Confirmed or Dropped? Reliability Analysis of Transactions in PoW Blockchains

Ivan Malakhov, Andrea Marin, *Senior Member, IEEE*, Sabina Rossi, and Daniel Sadoc Menasché

***Abstract*—Blockchains based on Proof-of-Work (PoW) have introduced a new paradigm for distributed ledgers on the Web. In these systems, transactions compete to obtain a position inside the new blocks by offering a fee to be confirmed before others. A finite amount of memory is devoted to store unconfirmed transactions, called Mempool. When new transactions arrive and the Mempool is full, silent droppings of the cheapest unconfirmed transactions occur, without any notification to the owners. This challenge becomes particularly pressing as users have the freedom to append various types of data to the blockchain, including large media files, leading to swift Mempool depletion. In this article, we study the reliability of PoW blockchains from a user perspective. We provide a numerical model to answer the question: *What is the probability of confirmation for a transaction offering a fee f when the system is in a certain state?* Our model allows blockchain-based applications to analyse the trade-off between running costs and reliability, i.e., fees offered for the transactions and probability that the transactions will be eventually confirmed. The proposed method is proactive and does not require historical data on dropped transactions that, in fact, are not logged anywhere in the blockchain. This article presents significant contributions, summarized as follows: (i) the introduction of a stochastic model and its efficient solution for analyzing dropping probability in blockchain systems; (ii) validation of the model through real traces extracted from the Bitcoin blockchain.**

***Index Terms*—Blockchain, PoW, reliability analysis, markovian models.**

I. Introduction

Blockchains are distributed ledgers that store transactions clustered into blocks. A network of peers validates, stores, and guarantees the immutability of information according to some algorithms that characterize the blockchains. Henceforth, we will focus on the most popular class of blockchains, i.e., those inspired by Nakamoto's Proof-of-Work (PoW), as implemented in Bitcoin[1] and other networks, such as Bitcoin cash[2], Litecoin[3] and Ethereum Classic[4].

Transactions are added to blocks by special users called *miners* and are selected from a queue of pending transactions, namely the *Mempool*, according to an auction-based policy. Each block is usually formed with the transactions offering the highest ratios of fee per byte. The particular policy adopted by Bitcoin is described in Section II. However, when the number of pending transactions exceeds a certain threshold the least valuable transactions are evicted and will not be included in the blockchain unless they are rebroadcasted later with a possibly higher fee.

The challenge of dealing with large Mempools becomes particularly pressing as users have the freedom to append various types of data to the blockchain, including large media files, leading to swift Mempool depletion. As an example, between February and May of 2023, the Bitcoin network was flooded with transactions with very low ratios of fee per byte (with an average around 5 sat/B). This phenomenon was caused by an implementation of fungible tokens, such as BRC-20 [1], similar in spirit to the ERC-20 Ethereum tokens[5]. Roughly speaking, such extensions allow Bitcoin clients to attach a large amount of information to each individual satoshi in the network, thanks to the Ordinals protocol[6]. As a consequence, we observe that the Mempool is continuously filled with transactions that transfer satoshis "inscribed" with user data, causing delays and dropping of transactions.

To assess the extent to which the dropping of transactions impacts the real Bitcoin network, we leverage more than six years of data monitored from the Bitcoin blockchain, with a modified mining tool that has a virtually infinite Mempool. Comparing the virtual Mempool against the actual one, we determine the fraction of transactions that were dropped. Figs. 1 and 2 show the states of the Mempools with default and infinite capacity, respectively, between April and May 2023 [13]. The data is categorized into various fee tiers, in satoshis per byte. The bottommost colored segment represents transactions that offer the least amount of satoshis per byte. Clearly, more than 50% of the infinite Mempool is occupied most of the time by

Manuscript received 1 September 2023; revised 7 December 2023; accepted 19 January 2024. Date of publication 30 January 2024; date of current version 12 June 2024. The work of Ivan Malakhov, Andrea Marin, and Sabina Rossi was supported in part by Project PRIN 2020 Nirvana - Noninterference and Reversibility Analysis in Private Blockchains, in part by Project Indam-GNCS 2023 RISICO, and in part by Project SERICS (PE00000014) through the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. The work of Daniel Sadoc Menasché was supported in part by CAPES, CNPq, and FAPERJ under Grant SEI E-26/201.376/2021. Recommended for acceptance by Dr. Chau Yuen. *(Corresponding author: Andrea Marin.)*

Ivan Malakhov, Andrea Marin, and Sabina Rossi are with the DAIS, Università Ca' Foscari Venezia, Venezia 30122, Italia (e-mail: ivan.malakhov@unive.it; marin@unive.it; sabina.rossi@unive.it).

Daniel Sadoc Menasché is with the Federal University of Rio de Janeiro, Rio de Janeiro 21941590, Brazil (e-mail: sadoc@dcc.ufrj.br).

Digital Object Identifier 10.1109/TNSE.2024.3360080

[1] https://bitcoin.org
[2] https://bitcoincash.org
[3] https://litecoin.org
[4] https://ethereumclassic.org
[5] https://ethereum.org/en/developers/docs/standards/tokens/erc-20
[6] https://docs.ordinals.com

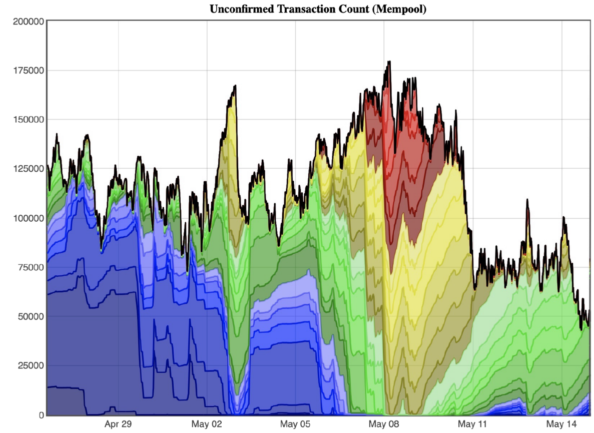

Fig. 1. Mempool occupancy per fee level of the default Mempool.

