu_{AS}(x-0.2)} H(0.4-x)H(x-0.2) \Big) \end{aligned} \tag{3}$$

where C_f is derived from the condition of total probability as

$$\begin{aligned} C_f = \Bigg(\int_{x=0.01}^{0.4} & \Big(0.3\mu_{NA} e^{-\mu_{NA}(x-0.01)} H(0.1-x)H(x-0.01) \\ & + 0.5\mu_{EU} e^{-\mu_{EU}(x-0.1)} 0.2H(0.2-x)H(x-0.1) \\ & + 0.2\mu_{AS} e^{-\mu_{AS}(x-0.2)} H(0.4-x)H(x-0.2) \Big) dx \Bigg)^{-1} \end{aligned} \tag{4}$$

However, block and transaction distribution protocol requires one RTT and a single one-way propagation time as shown in Fig. 2. If we assume high correlation of propagation times in both directions, then block/transaction distribution protocol requires 1.5RTT. Its pdf $f_{1.5RTT}(x)$ can be calculated similar to (3) but using 1.5 threshold values. Laplace-Stieltjes transforms (LST) for single and double RTT can then be computed as

$$L^*_{RTT}(s) = \int_{x=0.01}^{0.4} f_{RTT}(x)e^{-sx}dx$$
$$L^*_{1.5RTT}(s) = \int_{x=0.015}^{0.6} f_{1.5RTT}(x)e^{-sx}dx \qquad (5)$$

Resulting cumulative distribution function (CDF) is shown in Fig. 1(b) and it closely resembles measured data from [1]. Furthermore, moments of this distribution are close to those of an exponential distribution.

C. Modeling block and transaction transmission times

Time to transmit a block or transaction depends on their length. In the period of 13 weeks between February 24 and May 26, 2019, block size has ranged between 0.55MB and 1.1MB according to the information on the tracking sites[2]. Block size distribution can be approximated by

$$B^*_{lk}(\beta) = \frac{e^{-6.8\beta}+2e^{-7.45\beta}+e^{-8.3\beta}+2e^{-8.7\beta}+7e^{-9.05\beta}}{13} \qquad (6)$$

expressed in hundreds of KBytes. Given the TCP throughput of 2Mbps per connection which is compliant with real measurements [10], this results in the distribution of transmission times (in seconds) described with the LST transform of

$$L^*_{Tr}(s) = B^*_{lk}(0.4s) \qquad (7)$$

with mean block size and transmission time of around 850 KBytes and 3.4s, respectively.

Total block arrival rate in the network is $\lambda_{b,tot} = \frac{1}{600}$ since one block is mined every 10min in the entire network with N nodes. To model the existence of mining clusters and to account for the impact of traffic injected through their gateways [6], [10], [21], we assume that all nodes participate in block distribution but only $K_f N$ of them inject mined blocks to the network with a rate of $\lambda_b = \frac{\lambda_{b,tot}}{K_f N}$ where $0 < K_f < 1$.

Mean transaction arrival rate ranges around 4.07 to 4.31 per second according to the information available on the aforementioned tracking sites. We have assumed the total transaction arrival rate per network of $\lambda_{t,tot} = 4.31$ per second. Due to heterogeneity of Bitcoin businesses, it is reasonable to assume that transaction arrival rate is uniform across the network with a rate of $\lambda_t = \lambda_{t,tot}/N$. Transaction verification time has the order of tens μs as well as its transmission time. Therefore transaction processing time mostly depends on one and half RTT and we assume that it follows the probability distribution given by (5).

IV. Data distribution and propagation

We consider BTC network with N nodes where node connectivity has PGF $Mx(z)$ described in (2). Probability that two nodes are connected is $pb = \sum_{i=m_{min}}^{m_{max}} mx_i \frac{i}{N} = \frac{\overline{Mx}}{N}$, and the diameter of the network, $D_{N,Mx(z)}$, depends on the network size and connectivity among the nodes.

Block and transaction delivery protocol follows a two-way handshake over each TCP connection, as shown in Fig. 2. Node which is the source of block or transaction will send an *inv* (inventory) message to its direct neighbors. Node that received *inv* message from neighboring node will send a *getdata* request if it does not have block/transaction being announced, or otherwise ignore the *inv* message. As the result, blocks and transactions propagate through the network in phases (generations).

We model data (i.e., block and transaction) forwarding through the network using branching processes [14]. Propagation protocol has the same behavior for the block that was mined by the particular node or transaction that has arrived to that node as the ingress of the overlay network. In the first phase, source node will act as the root of the spanning tree of the graph and transmit the data over all its TCP connections. Its first hop neighbors comprise the first generation of nodes that has received the data, which they will send out to *their* next-hop neighbors (the second generation), and so on. For clarity, we will assign index $i = 0 \ldots\ D_{N,Mx(z)} - 1$ to each phase of the distribution protocol. The PGF for the number of nodes in the first generation is $H_1(z) = \sum_{i=m_{min,1}}^{m_{max,1}} mx_i z^i = Mx(z)$, where $Mx(z)$ is the connectivity PGF defined in (2). Mean number of nodes in first data distribution phase (generation) is $\overline{H_1} = H'(1)$.

Since the probability distribution of the total number of TCP connections per node is homogeneous over all nodes and TCP connections are bidirectional, then each node is source and destination for the same number of connections per each generation i. However, we need to address the following scenarios.

- In i-th phase of the distribution algorithm, some nodes may be interconnected so they will not transmit data to each other, as they already got it from a node in phase $i-1$.
- Furthermore, some nodes in the current generation i may be connected to the same node in phase $i+1$ which will decrease node population in generation $i+1$. This also decreases the number of TCP connections which a node in generation $i+1$ will use to further distribute the data to nodes in generation $i+2$.

The challenge in modeling the BTC gossip algorithm stems from the fact that connections through which the node has received a data item and connections towards the nodes that already have that item cannot be used for data distribution. Let $Mx_i(z)$, $Loc_i(z)$, and $O_i(z)$ denote the PGFs for the total number of connections available for transmission to a node in generation i, the number of connections towards other nodes in the same generation i, and the number of connections available to transmit data to nodes in generation $i+1$, respectively. For a node in the first generation receiving data from the miner

[2] https://bitinfocharts.com/bitcoin/ and https://www.blockchain.com/en/charts/avg-block-size, last accessed May 26, 2019.

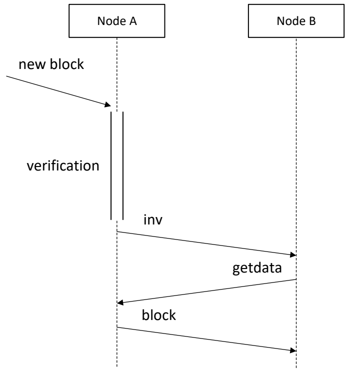


Fig. 2. Data forwarding between neighbors.

