Discrete-Time Analysis of the Blockchain Distributed Ledger Technology

Stefan Geissler*, Thomas Prantl*, Stanislav Lange§, Florian Wamser*, Tobias Hossfeld*

*University of Würzburg, Institute of Computer Science, Würzburg, Germany
{stefan.geissler|thomas.prantl|florian.wamser|tobias.hossfeld}@informatik.uni-wuerzburg.de
§Department of Computer Science and Engineering, POSTECH, Korea
stasl@postech.ac.kr

***Abstract*—Blockchain and distributed ledger technologies have become more and more popular and widespread during recent years. After the initial hype about the technology and many cryptocurrency related use cases, the technology slowly starts to make its way into other domains like food tracking and document management. In order to further contribute to the search of what this technology can be used for, more detailed performance evaluations are required in order to investigate key performance indicators and general limits of the technology. To this end, we develop a discrete-time queueing model that allows a detailed evaluation of the characteristics of a blockchain system, such as the transaction waiting time distribution. Furthermore, we validate the model by comparing the results to values obtained from measurements in a closed lab environment based on the Ethereum blockchain.**

***Index Terms*—Blockchain, Distributed ledger, Discrete-time analysis, Ethereum, Modeling, Performance Evaluation, Measurement.**

I. Introduction

With the rise of cryptocurrency platforms such as Bitcoin [1] and Ethereum [2], their fundamental technology, blockchain, has gained significant traction in several areas. Countless applications have been devised, which are all based on the same technological principle of the blockchain and distributed ledger technologies. This principle, to have a public, immutable ledger that is not governed by a central authority, but instead maintained by a distributed network of equal peers has led to a plethora of new use cases such as decentralizing privacy [3], privacy-preserving smart contracts [4], as well as applications in the internet-of-things [5, 6].

However, since all of these use cases are realized with the same underlying system, they also inherit all the accompanying challenges. One of the most crucial challenges of modern blockchain systems is their throughput in terms of transactions per second [7, 8, 9].

Hence, the goal of this paper is to evaluate key performance indicators and influence parameters of the fundamental technology all of these use cases have in common, the blockchain or distributed ledger technology. To this end, we present an abstract discrete-time model of a blockchain system that can, due to its generality, be applied to various use cases built upon the technology. We provide a detailed description of the model and perform an extensive validation based on measurement values obtained from a private instance of the Ethereum blockchain in a closed environment.

Our evaluation shows that the model is accurate when it comes to real world scenarios. Additionally, the model can be used to perform extensive parameter studies in order to identify key influence factors of a system to pinpoint optimization potential.

The remainder of this paper is structured as follows. Section II provides a brief introduction to the blockchain technology as it is required for the proposed model. Section III presents related work regarding the performance evaluation of blockchain-based systems as well as relevant analytical models. A description of the proposed model is presented in Section IV. Section V presents a parameter study to investigate the impact of different input parameters. In Section VI, we provide a comparison between baseline data obtained from the Ethereum blockchain and results obtained from the model. Finally, Section VII summarizes the content of this paper and outlines future research directions.

II. Background on Blockchain Technology

This section describes the basic functionality of the blockchain technology and declares the abstractions introduced during the development of the model.

A blockchain is, in its most basic form, a distributed data structure that is fully replicated by every member of an overlay peer-to-peer network. It was originally introduced to solve the double spending problem [10] in a fully decentralized electronic cash system [1] with cryptocurrencies [11] being its most prominent manifestation. To most basic version of the blockchain protocol employs computational puzzles to limit write access to the distributed data structure in a mechanism called Proof-of-Work (PoW) [12]. While other mechanisms, such as Proof-of-Stake [13], exist, Proof-of-Work is still the most prominent solution. The cryptographic problems used by PoW ensure that the system is working correctly as long as the majority of computational power in the network is provided by honest members, as opposed to relying on the number of honest participants in the network. This mechanism is, on the one hand, used to ensure that data written to the

data structure is valid, as it enforces significant computational effort to acquire write permission and invalid data can easily be detected by other peers in the system. On the other hand, the mechanism is used to achieve consensus in the network in case of write conflicts [14]. As opposed to common distributed systems, no special consensus protocol to resolve conflicts is used in blockchain environments. This form of consensus is often referred to as probabilistic consensus since the system continuously converges against a valid state on its own, instead of specifically resolving conflicts. The system can thus be described as a probabilistic state machine [15]. In the following, we provide a high level overview of the functionality of a blockchain or distributed ledger system and cover the features of the system required for the remainder of this paper. A more in-depth description of the technology and its workings has been published in [14].