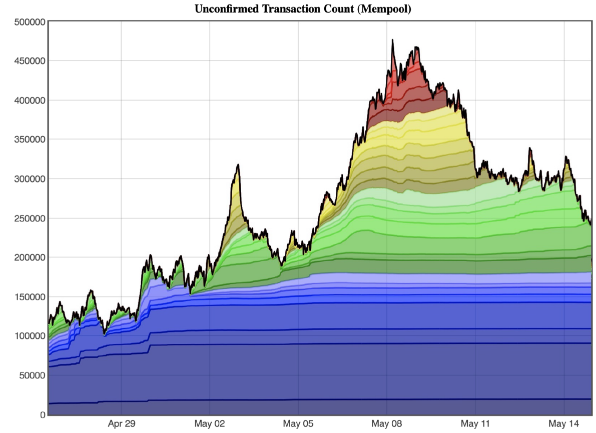

Fig. 2. Mempool occupancy per fee level of the infinite size Mempool.

low-fee transactions (with a fee below 10 sat/B). However, most of those transactions are being effectively excluded from the default Mempool when competing against the transactions with higher fees. This example suggests that the transaction dropping probability may remain significant for long periods, indicating the relevance of determining adequate transaction fees to avoid dropping.

Prior art: There has been substantial previous work on predicting the confirmation of Bitcoin transactions [15], [17], [18]. However, the lack of confirmation considered in most previous works is due to invalid transactions, e.g., suspicious transactions that may relate to double-spending are not confirmed. In this work, in contrast, we focus on a different class of dropping, namely the dropping due to saturation of resources which, to the best of our knowledge, has not been considered so far. We examine the corner case when the system is under heavy load, i.e., the intensity of the arrival process is close or above to the speed of service provided by miners.

Goal: Our goal is to assess the reliability of applications based on blockchains. More precisely, in this study we propose a method to estimate the probability of transactions to be dropped (evicted from the Mempool) during periods of intense competition among pending transactions. Indeed, given that transaction creators are not notified if their transactions are dropped, it is key to understand how fees impact dropping, before transactions are submitted. In essence, transactions with higher fees have a lower probability of being evicted, but high fees do not bode well with the distributed and free nature of blockchains, motivating the following question: *what is the lowest fee that should be offered to ensure a confirmation probability higher than a given threshold?*

Contributions: Our key contributions are twofold.

Model for transaction confirmation: We provide a new, non-trivial, generalization of the Gambler's ruin model to answer our main research question. To the best of our knowledge, this is the first model that focuses on the reliability analysis of PoW blockchains, intended as a measure of the confidence that a user can have about the inclusion of his/her transactions in the ledger. The model is solved with an efficient numerical algorithm that, compared to simulation-based solutions that heavily rely on historical data, allows for the optimization of the trade-off between reliability and running costs of blockchain-based applications with lower computational effort.

Validation using real Bitcoin data: The model is parameterized with publicly known information about the state of the blockchain: the intensity of the transaction arrival process, the distribution of the fees offered by the transactions, the capacity of a block, the block generation rate, and the occupancy of the Mempool at the instant at which a transaction arrives, together with the fee offered by the arriving transaction.

Key observations: The outcomes of our model naturally provide two major takeaways:

- *User-related:* As for a user creating transactions, he/she is not always keen to learn the delay his/her transaction has to experience until it is finally confirmed. Instead, it can be enough for him/her to know that the transaction is eventually accepted with a certain probability. It effectively helps a user to optimize his/her transactions to have small enough fees but be confirmed.
- *System-related:* Our model is a valuable tool for network developers to assess and study the real-time performance metrics of their blockchain systems, providing insights on transaction dropping in the system under heavy load.

Outline: This paper is structured as follows. Section II provides a brief description of PoW-driven blockchains and describes the motivations of the work. Section III introduces the analytical model. In Section IV, we validate our model against historical data. Literature review is provided in Section V. Section VI concludes the paper and proposes future research directions, and Appendix A contains the proof of Theorem 1.

II. Background and Motivation

In this section, we briefly recall the PoW consensus algorithm for public blockchains and introduce the reliability problem that we intend to study.

A. Proof of Work (PoW)

In PoW blockchains, special users called *miners* maintain an entire or partial copy of the blockchain and append new blocks to it. New blocks contain only valid transactions, where the validity is established by the blockchain protocol. For example, in Bitcoin, a valid transaction that moves cryptocurrency must contain the correct spenders' signatures and avoid double-spending. In blockchains with smart contracts, the validation may be more complicated and computationally intensive. In [26], the author describes the Bitcoin PoW. PoW plays a key role in reaching consensus among miners and guaranteeing the immutability of the information stored in the ledger.

In PoW, each miner maintains a memory pool (using Bitcoin terminology, a *Mempool*) where pending transactions are stored. Therefore, when a user submits a new transaction to the system, this is flooded in the peer-to-peer network of miners and they all store it in their local Mempool.

Each miner selects a subset of the transactions in his/her Mempool to form a candidate block. Notice that the blockchain protocol does not enforce a policy regarding the order in which transactions should be picked from the Mempool. Miners typically select transactions based on the ratio of offered fee per byte, and we will account for this behavior in our reliability analysis. All transactions in the candidate block are validated by the miner. Then, the PoW takes place, i.e., the miner is required to perform a certain computation (namely, to *solve a puzzle*) that requires a high computational effort, but that is easy to verify.

The difficulty of the puzzle is dynamically set in such a way that the average block generation delay is constant (e.g., 600 seconds in Bitcoin). This property, together with the maximum block size (e.g., 1 MB for Bitcoin), imposes the maximum theoretical throughput for the blockchain (e.g., approximately 4.5 transactions per second in Bitcoin).

PoW is a memoryless process, i.e., for each miner the probability of solving the puzzle in a certain time slot is independent of how long he/she has been working on it. This has two consequences [29]. First, the time between two consecutive block consolidations is independent and exponentially distributed, since the exponential distribution is the only non-negative continuous distribution that satisfies the memoryless property. Second, the transactions in the candidate block can be changed at any time, because restarting a new puzzle or continuing the old one have the same probability of success.