(which acts as the root of the spanning tree), these PGFs are

$$Mx_1(z) = Mx(z)/z \tag{8}$$

$$Loc_1(z) = \sum_{i=m_{min,1}}^{m_{max,1}} mx_i \sum_{k=0}^{i-1} \binom{i-1}{k} pb^k z^k (1-pb)^{i-1-k}$$

$$O_1(z) = \sum_{i=m_{min,1}}^{m_{max,1}} mx_i \sum_{k=0}^{i-1} \binom{i-1}{k} pb^k (1-pb)^{i-1-k} z^{i-1-k}$$

Based on the equal number of output and input connections for each phase, second generation node will have the number of input connections described with PGF $O_1(z)$. After excluding local connections in the second generation, the number of connections available to send data towards the third generation nodes is

$$O_2(z) = \sum_{i=m_{min,1}}^{m_{max,1}} mx_i \sum_{k=0}^{i-1} \binom{i-1}{k} pb^k (1-pb)^{i-1-k} \cdot \frac{\displaystyle\sum_{l=1}^{i-1-k} \binom{i-1-k}{l} pb^{i-1-k-l} (1-pb)^l z^l}{\displaystyle\sum_{l=1}^{i-1-k} \binom{i-1-k}{l} pb^{i-1-k-l} (1-pb)^l} \tag{9}$$

As noted above, some of the connections going from the first to the second generation may end at the same node. We can model the number of overlapping connections coming to the second generation node with the PGF of

$$Ot_2(z) = \frac{\displaystyle\sum_{k=1}^{\overline{H_1}} \binom{\overline{H_1}}{k} pb^k (1-pb)^{\overline{H_1}-k} \cdot z^{1/k}}{\displaystyle\sum_{k=1}^{\overline{H_1}} \binom{\overline{H_1}}{k} pb^k (1-pb)^{\overline{H_1}-k}} \tag{10}$$

Therefore, the PGF for the number of TCP connections available to a node in second generation to continue data distribution is

$$Mx_2(z) = O_2(Ot_2(z))) \tag{11}$$

which leads to the PGF for the number of nodes (population) in the second generation of data distribution algorithm as

$$H_2(z) = H_1(Mx_2(z)) \tag{12}$$

Values of $Mx_2(z)$ and $H_2(z)$ are, then, used in a set of equations analogous to (9) to (12) to compute node connectivity and population for the third phase; this is repeated with the next phase, for a total number of $D_{N,Mx(z)} - 1$ phases. In each phase mean node populations in generation i, $i = 0 \ldots D_{N,Mx(z)} - 1$ are calculated as $\overline{H_i} = H_i'(1)$, and probability that a given node is reached in i-th generation is

$$Pt_i = \frac{\overline{H_i}}{N}. \tag{13}$$

Mean number of nodes reached in the last generation and probability that data will not be forwarded can be then computed as

$$\overline{H_{D_{N,Mx(z)}}} = N - \sum_{i=0}^{D_{N,Mx(z)}-1} \overline{H_i}$$

$$Pnt = 1 - \sum_{i=0}^{D_{N,Mx(z)}-1} Pt_i \tag{14}$$

V. Transaction and block rates

The model from Section IV describes network distribution of one data item. However nodes in the network can also inject newly mined blocks for verification distribution, and any node can inject new transactions for verification and distribution. Therefore traffic offered to any network node consists of fresh traffic to undergo further distribution and traffic which is in the process of distribution over the network.

A new block is mined when all transactions within it are valid and Proof of Work (PoW) is completed. Upon hearing the newly mined block, nodes which are currently mining will cancel their mining process. All nodes participate in block distribution but, as mentioned in Section III, only $K_f N, 0 < K_f < 1$, nodes are injecting mined blocks to the network.

Due to finite data distribution time, competing blocks may be mined and distributed through the network by two or more nodes in the time window shorter than block distribution time. This results in a fork, the situation in which different blocks are linked as blockchain heads in different areas of the network. Assuming that all nodes have the same computational power (hashrate), Poisson arrival rate of new blocks per node can be written as

$$\lambda_b = \frac{1}{600 \cdot K_f N}(1 + P_{fork}) \tag{15}$$

where P_{fork} denotes forking probability which will be derived in Section VII below.

We assume that new transactions arrive to each node at a rate of λ_t as stated in Section III. At each node of the network, PGFs for the fresh traffic for blocks and transactions, respectively, entering the network are

$$\lambda_{b,0}(z) = K_f z^{\lambda_b} + (1 - K_f) z^0 \tag{16}$$

$$\lambda_{t,0}(z) = z^{\lambda_t}$$

Let ω_b and ω_t denote output block and transaction rates, respectively, from each node. For the initial phase $i = 0$, input and output rates for block and transaction traffic are the same:

$$\begin{aligned}\lambda_{b,0}(z) &= \omega_{b,0}(z) = K_f z^{\lambda_b} + (1-K_f)z^0 \qquad (17)\\ \lambda_{t,0}(z) &= \omega_{t,0}(z) = z^{\lambda_t}\end{aligned}$$

For each further phase $i = 1 \dots D_{N,Mx(z)} - 1$, since TCP connections are bidirectional, PGFs for the input and output, block and transaction traffic are

$$\begin{aligned}\lambda_{b,i}(z) &= Mx_i(\omega_{b,i-1}(z)) \qquad (18)\\ \lambda_{t,i}(z) &= Mx_i(\omega_{t,i-1}(z))\\ \omega_{b,i}(z) &= \lambda_{b,i}(z)\left(1 - Pnt - \sum_{j=0}^{i-1} Pt_j\right) + Pnt + \sum_{j=0}^{i-1} Pt_j\\ \omega_{t,i}(z) &= \lambda_{t,i}(z)\left(1 - Pnt - \sum_{j=0}^{i-1} Pt_j\right) + Pnt + \sum_{j=0}^{i-1} Pt_j\end{aligned}$$

Since each node distributes the data for all phases total traffic offered to each node can be described with

$$\begin{aligned}\lambda_{b,tot}(z) &= \prod_{j=0}^{D_{N,Mx(z)}-1} \lambda_{b,j}(z) \qquad (19)\\ \lambda_{t,tot}(z) &= \prod_{j=0}^{D_{N,Mx(z)}-1} \lambda_{t,j}(z)\\ \omega_{b,tot}(z) &= \prod_{j=0}^{D_{N,Mx(z)}-1} \omega_{b,j}(z)\\ \omega_{t,tot}(z) &= \prod_{j=0}^{D_{N,Mx(z)}-1} \omega_{t,j}(z)\end{aligned}$$

for blocks and transactions, respectively:

Therefore, input and output data rates from each node are random variables due to the combination of distribution of blocks and transactions in different phases and due to the random connectivity of each node towards the rest of the network. First two central moments of the output data rate distributions for block traffic can be obtained from (19) as $\overline{\omega_{b,tot}} = \frac{d}{dz}\omega_{b,tot}(z)\big|_{z=1}$ and $var(\omega_{b,tot}) = \frac{d^2}{dz^2}\omega_{b,tot}(z)\Big|_{z=1} + \overline{\omega_{b_{tot}}} - \overline{\omega_{b,tot}}^2$; corresponding values for transaction traffic can be obtained in the analogous manner. We do not show expressions for the third and fourth central moments due to space restrictions.