1) A member of the network has the intent to publish data, e.g., a transaction to transfer an asset to another user of the network. Note that the data to be published can be arbitrary and is, in general, not limited to financial transactions. These transactions can in the process be equipped with a priority, based on the fee provided by the creator of a transaction that is paid out to the user who later validates this transaction into a new block.
2) The blockchain client now distributes the data, called transaction, to connected peers in the overlay network, which in turn distribute the data to their connected peers recursively in order to make the data known to the whole network.
3) Special nodes in the network, called Miners, subsequently validate all open transactions and upon success try to incorporate them in newly generated blocks by solving the mentioned cryptographic riddles to obtain write access to the data structure. Thereby, open transactions are in general prioritized based on their fees, as the goal of the Miner is to maximize the reward for generating a new block.
4) As soon as one of the Miners finds a solution to the computational puzzle, he generates a new block, thereby incorporating its solution into the header, and distributes the newly mined block to his connected peers. These can now validate the block by checking whether the included solution is valid and recursively distribute the new block among their peers, respectively. If several Miners simultaneously find and distribute different valid possibilities for the next block, other Miners generally accept the first version they receive from peers. This means that at any point in time, there can be several different valid blockchain forks present in the network. This situation is, unlike in common distributed systems, not actively resolved through the use of consensus algorithms. Instead, the different versions co-exist until one Miner finds another block, thereby extending its version of the chain by one more block compared to other currently existing versions, which is then accepted by all Miners as a longer chain that replaces all versions with less blocks. A more in-depth description of the consensus mechanisms in blockchain systems can be found in [14].
5) Finally, the whole network has been informed about a newly mined block and the transactions included are thus replicated by all members of the network.

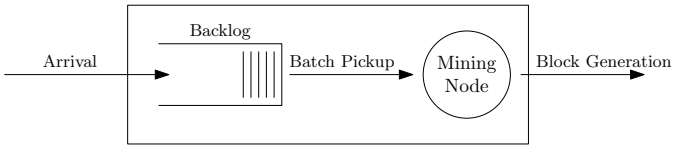

Fig. 1. Schematic presentation of the modeled system.

Note that blockchain networks work asynchronously and the steps outlined above all occur concurrently in different parts of the network. The central feature of the used mechanism revolves around the fact that all elements of the network continuously converge towards a common and valid state, independent of propagation delays as well as network topology.

In this work, we aim to evaluate the performance of Step 3 in the listing above, as it describes the number of newly generated blocks as well as the number of included transactions per block and thus dictates the overall performance of the network. For the remainder of this work, we neglect transaction priorities as well as the time it takes to distribute transactions and blocks over the network and do not include the impact of blockchain forks [16] on the performance of the system.

Figure 1 shows the abstracted system used for the model developed in this work. Transactions arrive at the system according to a specified rate and are stored in a queue of infinite size, called backlog. From there, transactions are confirmed by a batch processing mechanism in intervals according to the inter-block generation time. A more detailed description of the parameters of the model follows in Section IV.

III. Related Work

In this section, we cover relevant publications that aim at evaluating the performance of modern blockchain systems as well as works in the area of systems modeling that apply queueing theory.

A. Experimentation and General Work

A great deal of work has already been done in regard to experimental evaluation and general discussion of modern blockchain systems, especially Bitcoin and Ethereum. Swan performed an extensive dissection regarding the development and chances of blockchain [17]. Tschorsch and Scheuermann discuss research directions and show that the blockchain technology impacts fields beyond cryptocurrencies [18]. Zheng et al. [14] discuss various different consensus mechanisms and lay out possible future developments for the blockchain technology such as testing and evaluation procedures for blockchain systems. The model proposed in this work is a

first step to establish an evaluation framework for blockchain systems.

B. Systems Modeling

Blockchain systems essentially behave like bulk queuing systems, which have already been studied in the past [19]. In case of Markovian bulk input $M^{[x]}/M/1$ as well as bulk processing systems $M/M^{[x]}/1$, closed form solutions are provided by [20]. While systems exhibiting bulk-arrival processes have been studied extensively [21, 22] and have been applied to several real world use cases [23, 24, 25], models that focus on batch processing, as present in modern blockchain systems, are largely missing to date. We recently published another model in this area that describes the behavior of batched processing in the context of virtual network functions [26]. However, models targeting the specific behavior of blockchain systems are still largely missing in the literature. Li et al. have proposed a Markovian batch-service model to describe the average number of transactions waiting to be confirmed, their average confirmation time, as well as the mean number of transactions per block [27]. Pass et al. analyze the blockchain protocol in asynchronous networks and show that the applied consensus mechanism guarantees consistency and liveness even when taking information propagation within the network [16] into account [28].

IV. Model Description

In the following section, we describe our generic $GI/GI^N/1$ model of the blockchain mining process. In order to simplify the description of the model itself and improve readability, we first introduce the notation that is used throughout the rest of the paper. Table I shows the variables and their description as a central reference. The top half describes input parameters of our model, while the bottom half describes the model output and consists of key performance indicators of the blockchain system.