B. The Auction Among the Transactions

Miners are rewarded for their work in two ways: they obtain a certain amount of cryptocurrency at the block announcement and acceptance (the coinbase transaction in Bitcoin) and they take the fee offered by the transactions included in the block. In order to maximize their profit, miners should order the transactions by offered fee per byte in descending order and include in the block the most valuable ones. This is known as the transaction auction. Notice that, thanks to the memoryless property of the mining process, the candidate block is always updated with the most valuable transactions.

Given that most miners are primarily motivated by profit maximization, and considering the significantly reduced propagation delays within the network when compared against the average mining time, it is reasonable to posit that all miners have access to an approximately identical Mempool. Indeed, in [27] the authors have analyzed the Bitcoin peer-to-peer network, leading to three major findings.[7] First, they found that the vast majority of users adopt the standard Bitcoin agent. Second, a new transaction reaches 90% of miners in 16 seconds. Third, the block propagation delay drops from 20 seconds to reach the entire network to less than 3 seconds to cover 90% of the network. Comparing these timeframes against the average block consolidation time of 600 seconds, it is reasonable to assume homogeneity in Mempools among miners, and this viewpoint finds acceptance within the practitioner community.[8]

C. Transaction Confirmation Process

Recall that the key purpose of miners in the network is to confirm the transactions that they see in the Mempool by gathering them in blocks. However, not every transaction appears in a new block due to the fact that the vast majority of miners strive to maximize their profit. So they often pick the most valuable transactions first. The rest can be left waiting for confirmation for a long time until it is eventually evicted from the Mempool due to the dropping policy that we discuss in the following section.

As a result, users require a methodology to navigate the trade-off between transaction processing costs and confirmation delay. Current optimal fee determination methods employ Monte Carlo simulations and history-based approaches, such as the *'estimatesmartfee'* method first introduced in Bitcoin core version 0.16 [2], [25], as well as some other tools presented online, or by use of an Application Programming Interface (API) [9][10][11][12][13]. The latter tools do not share information about the methodologies that are used for their fee estimations. However, in [21] authors outline the reactive nature of existing fee estimation algorithms, which rely on past statistics and introduce a proactive model that promptly responds to Mempool changes, taking into account the arrival rate of transactions, current Mempool occupancy, and fee distribution of pending transactions. Conversely, our proposed model deals with the trade-off between offered fee and confirmation probability rather than confirmation time.

D. Transaction Dropping Policy

The Mempool occupancy can grow significantly, considering the intrinsic randomness in the block consolidation process and the fact that for a long period of time we may observe an intensity of the arrival process significantly higher than the maximum throughput. To avoid excessive resource consumption at miners'

[7] https://www.dsn.kastel.kit.edu/bitcoin
[8] See, e.g. https://blockchain.com
[9] https://bitcoiner.live
[10] https://blockchain.info
[11] https://BTC.com
[12] https://blockchair.com
[13] https://buybitcoinworldwide.com/fee-calculator

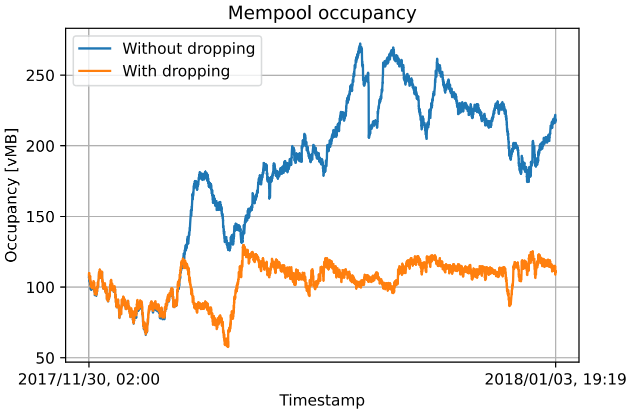

Fig. 3. Comparison of the Mempool occupancy in a system with and without dropping measured in vMB under heavy load conditions.

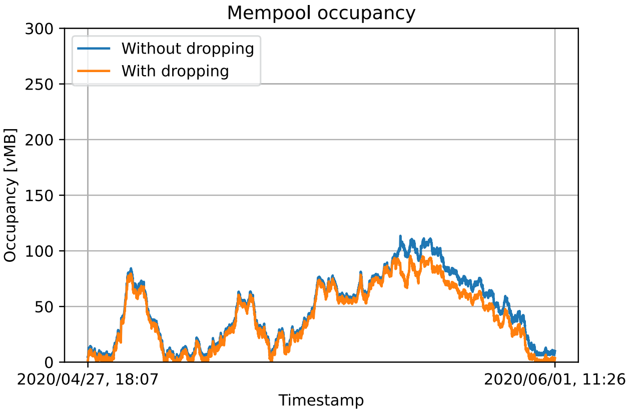

Fig. 4. Comparison of the Mempool occupancy in a system with and without dropping measured in vMB under moderate load conditions.

servers, the implementation of the protocol usually puts some limits on Mempool size. For the standard implementation of Bitcoin this is 300 MB[14]. In addition, Bitcoin transactions that reside in the Mempool for more than 2 weeks are dropped. In the context of this paper, our focus lies primarily on the former dropping mechanism, as the latter typically corresponds to transactions that have no associated fees and thus may not be included in blocks, even if there is available space.

Whenever we need to evict a transaction from the Mempool, the one with the lowest fee per byte is chosen. In the event of multiple transactions having identical fee rates, the oldest transaction takes precedence.

E. The Extent of the Dropping Policy in Real Systems

To understand the extent of the transaction dropping problem, we have monitored the Bitcoin blockchain for the last six years. Recall that, according to the Bitcoin protocol, transaction droppings do not leave traces in the system, in the sense that their identifiers are neither logged in blocks, maintained in some register nor notified to the owners.