Due to complexity of (19), we have modeled the probability distribution of output data rates for blocks and transactions, respectively, as Gamma distribution with densities of

$$f_b(y) = \frac{1}{\Gamma(c_{\omega,b})} b_{\omega,b}^{c_{\omega,b}} y^{c_{\omega,b}-1} e^{-y/b_{\omega,b}} \qquad (20)$$

$$f_t(y) = \frac{1}{\Gamma(c_{\omega,t})} b_{\omega,t}^{c_{\omega,t}} y^{c_{\omega,t}-1} e^{-y/b_{\omega,t}} \qquad (21)$$

where parameters of block and transaction traffic defined as

$$\begin{aligned}b_{\omega,b} &= var(\omega_{b,tot})/\overline{\omega_{b,tot}} \qquad (22)\\ c_{\omega,b} &= \overline{\omega_{b,tot}}/b_{\omega,b}\\ b_{\omega,t} &= var(\omega_{t,tot})/\overline{\omega_{t,tot}} \qquad (23)\\ c_{\omega,t} &= \overline{\omega_{t,tot}}/b_{\omega,t}\end{aligned}$$

VI. Queuing model of the distribution network

Nodes receive blocks and transactions in different distribution phases over its TCP connections, but they receive and forward only the data they don't already have. Furthermore, blocks and transactions join different pools for verification, with the former given priority over the latter. This framework can be modeled using the Jackson network approach [19] applied to non-preemptive priority queues, as discussed in [15], [16], [26]. This approach is justified as the block arrival rate is much lower than the transaction arrival rate so the adverse effect of the former (which has higher priority) on the latter will not be high. Thus each node in the network has two queues organized in priority order of blocks over transactions. Each queue is fed by external arrivals of mined blocks and new transactions, but the node also can eliminate a block or transaction if they are already in the queue and there is no need for further forwarding. Since data which will not be forwarded do not join forwarding queues, input rate for one node consists of output rates (of all protocol phases) of all nodes connected to it.

A. Node response time for a block

Time to process and forward a single block consists of block verification time, time to exchange *inventory and* getdata messages which consists of one and half mean round trip times (RTT-s), and time to transmit the block. Block verification time has the order of few milliseconds, while one and half RTTs with a cumulative distribution shown in (3) and (5) have a mean value around 0.19s. Probability distribution of the block transmission time over TCP connection is given by (7). The distribution of block processing time can be described with the LST of

$$B_b^*(s) = L_{1.5RTT}^*(s) L_{Tr}^*(s) \qquad (24)$$

Probability density function of block service time $b_b(x)$ can be found using properties of LST [19] as

$$\begin{aligned}b_b(x) = \frac{1}{13}(&f_{1.5RTT}(x-2.72) + 2f_{1.5RTT}(x-2.96)\\ &+ f_{1.5RTT}(x-3.32) + 2f_{1.5RTT}(x-3.48)\\ &+ 7f_{1.5RTT}(x-3.62)) \qquad (25)\end{aligned}$$

Transaction verification and transmission times are much smaller than 1.5RTT which is why the pdf of transaction service time may be approximated with $b_t(x) = f_{1.5RTT}(x)$.

For non-preemptive priority M/G/1 analysis of block and transaction queues [30] we need probability distributions of the number of block arrivals during block service time, the number of block arrivals during transaction service time, and the number of transaction arrivals during transaction service time, the PGFs of which will be denoted with $A_b(z)$, $A_{bt}(z)$, and $A_t(z)$, respectively.

A specific challenge in this model is that arrival rates of Poisson processes for blocks and transactions, $\omega_{b,tot}$ and $\omega_{b,tot}$, are random and the framework from [30] had to be modified. To calculate probability distributions of arrival process during block and transaction service times, we need to derive individual arrival probabilities:

$$a_{b,k} = \int_{y=0}^{\infty}\int_{x=0}^{\infty} \frac{(xy)^k}{k!} e^{-xy} f_b(y) b_b(x) dx dy \tag{26}$$

$$a_{bt,k} = \int_{y=0}^{\infty}\int_{x=0}^{\infty} \frac{(xy)^k}{k!} e^{-xy} f_b(y) b_t(x) dx dy$$

$$a_{t,k} = \int_{y=0}^{\infty}\int_{x=0}^{\infty} \frac{(xy)^k}{k!} e^{-xy} f_t(y) b_t(x) dx dy$$

from which we can derive the corresponding PGFs:

$$\begin{aligned} A_b(z) &= \sum_{k=0}^{\infty} a_{b,k} z^k \\ &= \int_{y=0}^{\infty}\int_{x=0}^{\infty} e^{-xy(1-z)} f_b(y) b_b(x) dx dy \\ &= \int_{y=0}^{\infty} B_b^*(y(1-z)) f_b(y) dy \end{aligned} \tag{27}$$

$$\begin{aligned} A_{bt}(z) &= \sum_{k=0}^{\infty} a_{bt,k} z^k \\ &= \int_{y=0}^{\infty}\int_{x=0}^{\infty} e^{-xy(1-z)} f_b(y) b_t(x) dx dy \\ &= \int_{y=0}^{\infty} B_t^*(y(1-z)) f_b(y) dy \end{aligned} \tag{28}$$

$$\begin{aligned} A_t(z) &= \sum_{k=0}^{\infty} a_{t,k} z^k \\ &= \int_{y=0}^{\infty}\int_{x=0}^{\infty} e^{-xy(1-z)} f_t(y) b_t(x) dx dy \\ &= \int_{y=0}^{\infty} B_t^*(y(1-z)) f_t(y) dy \end{aligned} \tag{29}$$

Computational complexity of Taylor series expansion of the PGF is rather high but we found that about 24 series members for $A_t(z)$ and six series members for $A_b(z), A_{bt}(z)$ provide sufficient accuracy of 10^{-6}, although the exact number depends on the load. Block traffic (higher priority) PGF for the number of blocks left in the queue after a block departure and LST for the block response time have the following form:

$$PP_b(z) = A_b(z) \cdot \tag{30}$$
$$\frac{\overline{\omega_{t,tot}} A_{bt}(z) - (\overline{\omega_{tot}} - \rho \overline{\omega_{b,tot}}) + z(1-\rho_{tot}) \overline{\omega_{b,tot}}}{\overline{\omega_{b,tot}}(z - A_b(z))}$$

$$T_b^*(s) = B_b^*(s) \frac{(1-\rho_{tot})s + \overline{\omega_{t,tot}}(1 - B_t^*(s))}{s - \overline{\omega_{b,tot}} + \overline{\omega_{b,tot}} B_b^*(s)} \tag{31}$$

where $\overline{\omega_{tot}} = \overline{\omega_{b,tot}} + \overline{\omega_{t,tot}}$, $\rho_{b,tot} = \overline{\omega_{b,tot}}/\mu_b$, $\rho_{t,tot} = \overline{\omega_{t,tot}}/\mu_t$ and $\rho_{tot} = \rho_{b,tot} + \rho_{t,tot}$. Then, k-th moment of node response time for blocks can be obtained as

$$\overline{T_b}^{(k)} = (-1)^k \, \frac{d^k}{ds^k} T_b^*(s) \bigg|_{s=0} \tag{32}$$

Coefficient of variation, skewness, and kurtosis for node response time for blocks are calculated as

$$cv(T_b) = \sqrt{(\overline{T_b}^{(2)} - \overline{T_b})}/\overline{T_b} \tag{33}$$

$$sk(T_b) = \frac{\overline{T_b}^{(3)} - 3\overline{T_b}(\overline{T_b}^{(2)} - \overline{T_b}^2) - \overline{T_b}^3}{\left(\sqrt{(\overline{T_b}^{(2)} - \overline{T_b})}\right)^3} \tag{34}$$

$$kt(T_b) = \frac{\overline{T_b}^{(4)} - 4\overline{T_b T_b}^{(3)} + 6\overline{T_b}^{(2)}\overline{T_b}^2 - 3\overline{T_b}^4}{\left(\sqrt{(\overline{T_b}^{(2)} - \overline{T_b})}\right)^4} \tag{35}$$