Furthermore, we follow the convention of noting random variables as uppercase letters such as A, while their respective distributions are written as

$$a(k) =_{\text{def}} \mathrm{P}(A = k), \quad k \in [0, \infty)\,.$$

The associated distribution function is defined as

$$A(k) =_{\text{def}} \mathrm{P}(A \leq i) = \sum_{i=0}^{k} a(k), \quad k \in [0, \infty)\,.$$

As already mentioned, we neglect information propagation delays and assume immediate distribution of transactions and blocks to all peers in the network. In other words, we collapse the whole network into a single node that generates new blocks according to $t(k)$ and at which transactions arrive with interarrival times according to $a(k)$. From this, the model is built around a fixed-point iteration of the queue size distribution. To achieve this, we represent the system state by the queue size Q_n at the time the n-th block is generated. The queue size has been selected as it is directly influenced by all possible system events, as shown in Figure 2.

TABLE I
Notation.

Variable	Description
β	Maximum block size.
A, $a(k)$	Transaction interarrival time.
T, $t(k)$	Service time of blocks.
S, $s(k)$	Size of transactions.
$x_{\tau,a}(k)$	Number of arrivals with interarrival time distribution a in an interval with duration distributed according to τ [29].
Q_n, $q_n(k)$	Queue size in number of transactions immediately before the n-th block has been generated.
Q, $q(k)$	Queue size in number of transactions at embedding times.
Q_b, $q_b(k)$	Queue size in byte at embedding times.
$\bar{Q}$, $\bar{q}(k)$	Queue size in number of transactions at random times.
$\bar{Q}_b$, $\bar{q}_b(k)$	Queue size in byte at random times.
O_n, $o_n(k)$	Block size in number of transactions for specific queue size n.
O, $o(k)$	General block size in number of transactions.
W, $w(k)$	Transaction waiting time.

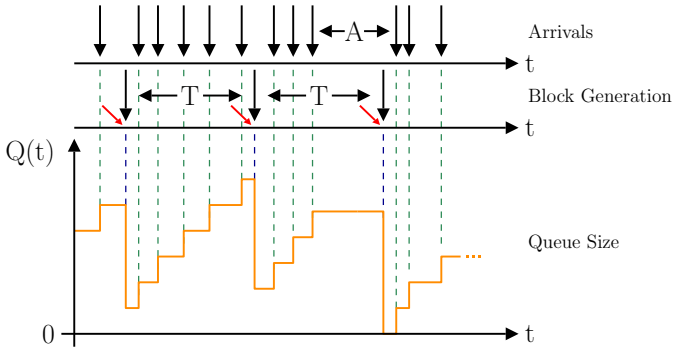

Fig. 2. Development of the queue size over time.

Every arriving transaction increases the queue size Q by one, while every block generation decreases the queue size by confirming a number of transactions in one batch. The number of transactions included in each block is thereby determined by the distribution of transaction sizes $s(k)$ as well as the maximum block size β and can be calculated as

$$o_n(k) = \begin{cases} \sum_{i=1}^{\beta} s^{*k}(i), \mathrm{k = n} \\ \sum_{i=1}^{\beta} s^{*k}(i) - \sum_{i=1}^{\beta} s^{*(k+1)}(i), 0 < \mathrm{k} < \mathrm{n} \\ 0, \mathrm{else.} \end{cases} \tag{1}$$

with $s^{*k}(i)$ describing the k-fold convolution of s with itself, evaluated at i and n being the current queue size.

Furthermore, since the blockchain backlog currently has no size limit, no transaction will be discarded, even if the arrival rate is larger than the processing rate, in which case the queue size will diverge to $+\infty$. Hence, the model is limited to system loads of $\rho < 1$, as otherwise, the following fixed-point iteration of the queue size does not converge. The iteration is based

on an embedded Markov chain with embedding times right before a block generation event as indicated by the red arrows in Figure 2. At these embedding times, we can calculate the progression of the queue size distribution as

$$q_{n+1}(k) = \sum_{i=0}^{\infty} q_n(i) \cdot \sum_{j=0}^{\infty} o_i(j) \cdot x_{t,a}(k-i+j). \quad (2)$$

Thereby, we recursively progress from embedding time n to $n+1$ by iterating over all possible queue sizes at time n, and weighting the probability of each value of Q_n with the probabilities of all possible numbers of transactions in the generated block (1) as well as $x_{t,a}(i)$ as the number of new arrivals during the corresponding inter block time T according to [29]. Finally, under the abovementioned condition that $\rho < 1$, the index n can be omitted in (2) and we get

$$q(k) = \lim_{n \to \infty} q_n(k). \quad (3)$$

Based on Equations 1 and 3, we can now infer the general block size distribution in terms of the number of transactions included in each block as

$$o(k) = \sum_{i=0}^{\infty} o_i(k) \cdot q(i). \quad (4)$$

Furthermore, we can obtain the queue size distribution at random times via Equation 5. We thereby exploit the temporal sequence of the queue size distribution by weighing the probability for a certain queue size with the recurrence time $E[R_A]$ if the queue size is assumed in the beginning or the end of an inter-block interval. If a queue size level is assumed at any other point in the interval, the corresponding probability is weighed with the mean interarrival time $E[A]$.