Therefore, we have considered a fictitious miner G with an ideally infinite Mempool size. G never drops any transaction and whenever a block is consolidated, if there is some free space, it assumes that its most valuable transactions would have been added, if also all other miners were with an infinite Mempool. Thus, we have monitored the occupancy of the Mempool of G and that of a real miner. We logged the total number of transactions (and their size) grouped in classes of fees. The differences in the occupancy of the real Mempool and that of G show the dropping of the transactions per class of offered fee.

In Fig. 3, we show the occupancy of two Mempools in a situation of very heavy load. We are counting all transactions in each Mempool, regardless of their offered fees. Notice that the monitor logs the virtual MB (vMB) occupied by the transactions as a standard in Bitcoin measurements. In fact, thanks to introduction of the segregation witness (SegWit) extension in the Bitcoin protocol, such transactions occupy only a *third* of the space effectively occupied by a normal transaction, where the precise ratio depends on the transaction characteristics such as the number of addresses that it contains (see, e.g., [5]). We notice that the real Mempool, with droppings, reaches a plateau around 110 vMB, corresponding to the 300 MB of space reserved by the default installation of Bitcoin Core while the fictitious Mempool reaches a size that exceeds 250 vMB, in some moments. Clearly, this situation denotes a heavy dropping activity, i.e., if $M(t)$ and $G(t)$ are the populations of the real and fictitious Mempools, respectively, the dropping rate is given by:

$$d(t) = \frac{\partial}{\partial t}\left(G(t) - M(t)\right) ,$$

where $d(t) < 0$ is the rate at which transactions, which are eventually dropped, would have been included in actual consolidated blocks if the Mempool sizes were unbounded.

Conversely, in Fig. 4, we show a situation of moderate load. We may see that the occupancy of the fictitious Mempool and the real one are almost overlapped with some difference introduced by the peak of high traffic that causes a few droppings.

F. Problem Statement and Engineering Implications

Given the auction governing the transaction confirmations, it is natural to study the trade-off between reliability (i.e., the probability of a transaction being eventually confirmed) and running costs in terms of offered fees. We take a user perspective, i.e., our model accounts for all publicly available information that can help an effective decision: the instantaneous intensity of the transaction arrival process per class of fee per byte, the distribution of the fees, and the occupancy of the Mempool at the instant at which a transaction arrives, together with the fee offered by the arriving transaction. Our main question of interest is: *If a user issues a transaction offering f as fee per byte, what is the probability of being eventually confirmed?* Clearly, the model can be used also to solve the inverse problem, as stated in the introduction: *What is the minimum fee per byte that we should offer to have a confirmation probability higher than a certain threshold?*

[14] https://bitcoin.org/en/bitcoin-core

TABLE I
MODEL NOTATIONS

Variable	Description
K-1	Mempool capacity in number of transactions
B	maximum number of transactions per block
λ	arrival rate of transactions
μ	service rate of pending transactions
α	probability of transaction arrival before next block mining
β	probability that B arrivals occur before block mining
τ	arriving transaction
t	arrival time
i	# of pending transactions found by the arriving transaction
p_i	probability that the arriving transaction is eventually dropped

Answering the above questions is relevant from an engineering standpoint for applications using PoW blockchains and requiring an assessment of the chances that a given transaction will be included in the ledger. Even if the applications do not have constraints on the confirmation time, they may still need to be sure that, up to a certain probability, their transactions will be eventually included in the ledger. This is, for example, the case of applications that use the blockchain to store monitored data collected by IoT systems [8]. Without a proper estimation of the confirmation probability, the application could incur into unnecessary running costs. Moreover, the lack of notification of the transaction droppings would require the application to consult the Mempools to determine if their transactions are still present, and this is often costly or unfeasible. In those cases, the model can be used to estimate the confirmation probability of the transaction already in the Mempool and the cost of issuing another transaction that replaces the former, with a higher confirmation probability. In fact, given the auction mechanism for the transactions, the new more expensive transaction would be confirmed before the older one. If the new transaction spends the same cryptocurrency output as the old one, it immediately invalidates the latter, given the impossibility of double-spending.

III. A GAMBLER'S RUIN BASED MODEL TO ESTIMATE THE DROPPING PROBABILITY

In this section, we present a model for the estimation of the transaction dropping probability given its offered fee per byte and the state of the Mempool. We will formulate the problem as the probability of absorption in a continuous time Markov chain (CTMC).

A. Modeling Assumptions and Notation

In this section, we introduce the model description and the notation that is summarized in Table I. Let $K-1$ be the maximum number of transactions that can be stored in the Mempool. We assume that transaction bids are independent and identically distributed (i.i.d.) random variables characterized by a random variable with Cumulative Density Function (CDF) $F(x)$. In order to accommodate the occurrence of ties, F may not be necessarily continuous but can be càdlàg, $\overline{F}(x) \triangleq 1 - F(x)$. Transactions arrive according to a homogeneous and independent Poisson process with intensity λ and blocks are generated on average every μ^{-1} seconds with an independent exponentially distributed delay. The number of transactions in a block is at most B: if the Mempool contains less than B transactions, the block is generated with all available transactions. The model exploits the memoryless property of the mining process, i.e., if miners are working on a candidate block in which the less valuable transaction offers f_1 and a transaction with a bid higher than f_1 arrives, the latter immediately replaces the cheapest one in the candidate block. The cheapest transaction returns to the Mempool if some space is available, or is evicted otherwise.

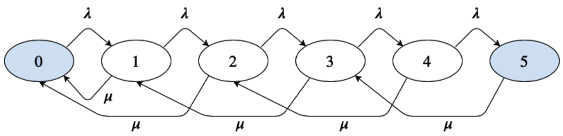

Fig. 5. First toy example. CTMC underlying the model for $B = 2$ and $K = 5$. The blue filled states are transformed into absorbing states characterizing a confirmation or a dropping if the CTMC reaches the leftmost or rightmost states, respectively. Note that the transition from state 5 to 3 is never realized after transforming state 5 into an absorbing state.