B. Node response time for transactions

Transactions have lower priority and response time for transactions needs to include the entire busy period for block service. To this end, we need to find PGF $F(z)$ for the number of blocks served in the busy period of the node. This PGF can be found from the difference equation [30] as

$$F(z) = z B_b^*(\overline{\omega_{b,tot}} - \overline{\omega_{b,tot}} F(z)) \tag{36}$$

which can be solved by expanding both sides into power series and equating the corresponding coefficients. The LST for the block busy period can be found as

$$\Phi^*(s) = F(B_b^*(s)) \tag{37}$$

The PGF for the number of transactions left in the queue after transaction departure and LST for the transaction response time can be written as

$$PP_t(z) = \frac{A_t(z)((1-\rho_{tot})\sigma^* + \overline{\omega_{t,tot}}(1 - B_t^*(\sigma^*))}{\overline{\omega_{t,tot}}(B_t^*(\sigma^*) - z)}$$

$$T_t^*(s) = B_b^*(s) \frac{(1-\rho_{tot})\sigma}{s - \overline{\omega_{t,tot}} + \overline{\omega_{t,tot}} B_t^*(\sigma)} \tag{38}$$

where $\sigma = s + \overline{\omega_{b,tot}}(1 - \Phi^*(s))$ and $\sigma^* = (\overline{\omega_{b,tot}} + \overline{\omega_{t,tot}}) - z\overline{\omega_{t,tot}} - \overline{\omega_{b,tot}} \Phi^* \left(\overline{\omega_{t,tot}}(1-z)\right)$. k-th moment of node response time for transactions can be found as $\overline{T_t}^{(k)} = (-1)^k \, \frac{d^k}{ds^k} T_t^*(s) \Big|_{s=0}$.

C. Data distribution time

Using the probabilities of phases in the data distribution protocol calculated in Section IV, we can calculate the LST of the duration of total block/transaction distribution as

$$\Xi_b^*(s) = \sum_{i=0}^{D_{N,Mx(z)}-1} Pt_i (T_b^*(s))^{(i+1)} + Pnt (T_b^*(s))^{(D_{N,Mx(z)}+1)}$$
$$\tag{39}$$
$$\Xi_t^*(s) = \sum_{i=0}^{D_{N,Mx(z)}-1} Pt_i (T_t^*(s))^{(i+1)} + Pnt (T_t^*(s))^{(D_{N,Mx(z)}+1)}$$

from which all necessary moments of distribution time can be found. Due to complexity of (39), it is hard to get probability density functions using inverse transformation. Instead we have estimated data distribution time using Gamma distribution with probability density function as

$$f_{n,b}(x) = \frac{1}{\Gamma(c_{n,b})} b_{n,b}^{c_{n,b}} x^{(c_{n,b}-1)} e^{-x/b_{n,b}} \tag{40}$$

$$f_{n,t}(x) = \frac{1}{\Gamma(c_{n,t})} b_{n,t}^{c_{n,t}} x^{c_{n,t}-1} e^{-x/b_{n,t}}$$

where values $b_{n,b}$, $c_{n,b}$, $b_{n,t}$, $c_{n,t}$ are defined by

$$b_{n,b} = var(\Xi_b)/\overline{\Xi_b}, \quad c_{n,b} = \overline{\Xi_b}/b_{n,b}$$

$$b_{n,t} = var(\Xi_t)/\overline{\Xi_t}, \quad c_{n,t} = \overline{\Xi_t}/b_{n,t} \tag{41}$$

VII. Forking

Normal operation of Bitcoin's distributed ledger assumes that all nodes have the same list of linked blocks, with all ledgers having the same block Y as the head of their respective chains. However it is possible that two or more competing blocks are mined and sent out during the time window when neither of the blocks has completed distribution through the network. The set of block transactions may overlap in part or in full, except for the so-called coinbase transactions that allocates the mining fee to the miner node and, therefore, must be specific to each block [22].

Let us assume that block $\mathcal{A}$ has been mined at time t_0 and is in the process of distribution, and that at time $t_0 < t_1 < \Xi_b$ another node mines block $\mathcal{B}$ and begins distributing it to the network. As the result, some nodes will link block $\mathcal{A}$ as the head, while others will link block $\mathcal{B}$ at the head of their local ledger. This creates two partitions with different heads in the blockchain and creates a fork – an inconsistent state in distributed ledger, as shown in Fig. 3. It is possible to have multiple-way forking but its probability is very low and we will neglect it in further considerations.

Inconsistency is resolved by the arrival of a new block $\mathcal{C}$. In one case, block $\mathcal{C}$ arrives after blocks $\mathcal{A}$ and $\mathcal{B}$ have finished propagation, as shown in the top diagram of Fig. 4, and we need to consider the partition in which the block $\mathcal{C}$ is mined. In the originating partition with, say, $\mathcal{A}$ as the head, block $\mathcal{C}$ will be linked to $\mathcal{A}$ without a problem and become the new main head; in non-originating partition, with $\mathcal{B}$ as the head, block $\mathcal{C}$ will become the main head by virtue of its height (i.e., distance from the beginning of blockchain – the genesis block) exceeding that of the current head while $\mathcal{B}$ will become a side head. Transactions that are unique to the new side head (i.e., not contained in the new head) will be moved back to the transaction pool buffer and considered unconfirmed in nodes from non-originating partition.

in the other case, block $\mathcal{C}$ may be mined while previous block(s) are still in the process of distribution, as shown in the lower diagram of Fig. 4. Nodes that know of $\mathcal{A}$ or $\mathcal{B}$ will process the new block in the manner similar to the case above. Nodes that do not know yet of either $\mathcal{A}$ or $\mathcal{B}$ will treat $\mathcal{C}$ as an orphan block: i.e., they will request its previous block(s) from the transmitting node before processing it and eventually appending it to its blockchain.

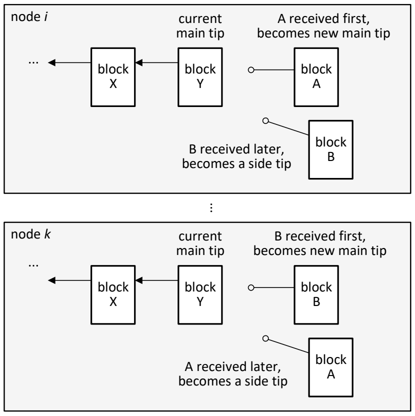

node i
current main tip
A received first, becomes new main tip
...
block X
block Y
block A
B received later, becomes a side tip
block B
⋮
node k
current main tip
B received first, becomes new main tip
...
block X
block Y
block B
A received later, becomes a side tip
block A

Fig. 3. Two-way fork with a main and a side head.

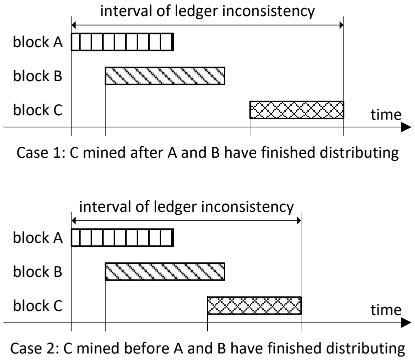

interval of ledger inconsistency
block A
block B
block C
time
Case 1: C mined after A and B have finished distributing
interval of ledger inconsistency
block A
block B
block C
time
Case 2: C mined before A and B have finished distributing

Fig. 4. Inconsistency period for a two-way fork.