From the queue size $\bar{Q}$ at random times, we can now derive further key performance indicators of the system. The backlog, represented by the queue size in bytes instead of the number of transactions, can be obtained as

$$\bar{q}_b(k) = \sum_{i=0}^{\infty} q(i) \cdot s^{*i}(k). \quad (6)$$

Analogously, we can compute the queue size in bytes at embedding times by using $q(k)$ instead of $\bar{q}(k)$.

As a key performance indicator, we show the evaluation of the waiting time distribution, which, in this context, equals the time to confirmation of transactions. To this end, we follow a recursive approach to determine the distribution of the number of block generation cycles required until a transaction is confirmed:

$$c_n(1) = \sum_{i=n}^{\infty} o_n(i). \quad (7)$$

$$c_n(k) = \sum_{i=1}^{n-k+1} o_n(i) \cdot c_{n-i}(k-1). \quad (8)$$

Thereby, we calculate $c_n(k)$ by recursively calculating the probability to pick up i transactions in the k-th cycle and multiplying it with the probability of having to pick up $n-i$ transactions in $k-1$ cycles. Based on this, we can then omit the index n by calculating the weighted probability while taking into account the queue size distribution at random times $\bar{Q}$ as

$$c(k) = \sum_{i=0}^{\infty} \bar{q}(i) \cdot c_i(k). \quad (9)$$

We can then determine the waiting time distribution as

$$w(k) = \sum_{i=1}^{\infty} (r_T * t^{*(i-1)})(k) \cdot c(i), \quad (10)$$

with r_T being the recurrence time distribution of the block generation process according to

$$r_T(x) = \frac{1}{E[T]} \cdot (1 - T(x)). \quad (11)$$

Here we determine the probability for one recurrence time as well as i pickup cycles taking exactly duration k and calculate the weighted sum by using the probability of i pickup cycles $c(i)$ as the weight.

V. Performance Evaluation

In this section, we perform a parameter study to investigate the influence of different input parameters on key performance characteristics of blockchain and distributed ledger systems. To this end, we evaluate the impact of the arrival process as well as transaction sizes on the queue size as well as the waiting time until transactions are processed. Other parameters, such as the service time of blocks or the maximum block size are regarded as fixed, deterministic values throughout the following evaluation scenarios. Furthermore, in order to better isolate the influence of single input parameters, we also regard the transaction size S as constant throughout the parameter study. This is necessary due to the dependencies of the various input parameters when it comes to the system load ρ as is shown in (12).

$$\rho = \frac{E[T]}{E[A]} \cdot \frac{E[S]}{\beta} \quad (12)$$

Accordingly, since $E[T]$, $E[S]$, as well as β are deterministic, we can control the system load though adaptation of $E[A]$. Hence, whenever results are shown for increasing load, this is achieved by reducing the mean interarrival time between transactions. Finally, for all evaluations shown in this paper, we assume A to follow a negative binomial distribution as it is a close approximation of bursty traffic [30] and allows the precise configuration of the coefficient of variation c_A.

A. Queue Size

First, we examine the influence of system load as defined by the mean interarrival time, as well as the impact of the coefficient of variation on the queue size. Figure 3 shows the queue size distribution for two different load levels, 0.75 and 0.95, each for a coefficient of variation of 0.25 as well as 1.25. Thereby, the figure shows the number of transactions waiting to be confirmed along the x-axis and the corresponding

$$\bar{q}(k) = \sum_{i=0}^{\infty} q(i) \cdot o_i(i-k) \cdot x_{t,a}(o) + \sum_{i=0}^{\infty} q(i) \cdot o_i(i-k) \cdot \sum_{l=1}^{\infty} x_{t,a}(l) \cdot \frac{E[R_A]}{(l-1)E[A] + 2E[R_A]} + \sum_{i=0}^{\infty} q(i) \sum_{j=0}^{i} o_i(j) \cdot x_{t,a}(k+j-i) \cdot \frac{E[R_A]}{(k+j-i-1)E[A] + 2E[R_A]} + \sum_{i=0}^{\infty} q(i) \sum_{j=i-k+1}^{i} o_i(j) \sum_{l=k+j-i+1}^{\infty} x_{t,a}(l) \cdot \frac{E[A]}{(l-1)E[A] + 2E[R_A]}. \tag{5}$$

probability along the y-axis. Note that these probabilities are valid for arbitrary times, not only for the embedding times of the Markov chain.