Thanks to the independent and exponentially distributed delays, the stochastic process underlying the system is a CTMC. Fig. 5 shows the underlying process for the first toy example system with $B = 2$ and $K = 5$. First, let us consider the case of a transaction τ offering strictly less than all other transactions in the system. Suppose that at its arrival epoch t the Mempool contains i transactions. Then, all the transactions present in the Mempool and those arriving after τ but before the confirmation or eviction of τ will be confirmed before τ. The problem consists in computing the probability that τ is confirmed or evicted. From the perspective of τ, this means that if the CTMC is absorbed in state 0, then τ is confirmed, while if the process is absorbed in state K, then τ is evicted.

Thus, our goal is that of computing the probability of absorption in state 0 or 5 given the initial state i seen at the arrival epoch of τ. We will generalize this reasoning for the situations in which τ makes a general bid in the following subsection.

We introduce $\alpha \triangleq \lambda/(\mu + \lambda)$, i.e., the probability that a transaction arrives before next block consolidation, that is $0 < \alpha < 1$.

Remark 1: While the independent exponentially distributed times between consecutive blocks is determined by the intrinsic memoryless nature of the PoW, the assumption on the Poisson distribution for the arrival process is an abstraction required to have numerical tractability of the model. In Fig. 6, we show the comparison between the distribution of the number of arrivals in a minute monitored for three hours and the Poisson distribution with the same average. While we observe that there is a substantial agreement between the two curves, the real-world arrival process tends to be more noisy. In Section IV, we show that the model maintains a high accuracy in the prediction despite this discrepancy.

B. Model Analysis

Let p_i be the probability that a transaction offering the lowest possible fee per byte is dropped, given that at its arrival epoch the Mempool contains i transactions including itself. We write the system of equations for the probability of absorption in state

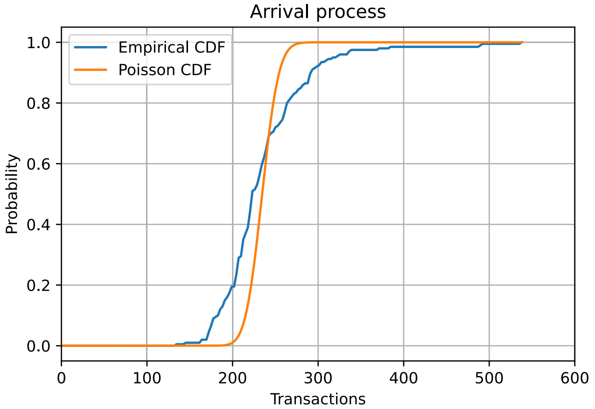

Fig. 6. Distribution of the number of arrivals per minute in the 3 hours time interval starting from 2021/01/19 11PM GMT and its Poisson approximation.

K from state i as follows [14]:

$$\begin{cases} p_0 = 0 & \\ p_i = \alpha\, p_{i+1} & 1 \leq i \leq B \\ p_i = (1-\alpha)\, p_{i-B} + \alpha\, p_{i+1} & B < i < K \\ p_K = 1\,. & \end{cases} \tag{1}$$

Let β denote the probability that B arrivals occur before a block is consolidated,

$$\beta = \alpha^B (1-\alpha). \tag{2}$$

Theorem 1 gives the expression of the probability of eviction for a transaction arriving when the Mempool occupancy is i, as a function of α, β, K, and B. Henceforth, binomial coefficients with negative upper index are assumed to have value 0.

Theorem 1: For $0 \leq i \leq K$, the solution of the system of equations (1) is:

$$p_i = \frac{T_i}{T_K}\,, \tag{3}$$

where

$$T_i = \frac{1}{\alpha^{i-1}} \sum_{l=0}^{m_i} \beta^l \binom{l(B+1)-i}{l} \tag{4}$$

and

$$m_i = \left\lfloor \frac{i-1}{B+1} \right\rfloor.$$

Throughout this paper, $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x, and for $n \in \mathbb{Z}$ and $l > 0$ we use the definition $\binom{n}{l} \triangleq \frac{(-1)^l(-n)_l}{l!}$ with $(n)_l$ being the rising factorial of n, $(n)_l = n(n+1)\cdots(n+l-1)$ [28].

The proof of the above theorem is given in Appendix A. Notice that if we define $T \triangleq T_K$, we can rewrite the system of equations (1) as:

$$\begin{cases} p_0 = 0 & & \\ p_i = \frac{\alpha^{1-i}}{T} & 1 \leq i \leq B & (5) \\ p_i = (1-\alpha)p_{i-B} + \alpha p_{i+1} & B < i < K & (6) \\ p_K = 1\,. & & \end{cases}$$

In practice, Theorem 1 suffers a problem of numerical stability because of the presence of the binomial coefficients that may reach high values at the numerators and denominator. Therefore, we propose a more computationally efficient method based on the theory of difference equations.

We call cases $0 < i \leq B$ *initial conditions*, and $B < i < K$ the *general difference equation*.

Given the general difference equation (6), we can derive the characteristic polynomial [10] by replacing p_i with x^i, for a variable $x \in \mathbb{C}$, $x \neq 0$. Then, divide the equation by x^{i-B} and obtain:

$$P(x) = \alpha x^{B+1} - x^B + (1-\alpha)\,.$$

Notice that $P(x)$ is independent of i. From the roots of $P(x)$, we will derive important properties of our system, as well as an alternative way to compute p_i that does not require the evaluation of large binomial coefficients. To this aim, we need the following lemma.

Lemma 1: If $\alpha \neq B/(B+1)$, all the roots of the characteristic polynomial $P(x)$ are distinct.