A. Forking probability and partition sizes

The information we have about block distribution allows us to develop forking probability for different distribution phases of block $\mathcal{A}$. Let $Pth_i^*(s) = T_b^*(s)^i$ denote the LST of the path delay over $1 \leq i \leq D_{N,Mx(z)}$ hops; the corresponding probability density function can be obtained through appropriate Gamma functions using moments defined in (32):

$$b_{pt,i} = var(Pth_i)/\overline{Pth_i} = \overline{T_b}^{(2)}/\overline{T_b}$$

$$c_{pt,i} = i\overline{T_b}/b_{pt,i}$$

$$f_{pt,i}(x) = \frac{1}{\Gamma(c_{pt,i})} b_{pt,i}^{c_{pt,i}} x^{(c_{pt,i}-1)} e^{-x/b_{pt,i}} \tag{42}$$

Approximation (42) is valid since block traffic has low intensity and has priority over transaction traffic.

We also note that a total of $cv_i = \sum_{j=0}^{i-1} \overline{H_i}$ nodes have already received block $\mathcal{A}$ at the moment when i-th distribution phase of this block begins. For completeness, we assume that $cv_0 = \overline{H_0} = 1$ and $P_{fork,0} = 0$. Then, the probability that block $\mathcal{B}$ appears (i.e., is mined) when block $\mathcal{A}$ is in its i-th distribution phase, is

$$P_{fork,i} = \int_{x=(i-1)\overline{T_b}}^{\infty} (1 - e^{-K_f(N-cv_i)\lambda_b x}) \cdot \frac{1}{\Gamma(c_{pt,i})} b_{pt,i}^{c_{pt,i}} x^{c_{pt,i}-1} e^{-x/b_{pt,i}} dx \tag{43}$$

Mean forking probability over all distribution phases is

$$P_{fork} = \sum_{i=1}^{D_{N,Mx(z)}} Pt_i P_{fork,i} \tag{44}$$

Unfortunately P_{fork} from (44) participates in block arrival rate defined by (15). This creates the need to iteratively solve a system of equations starting from low tentative values, say $P^0_{fork} = 0.001$, until the difference between P_{fork} values computed in two successive iterations drops below a predefined threshold.

Probabilities $P_{fork,i}$ also include information about sizes of forked blockchain partitions since block $\mathcal{A}$ is already linked in generations $0 \ldots i-1$. Also, all nodes from generation H_{i-1} are distributing block $\mathcal{A}$ while only the miner node of block $\mathcal{B}$ begins its distribution. Therefore, the remaining nodes (assuming that each node has connectivity PGF $Mx(z)$) will link blocks $\mathcal{A}$ or $\mathcal{B}$ with probabilities $P_{A,i} = \frac{\overline{H_{i-1}}}{\overline{H_{i-1}}+1}$ and $P_{B,i} = \frac{1}{\overline{H_{i-1}}+1}$, respectively. This further means that mean sizes of partitions A_i and B_i formed when block $\mathcal{B}$ appears while block $\mathcal{A}$ is in its i-th distribution phase, are approximately $A_i = cv_i + (N - cv_i)P_{A,i}$ and $B_i = (N - cv_i)P_{B,i}$.

B. Duration of ledger inconsistency

Period of inconsistency depicted in Fig. 4 begins when block $\mathcal{A}$ is mined, proceeds with mining and distribution of blocks $\mathcal{B}$ and $\mathcal{C}$, and ends when block $\mathcal{C}$ is linked in the whole network. Since block $\mathcal{C}$ arrives at a random point in distribution time of block $\mathcal{B}$, we need the probability distribution of remaining distribution time of block $\mathcal{B}$ after the arrival of block $\mathcal{C}$. Based on LST for the probability distribution of block distribution time from (39), we will calculate the LST for the elapsed and remaining block distribution time as

$$\Xi^*_{b,-}(s) = \Xi^*_{b,+}(s) = \frac{1 - \Xi^*_b(s)}{s\overline{\Xi}} \tag{45}$$

Let $T^*_l(s)$ denote the LST of block linking time as main or side tip; note that it is much smaller than block distribution time. The period of ledger inconsistency due to a fork may be obtained as a weighted sum of durations of inconsistency of scenarios from Fig. 4; it can be expressed as

$$\begin{aligned} T^*_{i,1}(s) &= P_{fork}\Xi^*_{b,-}(s)^2\Xi^*_b(s)T^*_l(s) \\ &\quad + (1 - P_{fork})\Xi^*_{b,-}(s)I^*_b(s)\Xi^*_b(s)T^*_l(s) \end{aligned}$$

It is possible to have multiple consecutive forking events, i.e., that instead of the arrival of a single block that will resolve the fork, two new blocks arrive that build on different partitions and thus prolong the fork. In the general case, the inconsistent state may consist of any number of consecutive forking events which leads us to the general expression for the distribution of inconsistency time of the ledger:

$$\begin{aligned} T^*_i(s) &= (1 - P_{fork})\Xi^*_b(s) + \sum_{k=1}^{\infty}(1 - P_{fork})P^k_{fork}T^*_{i,k}(s) \\ &= (1 - P_{fork})\Xi^*_b(s) \\ &\quad + \frac{(1 - P_{fork})P_{fork}\Xi^*_{b,-}(s)\Xi^*_b(s)T^*_l(s)}{1 - P_{fork}\Xi^*_{b,-}(s)I^*_b(s)} \\ &\quad \cdot \left(P_{fork}\Xi^*_{b,-}(s) + (1 - P_{fork})I^*_b(s)\right) \end{aligned} \tag{46}$$

where $T^*_{i,k}(s) = (1 - P_{fork})\Xi^*_{b,-}(s)^k I^*_b(s)^k \Xi^*_b(s)T^*_l(s) + P_{fork}I^*_b(s)^{(k-1)}\Xi^*_{b,-}(s)^{(k+1)}\Xi^*_b(s)T^*_l(s)$.

Note that the first term in (46) corresponds to the time for a block to propagate through the network and update the blockchain 'regularly,' while the second describes the time required to resolve a fork. Hence the last equation effectively describes the time needed to reach the consensus in the Bitcoin network.

VIII. Performance results

Our evaluation was conducted for BTC network size varied from $N = 2500$ to 5000 in steps of 250. Connectivity of each node was modeled using PGF $Mx(z)$ defined in (2) with $k_m = 0.4$ while the long-tail parameter in (1) was set to $\alpha = 2$ in order to match empirical results from [6]. Consequently, mean number of connections per node was around $Mx'(1) \approx 15.4$ and network diameter was $D_{N,Mx(z)} = 4$ in the range of network sizes under investigation.

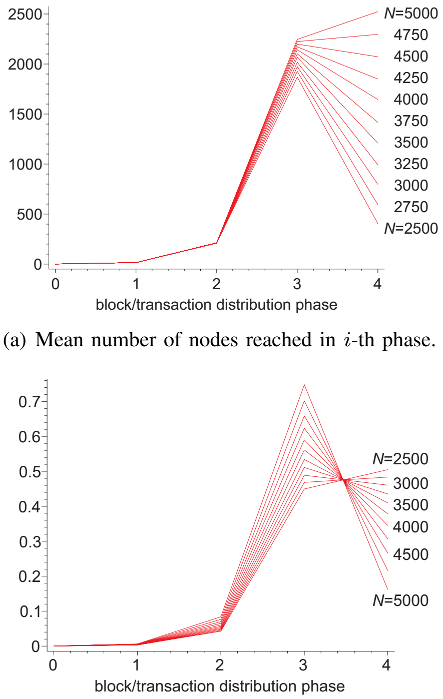
Portion of nodes that are injecting newly mined blocks into the network was set to $K_f = 1$ and $K_f = 0.5$, respectively. Total new transaction arrival rate per network is $\lambda_{t,tot} = 4.31$ per second, and $\lambda_t = \lambda_{t,tot}/N$ per node.