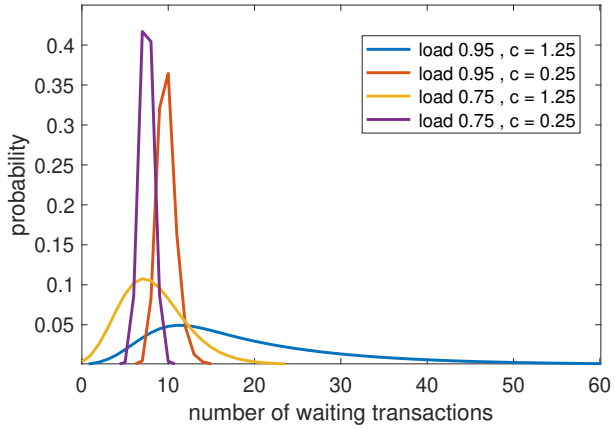

Fig. 3. Queue size distribution in number of transactions for $\rho \in \{0.75, 0.95\}$, $c_A \in \{0.25, 0.75\}$, $E[S] = 150$, $E[T] = 10000$, and a maximum block size of $\beta = 1500$.

As is expected, the figure shows that the coefficient of variation of the arrival process has a significantly larger impact on the distribution of the number of waiting transactions than the system load ρ. This is explained by the increased burstiness of arrival processes with a high coefficient of variation. This burstiness is directly reflected in a higher variation of the queue size distribution when compared to arrival processes with lower variance.

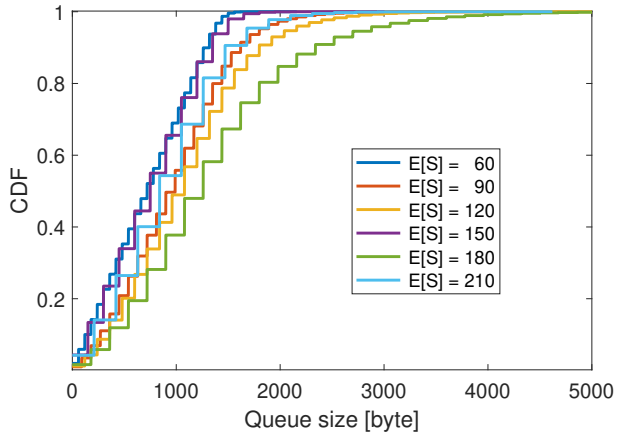

Fig. 4. Queue size distribution in byte for different deterministic transaction sizes, $\rho = 0.95$, $c_A = 0.25$, and a maximum block size of $\beta = 1500$.

Next, we investigate the impact of transaction sizes on the queue size. Thereby, we increase the transaction size in steps of 30 from 60 to 210. First, note that in this case we denote transaction sizes in byte. The model, however is not limited to any specific unit and can be applied to any arbitrary transaction size unit, e.g., gas in case of Ethereum [2] like we do in Section VI. Second, since according to (12), the transaction size directly influences the system load, in this scenario we modify the arrival process for each different value of S to achieve a consistent system load of $\rho = 0.95$. Finally, in this scenario, the number of transactions is no longer a comparable metric. Hence, we now introduce the queue size distribution in byte $\bar{Q}_b$ instead of the number of waiting transactions, which allows us to directly compare the resulting distributions.

Figure 4 shows the queue size along the x-axis and the cumulative distribution function along the y-axis with each color representing a different transaction size value from 60 to 210. It can be seen that the queue size distribution in byte does not directly follow the transaction size. This is explained by a closer examination of the used parameter combination. A blocksize $\beta = 1500$ in combination with deterministic transaction sizes leads to different values regarding the block utilization. In this work, we define the block utilization as

$$u_t(s) = \left\lfloor \frac{\beta}{E[S]} \right\rfloor \cdot \frac{E[S]}{\beta}$$

as it describes the efficiency of the usage of available space within one block. Table II describes the utilization values that lead to the curves seen in Figure 4. It can be seen that the curves are ordered by utilization group, meaning parameter combination with similar block utilizations also result in similar queue sizes.

TABLE II
BLOCK UTILIZATION [%] FOR DIFFERENT TRANSACTION SIZES AND A BLOCK SIZE OF $\beta = 1500$.

E[T] \ E[S]	60	90	120	150	180	210
$u_t(s)$	1	0.96	0.96	1	0.96	0.98
	Queue Size at random time					
$E[\bar{Q}_b]$	722.0	964.1	1072.6	759.8	1318.6	900.4
$E[\bar{Q}]$	12.0	10.7	8.9	5.1	7.3	4.3
	Queue Size at embedding times					
$E[Q_b]$	1434.4	1676.6	1785.0	1472.3	2031.1	1612.9
$E[Q]$	24.0	18.6	14.9	9.8	11.3	7.6

B. Waiting Time

In the second part, we investigate the waiting time distribution of transactions for different input parameters. To this end, we again isolate the key influence factors by keeping all but one parameter fixed in order to examine the influence of a single input value. In this scenario, we first examine the impact of different transaction sizes S on the mean waiting time $E[W]$. Figure 5 shows for increasing load levels between 0.75 and 0.95 along the x-axis the mean waiting time in seconds along the y-axis. Different transaction sizes between 60 and 210 are indicated by the different colors according to the legend. Similarly to the queue size evaluation shown in Figure 4, the mean waiting time does not increase continuously with increasing transaction sizes, but rather depends on the block utilization as shown in Table II. As a higher block utilization leads to more efficient packing of transactions into the available space within each block, more transactions can be confirmed in each cycle, leading to quicker confirmation times in return.