Proof: A root r of $P(x)$ has multiplicity higher than one if and only if $P(r) = 0$ and $P'(r) = 0$. We have:

$$P'(x) = \alpha(B+1)x^B - Bx^{B-1}\,.$$

$P'(x)$ has $B-1$ roots in 0 and another one in $B/((B+1)\alpha)$. Clearly, 0 is not a root of $P(x)$ and

$$P\left(\frac{B}{(B+1)\alpha}\right) = -\left(\frac{B}{(B+1)\alpha}\right)^B \left(\frac{1}{B+1}\right) + (1-\alpha)\,.$$

We seek the relation between α and B that makes this quantity equal to 0. This corresponds to finding the real roots of $Q(\alpha)$ in the interval (0,1) with:

$$Q(\alpha) = \alpha^{B+1} - \alpha^B + \left(\frac{B}{B+1}\right)^B \left(\frac{1}{B+1}\right).$$

This polynomial can be factorized as:

$$\begin{aligned} Q(\alpha) = {} & \left(\alpha - \frac{B}{B+1}\right)^2 \\ & \cdot \left(\alpha^{B-1} + \sum_{j=1}^{B-1} \frac{B^{j-1}(B-j)}{(B+1)^j}\alpha^{B-j-1}\right) \\ = {} & \left(\alpha - \frac{B}{B+1}\right)^2 \\ & \cdot \left(\frac{(B+1)^2(\alpha^{B+1}-\alpha^B) + (B+1)\left(\frac{B}{B+1}\right)^B}{(\alpha(B+1)-B)^2}\right). \end{aligned} \tag{7}$$

Notice that the root $B/(B+1)$ has multiplicity 2 but is excluded by the hypothesis of the theorem, and the second factor, as expressed in (7), is a sum of terms whose coefficients are all strictly positive. By Descrates' rule of signs, the second factor does not admit any positive real root.

In conclusion, for $\alpha \neq B/(B+1)$, there cannot be any root of $P'(x)$ that is also a root of $P(x)$. In the following, we will notice that $\alpha = B/(B+1)$ is a critical value for the stability of the system when $K \to \infty$. ■

Henceforth, we assume $\alpha \neq B/(B+1)$. We may study the solution also for this special case for which the general solution (8) does not hold since $P(x)$ has multiple roots in 1, but we omit it for the sake of brevity.

Hence, $P(x)$ admits $B+1$ distinct real or complex roots, namely $\{x_1, \ldots, x_{B+1}\}$. The complex roots come in pairs of conjugate numbers, and one trivial root is 1. Without loss of generality, let us assume $x_1 = 1$.

According to theory of difference equations (see, e.g., [10]), since all roots of $P(x)$ are different by Lemma 1, the solutions can be written as:

$$p_i = \sum_{j=1}^{B+1} C_j^* x_j^i \,, \tag{8}$$

where $C_k^* \in \mathbb{C}$ are coefficients to be determined thanks to the B initial conditions and the case $i = 0$. Thus, we need to solve the system:

$$\begin{cases} C_1^* + C_2^* + \ldots + C_{B+1}^* = 0 & i = 0 \\ C_1^* x_1^i + C_2^* x_2^i + \ldots + C_{B+1}^* x_{B+1}^i = \frac{1}{T}\alpha^{1-i} & 1 \leq i \leq B \,. \end{cases}$$

Let $C_i \triangleq C_i^* T$. Then, we can compute all p_i's as follows:

1) Compute the roots $\{x_1, \ldots, x_{B+1}\}$ of $P(x)$.
2) Solve the following system of linear equations in $\mathbb{C}$:
$$\begin{cases} C_1 + C_2 + \ldots + C_{B+1} = 0 \\ C_1 x_1^i + C_2 x_2^i + \ldots + C_{B+1} x_{B+1}^i = \alpha^{1-i} \; 1 \leq i \leq B. \end{cases} \tag{9}$$
3) To avoid the computation of T with (4), we can use the observation that $p_K = 1$ to write:
$$C_1^* x_1^K + C_2^* x_2^K + \ldots + C_{B+1}^* x_{B+1}^K = 1 \,,$$
and hence, multiplying both hand sides by T:
$$T = C_1 x_1^K + C_2 x_2^K + \ldots + C_{B+1} x_{B+1}^K \,. \tag{10}$$
4) Compute all p_i as:
$$p_i = \frac{\sum_{j=1}^{B+1} C_j x_j^i}{T} = \frac{\sum_{j=1}^{B+1} C_j x_j^i}{\sum_{j=1}^{B+1} C_j x_j^K} \,. \tag{11}$$

C. The Case of Infinite Mempool

In this section, we study the case $K \to \infty$ as in [16]. A misconception may suggest that if $K \to \infty$, then there is no transaction dropping. However, if $\lambda > \mu B$, the intensity of the arrival process is higher than the service capacity. Thus, transactions tend to form a backlog that grows with time and some of them will never be confirmed, irrespectively of K.

In practice, the scenario $K \to \infty$ yields an optimistic model of real systems. In fact, consider a blockchain with Mempool size K and a dropping probability close to the one derived in this section. In this case, it is useless to increase the size of the Mempool with the aim of reducing the dropping probability.

First, notice that the stability condition $\lambda < B\mu$ is equivalent to $\alpha < B/(B+1)$. Theorem 2 below describes what happens to the roots of $P(x)$ when this condition is (not) satisfied.

Theorem 2: The number of roots φ strictly inside the unit disk of $P(x)$ is given by

$$\varphi = \begin{cases} B-1, & \text{if } \alpha \leq B/(B+1), \quad (12) \\ B, & \text{otherwise.} \quad (13) \end{cases}$$

Proof: To prove this result, we resort to [9] [Thm. 2.1] stating that the trinomial $bx^n - ax^m + a - b$ has a number of zeros strictly inside the unit disk equal to $m - \gcd(m, n)$ if $a/b \geq n/m$, and m if $a/b < n/m$. The result immediately follows by the observation that, in $P(x)$, $b = \alpha$, $a = 1$, $n = B+1$, and $m = B$. ■

In our model, we have to consider two cases.