A. Node connectivity and data distribution

Mean number of nodes reached in each phase is shown in Fig. 5(a), where the lowest line corresponds to $N = 2500$ while the highest one corresponds to $N = 5000$; lines between these are monotonically ordered. All lines show exponential increase of mean population in each generation until the third, at which point the gradient towards the fourth generation becomes negative. Probability of a node belonging to i-th generation, Fig. 5(b), is obtained by dividing mean size of a generation size by network size. As the result, probability that a node belongs to the fourth generation is highest when $N = 2500$ and lowest when $N = 5000$, since network size is increasing but the connectivity distribution $Mx(z)$ does not change.

B. Data arrival rates

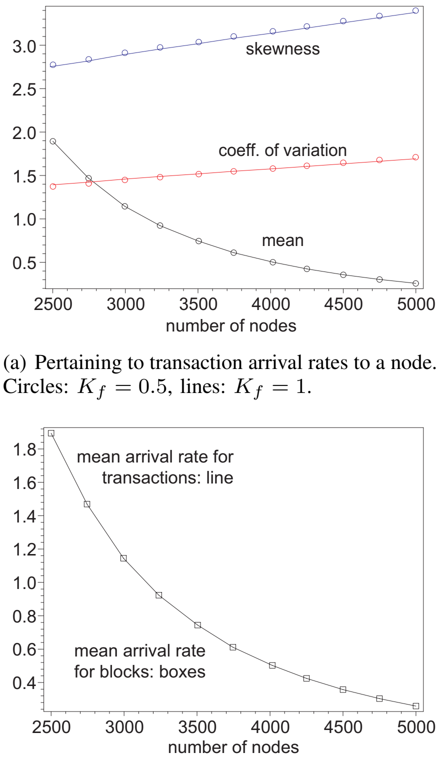
Total transaction arrival rates to a node are shown in FIg. 6(a), with circles and lines corresponding to results for $K_f = 0.5$ and $K_f = 1$, respectively. As can be seen, the mean value, coefficient of variation, and skewness are virtually identical for both values of K_f. (Similar observation applies to kurtosis, except that its value is in the range of 12 to 18, which is why it is not shown here.) Mean arrival rate decreases with



(a) Mean number of nodes reached in i-th phase.

(b) Probability that a node is reached in i-th phase.

Fig. 5. Data distribution in the network.



(a) Pertaining to transaction arrival rates to a node. Circles: $K_f = 0.5$, lines: $K_f = 1$.

(b) Mean arrival rates for transactions (line) and blocks (boxes) per node.

Fig. 6. On data arrival rates.

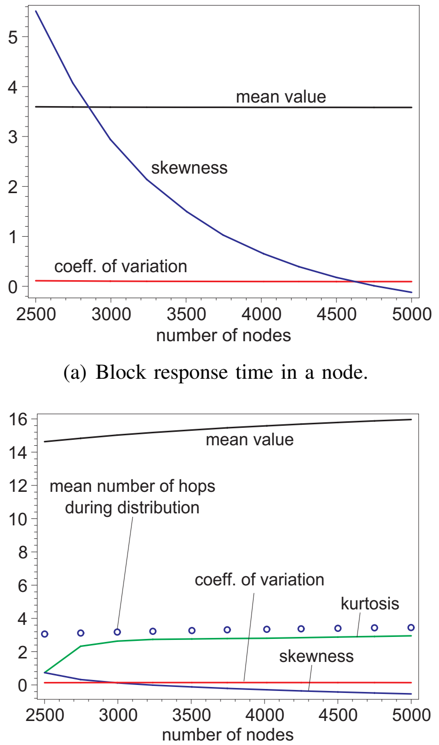
network size while both coefficient of variation and skewness show a mild linear increase. This is due to large size of the network and rich connectivity (but with homogeneous distribution) between nodes, which means that most of the traffic coming to a node is actually data relayed from other nodes, rather than data generated locally. On account of this, in the discussions that follow we will show only the results for $K_f = 1$.

Another important observation is that block and transaction arrival rates behave in the same manner, as can be seen from Fig. 6(b) which shows mean transaction arrival rate to a node (line) as well as mean block arrival rate (boxes) but scaled to the value of mean transaction arrival rate at $N = 2500$ nodes. This stems from the fact that data propagation through the network follows the same pattern regardless of the type of data, i.e., is it a block or a transaction. Mean arrival rates decrease with network size due to the higher volume of relaying traffic to that generated locally, as the total transaction arrival rate to the network is kept at a constant value.

C. Queuing performance of block traffic

Results of queuing analysis for block traffic are shown in Fig. 7 for $K_f = 1$; results for $K_f = 0.5$ are virtually identical, as explained above. Fig. 7(a) describes node response time for blocks. Mean value is very close to the sum of mean block transmission time, 1.5RTT, and small waiting time in the queue. It shows very small decline with network size due to decrease of the waiting time in the queue. Coefficient of variation changes is rather low and changes very little, from about 0.15 to 0.1, in the observed range of the number of nodes, while skewness ranges between 5 and 0 (for $N = 2500$ and $N = 5000$ respectively). Kurtosis is in the range between 20 and 10 for the same span of network sizes, hence it is not shown due to scaling with other parameters. Those values indicate that the distribution of node response time becomes narrower and more symmetric when the number of nodes increases and that the thickness of distribution tails becomes smaller. This is to be expected since the probability distribution of block sizes is close to uniform distribution and waiting time in the block queue decreases with the decrease of the total block arrival rate.

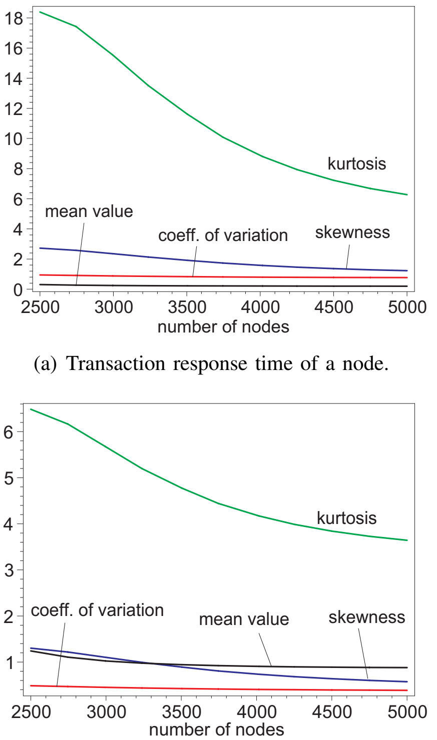
Network distribution time for blocks, shown in Fig. 7(b), is additionally influenced by the block distribution protocol. Mean number of hops in the distribution algorithm (shown with circles) increases by only 10% in the observed range of network sizes, on account of nearly constant network diameter and high number of links available. Mean block delivery time increases by almost the same amount, from 14.6s to about 16s, when the network size doubles from 2500 to 5000 nodes. At the same time, coefficient of variation is only around 0.15; skewness is between 0.8 and -0.6 which indicates thin distribution tails with right- and left-hand orientation, respectively; and kurtosis is smaller than 3. These results indicate that block propagation time follows a distribution close to the normal one, rather than exponential as reported in [5], [27]. The difference is likely due to different values of parameters such as network size and the fact that measurements on the real BTC network disrupt its operation, as noted in Section II.