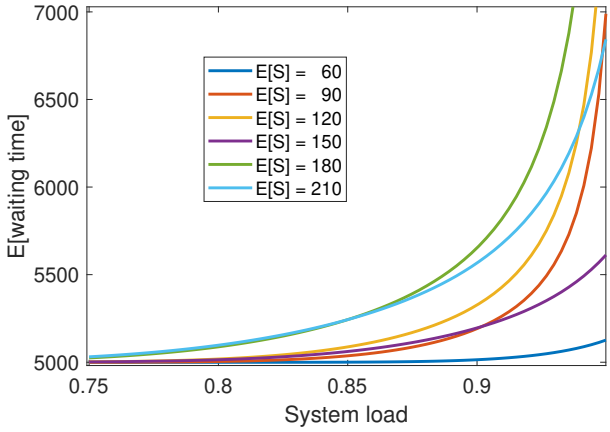

Fig. 5. Mean waiting time W for different transaction sizes, $c_A = 0.25$, ρ between 0.75 and 0.95, $E[T] = 10000$, and a maximum block size of $\beta = 1500$.

Next, we evaluate the impact of the coefficient of variation of the arrival process c_A on the mean waiting time. We here take a closer look on the scenarios with a block utilization of $u_t(s) = 1$, as these are the most efficient cases. Figure 6 shows the mean waiting time for transactions of size 60 along the y-axis, different load levels along the x-axis, and various values for c_A according to the annotation. Accordingly, Figure 7 shows the same values for transactions of size 150. Again, it can be seen that the coefficient of variation of the arrival process has a much more significant impact on the mean waiting time than the transaction size, even though both transaction sizes result in a high block utilization.

VI. Validation

In the final part of our evaluation, we validate the previously presented model. To this end, we perform measurements using a private instance of the Ethereum blockchain. In order to

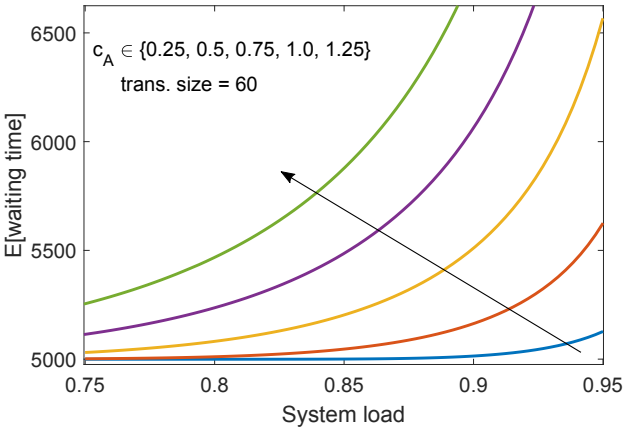

Fig. 6. Mean waiting time W for different c_A, ρ between 0.7 and 0.95, $E[S] = 60$, $E[T] = 10000$, and a maximum block size of $\beta = 1500$.

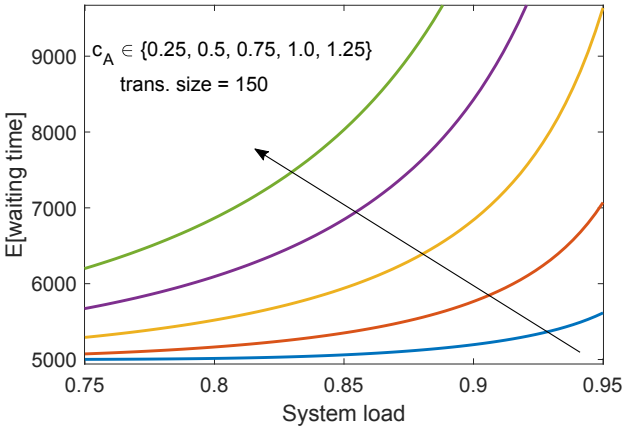

Fig. 7. Mean waiting time W for different c_A, ρ between 0.7 and 0.95, $E[S] = 150$, $E[T] = 10000$, and a maximum block size of $\beta = 1500$.