a) Stable system ($\alpha < B/(B+1)$): The model with infinite buffer has an underlying CTMC that will eventually be absorbed in state 0 from every state i with probability 1 since the intensity of the workload is lower than the maximum service capacity. Formally, $P(x)$ has $B-1$ roots strictly inside the unit disk, one is $x_1 = 1$ and let us call the remaining one x_{B+1}. This root must be real, because if it were complex, also its conjugated would be on the perimeter of or outside the unit disk. Moreover, we can also observe that it must lay strictly outside the unit disk because -1 is not a root of $P(x)$ and 1 cannot have multiplicity 2 by Lemma 1. Therefore $T \to \infty$ because all roots strictly inside the unit disk vanish for $K \to \infty$ and $x_{B+1}^K \to \infty$. Since for all finite i, the numerator of (11) is finite, then we conclude that $p_i \to 0$. This means that the probability of not being absorbed in state 0 is 0, as the intuition suggested. Thus, we can write:

$$K \to \infty \wedge \alpha < \frac{B}{B+1} \,, \qquad p_i = 0 \,.$$

b) Unstable system ($\alpha > B/(B+1)$): This is the most interesting case. In fact, while the workload intensity is higher than the maximum service capacity, if i is sufficiently close to 0 we may still have a high probability of being absorbed in state 0. Formally, all roots of $P(x)$ lay strictly inside the unit disk with the exception of $x_1 = 1$. This implies that all the terms of $C_i x_i^K$ vanish for $K \to \infty$ with the exception of x_1, i.e., $T = C_1$. Therefore, we have:

$$K \to \infty \wedge \alpha > \tfrac{B}{B+1} \,, \; p_i = \tfrac{1}{C_1} \sum_{j=1}^{B+1} C_j x_j^i = 1 + \sum_{j=2}^{B+1} \tfrac{C_j}{C_1} x_j^i \,.$$

Indeed, given the B roots of $\alpha x^{1+B} - x^B + (1-\alpha) = 0$ with absolute values strictly less than 1, namely, $x_2, \ldots, x_{B+1}$, it has been shown in [16], [32] that the above expression equals

$$p_i = 1 - \sum_{j=2}^{B+1} x_j^{i+B-1} \prod_{k=2, k\neq j}^{B+1} \frac{1 - x_k}{x_j - x_k} \,, \quad i \geq 1, \tag{14}$$

if $B > 1$, and $p_i = 1 - x_2^i = 1 - ((1-\alpha)/\alpha)^i$ if $B = 1$.

D. Second Toy Example

In order to support the intuition behind the results presented so far, we introduce another toy example. This time let us consider

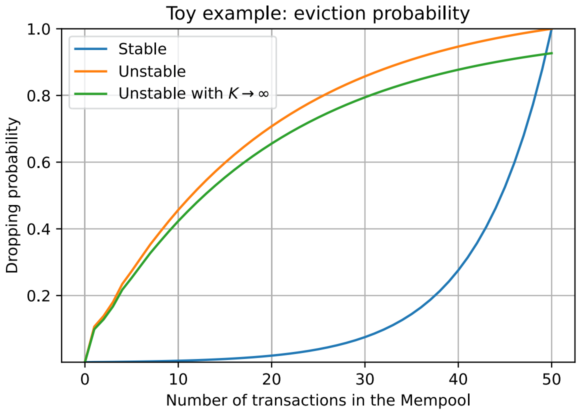

Fig. 7. Second toy example. Comparison of the dropping probabilities for the three cases.

a blockchain in which blocks consist of at most 3 transactions ($B = 3$), the intensity of the arrival process is $\lambda = 1.4$ tx/s and blocks are generated with rate $\mu = 0.6$ blocks/s. The blockchain is able to process $\mu B = 1.8$ tx/s, $\alpha = 1.4/(1.4 + 0.6) = 0.7$ and $B/(B + 1) = 3/4 = 0.75$. Therefore, if the Mempool has infinite capacity $K \to \infty$, we are in the case of a stable system, i.e., the probability of dropping is 0 regardless of the state seen by a transaction at its arrival. If the Mempool capacity is finite, e.g., $K = 50$, we must first find the roots of the characteristic polynomial $P(x)$ that turn out to be:

$$x_1 = 1\,, \qquad x_2 \simeq -0.354 - 0.501j\,,$$
$$x_3 \simeq -0.354 + 0.501j\,, \qquad x_4 \simeq 1.137\,.$$

Notice that, beside $x_1 = 1$ that is common to all possible $P(x)$, we have only one root outside the unit disk, x_4, that is real and positive and two complex conjugate roots inside the unit disk. The next step consists in finding the coefficients by solving the linear system of (9). We obtain:

$$C_1 \simeq -3.500\,, \qquad C_2 \simeq -0.1561 - 0.054889j\,,$$
$$C_3 \simeq -0.156 + 0.05489j\,, \qquad C_4 \simeq 3.812\,.$$

Finally, we compute $T \simeq 2337.29155$ with (10). The probability of dropping given the initial number of transactions found in the Mempool and $K = 50$ is shown in Fig. 7.

Let us assume now $\lambda = 2$ tx/s, and hence $\alpha = 2/(2 + 0.6) \simeq 0.769$, i.e., $\alpha > B/(B + 1)$. In this case, the system with infinite Mempool is unstable. In fact, the roots of the polynomial all lay strictly inside the unit circle, except for $x_1 = 1$. Thus, if $K \to \infty$ we have $T = C_1$ and the dropping probability can be expressed in closed form as a function of the three roots of $(1 - \alpha)x^{-B} + \alpha x = 1$ that lay inside the unit circle, namely $x_2 \simeq 0.950$, $x_3 \simeq -0.325 + 0.459j$, and $x_4 \simeq -0.325 - 0.459j$. Indeed, from [16] it follows (see (14)):

$$p_i = 1 - \sum_{j=2}^{4} x_j^{i+2} \cdot \kappa_j = 1 - \sum_{j=2}^{4} x_j^{i+2} \prod_{k=2,k\neq j}^{4} \frac{1 - x_k}{x_j - x_k}$$

where $\kappa_j = \prod_{k=2,k\neq j}^{4}(1 - x_k)/(x_j - x_k)$, $\kappa_2 \simeq 1.07$, $\kappa_3 \simeq -0.035 + 0.044j$, and $\kappa_4 \simeq -0.035 - 0.044j$.

Fig. 7 shows that the case $K \to \infty$ is a lower bound for the dropping probability for unstable systems. The bound becomes tighter for larger values of K and the result should be used to assess the reliability of the system given a Mempool size with respect to the ideal case.