(a) Block response time in a node.

(b) Block delivery time in the entire network.

Fig. 7. Performance of block traffic.



(a) Transaction response time of a node.

(b) Transaction delivery time in the entire network.

Fig. 8. Performance of transaction traffic.

D. Queuing performance of transaction traffic

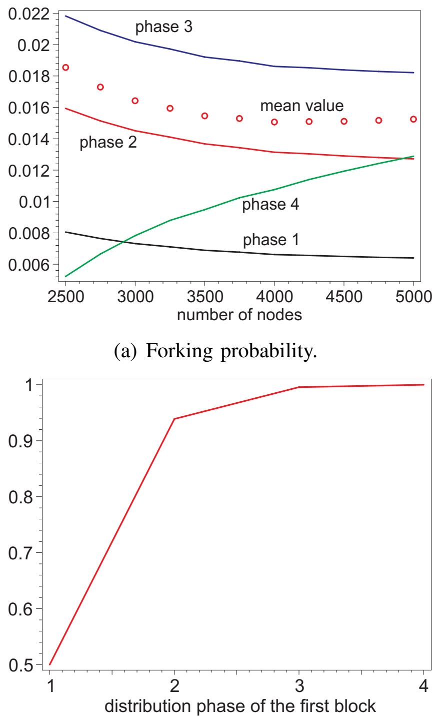
Transaction traffic has lower priority than block traffic, hence node response time for transactions reflects the dynamics of both traffic types. Fig. 8(a) shows mean value of response time which is lower than that for blocks due to much smaller transaction size, together with its coefficient of variation, skewness, and kurtosis. Since block and transaction traffic intensity declines with the increase of network size, as seen in Fig. 6, mean node response time shows a mild decrease due to the decrease of waiting time at the node buffer. It approaches transaction service time when the network size increases beyond 3000 nodes. Coefficient of variation has a value below 1, while skewness decreases from about 2.8 to below 1.4, which is below, but still close to exponential distribution. Kurtosis has values from 18 to 7 in the observed range of network sizes.

Parameters of network distribution time, mean value, coefficients of variation, skewness and kurtosis, are shown in Fig. 8(b), respectively. As can be seen, mean distribution time for transactions drops to about k times the value of transaction service time at network sizes above 3000. Such behavior is close to general Erlang-k distribution with parameter k in the range of 3 to 4.

E. Forking and inconsistency period

Probability of forking when a newly mined block $\mathcal{B}$ appears in the network during the i-th distribution phase of the original block $\mathcal{A}$, $1 \leq i \leq D_{N,Mx(z)}$, is shown in Fig. 9(a). In this case, the variable parameter is the distribution phase of the block that was first to appear (i.e., block $\mathcal{A}$ in our discussions above); mean value over all phases is shown with circles. Probability of forking in first three distribution phases from Fig. 9(a) somewhat resembles the shape of mean block arrival rate shown in Fig. 6. However, partial forking probability in phase 4 increases in rough proportion to network size, which is due to increased arrival rate of competing block(s) (block $\mathcal{B}$ in our discussions above), since node populations in early phases of block distribution are limited by node connectivity. Overall forking probability calculated via (44) first decreases with network size but then shows a slight increase when network size exceeds 4000 nodes which is due to the increased impact of forking in the last phase of block distribution. We note that mean forking probability of 1.6 to 1.8% is close to the value of 1.69% reported in [5].

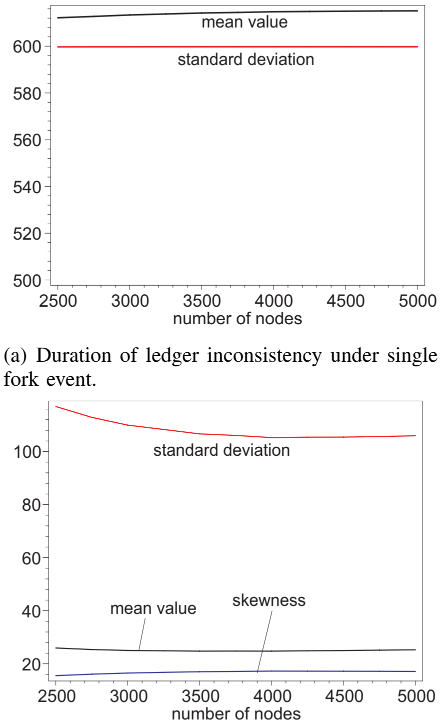
Fig. 9(b) shows the size of the partitions where nodes have linked block $\mathcal{A}$ when block $\mathcal{B}$ appears in the network, normalized to the size of the entire network. As these values are virtually independent of network size, we show them as the function of the distribution phase (1, 2, 3 and 4) of the first block to appear in the network. We observe that the partition size is about 0.5 (i.e., partitions with blocks $\mathcal{A}$ and $\mathcal{B}$ at the head of the respective blockchains have similar sizes) only if block $\mathcal{B}$ is mined during the first distribution phase of block $\mathcal{A}$. When the original block $\mathcal{A}$ has progressed to the second or later distribution phases when block $\mathcal{B}$ appears in the network, $\mathcal{A}$'s partition at the end of distribution will contain a majority of the nodes: about 94% for the second phase, and more than 98% and 99% for the third and fourth phases, respectively.



(a) Forking probability.

(b) FIrst block partition size.

Fig. 9. Pertaining to forking.



(a) Duration of ledger inconsistency under single fork event.

(b) General ledger inconsistency.

Fig. 10. Pertaining to ledger inconsistency due to forking.

Periods of distributed ledger inconsistency are shown in Fig. 10. Mean inconsistency period (solid line) after a single two-way fork and its standard deviation (circles) are shown in Fig. 10(a) while skewness (not shown) is around 2. This period is strongly influenced by the exponentially distributed time between block arrivals while block distribution time contributes to around 10% of the value even with forking probability of around 1.5%. However, given the current trend of increasing the network size and block size, forking probability and duration/characteristics of inconsistent period may well increase in future.

General inconsistency time between two block arrivals is presented in Fig. 10(b) for the mean value (solid line), standard deviation (circles), and skewness (diamonds), respectively. Here we make important observation that although mean inconsistency time of around 11 to 12 seconds is only about 2% of block interarrival time, standard deviation is higher than 100 seconds and coefficient of skewness is higher than 15, indicating a heavy tail. This confirms the impact of forking that may lead to occasional long periods of inconsistency, perhaps exceeding minutes in duration.

IX. Conclusion

In this work we have developed the analytical model of data delivery protocol over the Bitcoin blockchain network. Model incorporates many recent measurement results from the literature. We have also developed a priority-based queuing model of Bitcoin nodes and a Jackson network model of the whole network. These models give probability distributions of node populations in data distribution phases, response time of nodes, and network distribution times for both block and transaction traffic.

The results presented show strong qualitative and quantitative dependency of network performance on the distribution of node connectivity and network size. We have shown that relayed data arrive to nodes according to a non-homogeneous Poisson process and that rich relayed traffic overwhelms the traffic injected by the node itself, be it as mined blocks or new transactions. We also show that data distribution time in the network is sub-exponential and that the intensity of transaction traffic does not significantly affect performance of block traffic due to its higher priority. We demonstrate the use of the combined model on the calculation of forking probability, size of network partitions created by forking, and the duration of ledger inconsistency period. Our results indicate that node connectivity, network and block size are major factors affecting block forking probability and duration of inconsistent period.