create comparable scenarios, we have developed a testbed consisting of a private, isolated Ethereum blockchain using a single Geth client[1], a Go implementation of the Ethereum protocol. Using this implementation, we generate transactions of deterministic size of 21000 gas. The blocksize of our private blockchain is thereby 210000 gas. Gas is the measure of complexity of a transaction in the Ethereum blockchain and is used to limit the number of transactions included in a single block, as opposed to byte as in the case of Bitcoin. Due to this setup, a block can contain a constant number of 10 transactions. Based on this setup, we validate our model using the Proof-of-Authority consensus mechanism. This mechanism delegates the block generation process to a dedicated set of nodes that hold full authority on which transactions are included in a block and when a new block is generated. A more detailed description of the PoA mechanism can be found in [31]. This specific behavior allows us to determine the exact time between block generation events, which we

[1]https://github.com/ethereum/go-ethereum

keep deterministic at $T = 15000$ milliseconds in this case. In combination with the setup of a single block having enough capacity for exactly 10 transactions once more allows us to isolate a single parameter, the arrival process. In addition, varying the arrival process allows us to achieve a specific load level. In the following, we show results for a coefficient of variation $c_A = 1.25$ for load levels of $\rho = 0.75$ as well as $\rho = 0.95$ in Figure 8 and Figure 9, respectively. We thereby monitor the waiting times for 20000 transactions as observed in the test environment and compare these results to the prediction from our model.

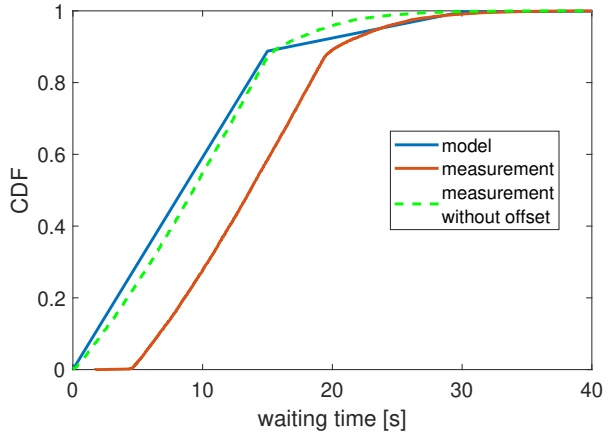

Fig. 8. CDF of waiting time W, $T = 15000$, $\beta = 210000$, $E[S] = 21000$, $c_A = 1.25$, and $\rho = 0.75$.

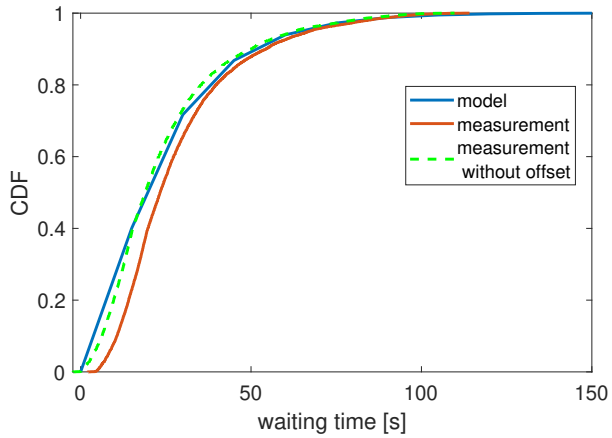

Fig. 9. CDF of waiting time W, $T = 15000$, $\beta = 210000$, $E[S] = 21000$, $c_A = 1.25$, and $\rho = 0.95$.

Both Figures 8 and 9 show the CDF of the waiting time W for both the model values in blue as well as the measurement in red. It can be seen that in both cases, the measurements differ significantly from the model values. This is explained by a characteristic of the technical system used to obtain the measurements. The model assumes that whenever a new transaction arrives in the system, it is, if there is enough capacity, included in the next generated block. However, the technical system functions slightly differently as it shifts transactions that arrive within 4.4 seconds before a new block is generated into the next block and only considers transactions that have arrived before this threshold. Hence, the CDF of the measurement is shifted by this amount. To this end, we include a third curve in our figures, shown in green, that represents the measurement values without this technical offset that have been calculated by simply subtracting the offset of 4.4 seconds from all measurement values. It can be seen that this curve fits closely to the values obtained via the model.

VII. Conclusion

Blockchain and distributed ledger technologies, initially only known for cryptocurrencies, have lately made their way into the focus of the research community and many the industry has started to evaluate the technology regarding its suitability for more and more use cases. Many of these have the same fundamental question when it comes to the blockchain technology. Namely, whether the technology is able to provide the required performance for the different use cases.

In order to address this question, we have developed a discrete-time queueing model that allows the evaluation of key performance indicators of blockchain and distributed ledger systems. Our $GI/GI^N/1$ model enables the evaluation and prediction of the queue size as well as transaction waiting time distribution based on various input parameters. Based on this model, we have performed a first parameter study to evaluate the impact of different input parameters on the system behavior. We have further investigated the validity of the model by means of measurements in a controlled environment based on a private Ethereum blockchain. This evaluation has shown that the predictions of the model hold true for the Proof-of-Authority consensus mechanism. Future research directions include a more in-depth parameter study to further investigate the impact of different system parameters as well as the investigation of the impact of different distribution types on system performance as well as further validation steps using different consensus mechanisms.

VIII. Acknowledgement

This research has been funded by the Federal Ministry of Education and Research of Germany in the framework KMU-innovativ - Verbundprojekt: Secure Internet of Things Management Platform - SIMPL (project number 16KIS0852) [32].

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
[2] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, 2014.
[3] G. Zyskind, O. Nathan *et al.*, "Decentralizing privacy: Using blockchain to protect personal data," in *IEEE Security and Privacy Workshops (SPW)*, 2015.
[4] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *IEEE symposium on security and privacy (SP)*, 2016.
[5] S. Huh, S. Cho, and S. Kim, "Managing iot devices using blockchain platform," in *International Conference on Advanced Communication Technology (ICACT)*, 2017.

[6] N. Kshetri, "Can blockchain strengthen the internet of things?" *IT Professional*, 2017.

[7] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work vs. bft replication," in *International Workshop on Open Problems in Network Security*, 2015.

[8] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: a survey," *International Journal of Web and Grid Services*, 2018.

[9] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer *et al.*, "On scaling decentralized blockchains," in *International Conference on Financial Cryptography and Data Security*, 2016.

[10] G. Karame, E. Androulaki, and S. Capkun, "Two bitcoins at the price of one? double-spending attacks on fast payments in bitcoin." *IACR Cryptology ePrint Archive*, 2012.

[11] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press, 2016.

[12] C. Dwork and M. Naor, "Pricing via processing or combatting junk mail," in *Annual International Cryptology Conference*, 1992.

[13] V. Buterin and V. Griffith, "Casper the friendly finality gadget," *arXiv preprint arXiv:1710.09437*, 2017.

[14] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *IEEE International Congress on Big Data (BigData Congress)*, 2017.

[15] K. Saito and H. Yamada, "What's so different about blockchain?—blockchain is a probabilistic state machine," in *IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2016.

[16] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2013.

[17] M. Swan, *Blockchain: Blueprint for a new economy*. " O'Reilly Media, Inc.", 2015.

[18] F. Tschorsch and B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," *IEEE Communications Surveys & Tutorials*, 2016.

[31] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Pbft vs proof-of-authority: applying the cap theorem to permissioned blockchain," 2018.

[19] I. W. Kabak, "Blocking and delays in m (x)/m/c bulk arrival queueing systems," *Management Science*, 1970.

[20] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris, *Fundamentals of queueing theory*. John Wiley & Sons, 2018, vol. 399.

[21] D. Manfield and P. Tran-Gia, "Analysis of a finite storage system with batch input arising out of message packetization," *IEEE Transactions on Communications*, 1982.

[22] D. M. Lucantoni, "New results on the single server queue with a batch markovian arrival process," *Communications in Statistics. Stochastic Models*, 1991.

[23] E. Altman, K. Avrachenkov, and C. Barakat, "A stochastic model of tcp/ip with stationary random losses," *ACM SIGCOMM Computer Communication Review*, 2000.

[24] A. Klemm, C. Lindemann, and M. Lohmann, "Modeling ip traffic using the batch markovian arrival process," *Performance Evaluation*, 2003.

[25] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling tcp throughput: A simple model and its empirical validation," *ACM SIGCOMM Computer Communication Review*, 1998.

[26] S. Lange, L. Linguaglossa, S. Geissler, D. Rossi, and T. Zinner, "Discrete-time modeling of nfv accelerators that exploit batched processing," in *IEEE Conference on Computer Communications (INFOCOM)*, 2019, to be published.

[27] Q.-L. Li, J.-Y. Ma, and Y.-X. Chang, "Blockchain queueing theory," *arXiv preprint arXiv:1808.01795*, 2018.

[28] R. Pass, L. Seeman, and A. Shelat, "Analysis of the blockchain protocol in asynchronous networks," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2017.

[29] S. Gebert, T. Zinner, S. Lange, C. Schwartz, and P. Tran-Gia, "Discrete-time analysis: Deriving the distribution of the number of events in an arbitrarily distributed interval," University of Wuerzburg, Tech. Rep., 2016.

[30] C. Larsson, *Design of Modern Communication Networks: Methods and Applications*. Academic Press, 2014.

[32] KMU-innovativ, "SIMPL Verwaltungsplattform für ein sicheres Internet der Dinge," https://www.forschung-it-sicherheit-kommunikationssysteme.de/projekte/simpl/, 2019, [Online; accessed 18-January-2019].