The model described in the paper will be extended to analyze probabilities of success of events such as orphaned blocks, transaction handling, and attacks such as double spending and others [5], [24], [27]. It can also be used to predict performance changes under increase/decrease of block size, node connectivity, number of nodes, and other effects caused by changes of business profiles in the network.

References

[1] S. Ben Mariem, P. Casas, and B. Donnet. Vivisecting blockchain P2P networks: Unveiling the Bitcoin IP network. In *ACM CoNEXT Student Workshop*, 2018.

[2] R. Bowden, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor. Block arrivals in the Bitcoin blockchain. *arXiv preprint arXiv:1801.07447*, 2018.

[3] M. Campbell-Verduyn. *Bitcoin and Beyond: Cryptocurrencies, Blockchains, and Global Governance*. Routledge, 2018.

[4] S. Davidson, P. De Filippi, and J. Potts. Economics of blockchain. In *Proc. of Public Choice Conference*, Ft. Lauderdale, FL, May 2016.

[5] C. Decker and R. Wattenhofer. Information propagation in the Bitcoin network. In *Proc. 13th IEEE Int. Conf. Peer-to-Peer Computing (P2P'13)*, volume 26, 2013.

[6] S. Delgado-Segura, S. Bakshi, C. Pérez-Solà, J. Litton, A. Pachulski, A. Miller, and B. Bhattacharjee. TxProbe: Discovering Bitcoin's network topology using orphan transactions. *arXiv preprint arXiv:1812.00942*, 2018.

[7] S. Delgado-Segura, C. Pérez-Solà, J. Herrera-Joancomartí, G. Navarro-Arribas, and J. Borrell. Cryptocurrency networks: A new P2P paradigm. *Mobile Information Systems*, 2018.

[8] J. A. D. Donet, C. Pérez-Sola, and J. Herrera-Joancomartí. The Bitcoin P2P network. In *Int. Conference on Financial Cryptography and Data Security*, pages 87–102. Springer, 2014.

[9] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *arXiv preprint arXiv:1311.0243*, 2013.

[10] A. E. Gencer, S. Basu, I. Eyal, R. Van Renesse, and E. G. Sirer. Decentralization in Bitcoin and Ethereum networks. *arXiv preprint arXiv:1801.03998*, 2018.

[11] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. On the security and performance of proof of work blockchains. In *ACM SIGSAC Conf. Computer Comm. Security*, pages 3–16. ACM, 2016.

[12] A. Gervais, H. Ritzdorf, G. O. Karame, and S. Čapkun. Tampering with the delivery of blocks and transactions in Bitcoin. In *22nd ACM SIGSAC Conf. Computer Comm. Security*, pages 692–705. ACM, 2015.

[13] J. Göbel, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor. Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay. *Performance Evaluation*, 104:23–41, 2016.

[14] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes*. Oxford University Press, Oxford, UK, 2nd edition, 1992.

[15] M. Harchol-Balter and T. Osogami. Multi-server queueing systems with multiple priority classes,. *Performance Evaluation*, pages 331–360, 2005.

[16] E. Kao and K. Narayanan. Computing steady-state probabilities of a non-preemptive priority multiserver queue. *Journal on Computing*, 2(3):211–218, 1990.

[17] G. O. Karame, E. Androulaki, and S. Capkun. Double-spending fast payments in Bitcoin. In *Proc. 2012 ACM conference on Computer and communications security*, pages 906–917. ACM, 2012.

[18] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *41st Annual Symposium on Foundations of Computer Science*, pages 565–574, Redondo Beach, CA, 2000.

[19] L. J. Kleinrock. *Queuing Systems*, volume I: Theory. John Wiley and Sons, New York, 1972.

[20] M. Lischke and B. Fabian. Analyzing the Bitcoin network: The first four years. *Future Internet*, 8(1):7, 2016.

[21] A. Miller, J. Litton, A. Pachulski, N. Gupta, D. Levin, N. Spring, and B. Bhattacharjee. Discovering Bitcoin's public topology and influential nodes. report, 2015.

[22] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[23] T. Neudecker, P. Andelfinger, and H. Hartenstein. A simulation model for analysis of attacks on the Bitcoin peer-to-peer network. In *IFIP/IEEE Int. Symp. Integrated Network Mgmnt (IM)*, pages 1327–1332, 2015.

[24] T. Neudecker and H. Hartenstein. Network layer aspects of permissionless blockchains. *IEEE Communications Surveys Tutorials*, 2018. 10.1109/COMST.2018.2852480.

[25] S. Neumayer, M. Varia, and I. Eyal. An analysis of acceptance policies for blockchain transactions, 2018. https://hdl.handle.net/2144/31455.

[26] T. Nishida. Approximate analysis for heterogeneous multiprocessor systems with priority jobs. *Performance Evaluation*, 15(2):77–88, 1992.

[27] N. Papadis, S. Borst, A. Walid, M. Grissa, and L. Tassiulas. Stochastic models and wide-area network measurements for blockchain design and analysis. In *IEEE INFOCOM*, pages 2546–2554, 2018.

[28] G. Pappalardo, G. Caldarelli, and T. Aste. The Bitcoin peers network. In *2nd Int. Workshop P2P Financial Systems*, London, UK, Sept. 2016.

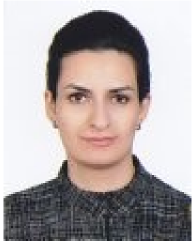
[29] D. Ron and A. Shamir. Quantitative analysis of the full Bitcoin transaction graph. In *Int. Conf. Financial Cryptography and Data Security*, pages 6–24. Springer, 2013.

[30] H. Takagi. *Queueing Analysis*, volume 1: Vacation and Priority Systems. North-Holland, Amsterdam, The Netherlands, 1991.

Jelena Mišić (M'91, SM'08, F'18) is Professor of Computer Science at Ryerson University in Toronto, Ontario, Canada. She has published 4 books, over 125 papers in archival journals and close to 190 papers at international conferences in the areas of computer networks and security. She serves on editorial boards of *IEEE Transactions on Vehicular Technology*, *IEEE IoT Journal*, *IEEE Network*, *Computer Networks* and *Ad hoc Networks*. She is a Fellow of IEEE and Member of ACM.

Vojislav B. Mišić (M'92, SM'08) is Professor of Computer Science at Ryerson University in Toronto, Ontario, Canada. His research interests include performance evaluation of wireless networks and systems and software engineering. He serves on the editorial boards of *IEEE transactions on Cloud Computing*, *Ad hoc Networks*, *Peer-to-Peer Networks and Applications*, and *International Journal of Parallel, Emergent and Distributed Systems*. He is a Senior Member of IEEE and member of ACM.

Xiaolin Chang is Professor at the School of Computer and Information Technology, Beijing Jiaotong University. Her current research interests include edge/cloud computing, network security, security and privacy in machine learning.



Saeideh Motlagh is currently a doctoral student at Ryerson University, Toronto, Canada. Her research interests include data distribution protocols in blockchain technology and applications.

M. Zulfiker Ali (S'17) is currently a post-doctoral research fellow at Ryerson University, Toronto, Canada. He received his PhD in Computer Science from Ryerson University in 2018. His research interests include wireless networking, Internet of Things and performance evaluation.