E. Computational Aspects

The heaviest computational effort for the model solution is the computation of the $B + 1$ roots of $P(x)$. In our implementation, we used the state of the art solution for this problem, i.e., the Aberth's method combined with multiprecision [7] in its implementation MPSolve[15].

Since all roots of $P(x)$ are distinct, the algorithm converges cubically [6] and its parallel version can handle sparse polynomials of degree up to one million, far above our needs.

Finally, the solution of the linear system (9) has an asymptotic complexity of $\mathcal{O}(B^3)$.

F. The Model for Transactions Offering a General Fee

So far, we have reasoned on transactions offering the lowest possible fee, i.e., 0. The model can be easily extended to account for arbitrary fees thanks to the observation that any transaction is insensitive to all transactions offering a fee per byte strictly lower than its own. Let X be the non-negative random variable modeling the fee per byte offered by a transaction. Transaction τ arrives at time t_0 at the blockchain and offers f fee per byte. Intuitively, τ competes only with those transactions with higher priority, i.e., offering a higher fee per byte or the same fee per byte but arriving before t_0. More formally, the perceived arrival process, from the point of view of τ, has intensity $\lambda\mathbb{P}\{X > f\}$.

Summing up, a transaction offering f fee per byte can evaluate its probability of being dropped as follows:

1) Count the number of transactions i_f offering a fee per byte higher or equal to f inside the Mempool at t_0.
2) Compute the intensity of the perceived arrival process $\lambda_f = \lambda\mathbb{P}\{X > f\}$.
3) Use the algorithm presented in Section III-B using λ_f as arrival rate to obtain p_{i_f} that represents the probability of dropping for the transaction.

All information required at steps (1) and (2) is publicly available through web services such as www.blockchain.com.

It is worth noticing that increasing the value of f has two positive effects on the reduction of the dropping probability: the reduction of the number of transactions in the Mempool seen by the new transaction, as well as the decrease of the persisting number of arriving transactions. In the context of a specific transaction, denoted as τ, an increase in its fee-per-byte ratio (f) significantly influences the transaction dynamics on the Mempool. As f increases, transaction τ becomes more favorable in the eyes of the miners, as they are inclined to prioritize it over other transactions with lower fees. Essentially, this prioritization implies that transactions with lower fees become nonexistent, from the standpoint of τ, within the Mempool. This selective perception extends to upcoming lower-fee transactions

[15] https://github.com/robol/MPSolve

entering the Mempool; miners filter them out, focusing primarily on the comparatively higher fees among the incoming transactions. Thus, with increased f, τ typically competes with transactions arriving at a lower frequency. Overall, such an impact favors a reduction of the dropping probability as perceived by τ.

IV. Experiments

In this section, we study the accuracy of the model with respect to the prediction of confirmation for transactions in Bitcoin blockchain.

A. Methodology

The model that we propose can be seen a probabilistic binary classifier [11] that receives the state of the blockchain and the fee per byte offered by a transaction τ and returns the probability for τ to be confirmed (class 0) or dropped (class 1). The classifier cannot be deterministic because of the intrinsic randomness of the blockchain system: the arrival process, the fees offered by the arriving transactions and the random times of block consolidations. We describe the methodology of validations in three steps: (a) analysis of the dataset, (b) parameterization of the model and (c) performance analysis of the probabilistic classifier.

a) Analysis of the dataset: We use a dataset containing the Mempool occupancy in vMB and transaction counts for the last six years of Bitcoin history. We consider two systems: one with infinite Mempool size that never drops transactions, and the standard one of Bitcoin Core. Transactions are clustered in 40 classes based on the fee per byte offered. Class 1 is that with the lowest priority (offering between 0 and 1 satoshis per byte[16]) and class 40 contains the transactions offering more than 2,000 satoshis per byte. The sampling time is of 1 minute.

Let $F_c(t)$ and $M_c(t)$ be the number of transactions at minute t belonging to class c in the infinite and real Mempools, respectively. The difference $D_c(t) = F_c(t) - M_c(t)$ is always non-negative and denotes the number of transactions present in the fictitious Mempool that have been dropped in the real one. At minute t the number of droppings is then $d_c(t) = (D_c(t) - D_c(t-1))^+$, where x^+ is x if $x > 0$ or 0, otherwise. $D_c(t) - D_c(t-1)$ can be negative if at minute t we observe a block confirmation in which the miner with infinite Mempool found some space in the block that could have potentially hosted some transactions of class c that had been previously dropped.

From these data, we infer a transaction dropping or confirmation event as follows. Consider a transaction τ arriving at time t_0 and finding a backlog of N transactions of the same class in front of it, i.e., $N := M_c(t_0)$. All transactions with lower class (offering lower fees) are irrelevant to determine the behavior of τ and are ignored. At each minute $t > t_0$, N is decreased by the number of class c dropped transactions $d_c(t)$ (since the oldest are chosen), or by the number of transactions of class c that entered a block. Assume that at t_1 we have $N < 0$. Then, τ is dropped if at time t_1 we do not have a block consolidation, and is confirmed if it enters a block consolidated at that time.

From the dataset, for each arrival epoch, we compute if a transaction offering a certain fee per byte has been confirmed or not.

b) Parameterization of the model: From the dataset, we obtain the other statistics of interest to configure our model in a trivial way. These are the size of the block B, the maximum capacity of the Mempool in number of transactions $K-1$, the state of the Mempool i_f at the transaction arrival time, and the arrival rate λ_f of transactions offering more than f fee per byte (see Section III-F). While the former two parameters are stable for long periods, the latter two change for each considered transaction. For all our experiments, we have considered $B = 2,100$ transactions and $K = 180,000$.

c) Performance analysis of the probabilistic classifier: In order to validate our model, we resort to the computation of Brier Score (see, e.g., [12]). This approach is known to be a simple way to assess and compare forecasting outcomes of models. Given a set of R observations obtained from the dataset $O = (o_1, o_2, \ldots, o_R)$ (where $o_i = 0$ if the transaction is confirmed and $o_i = 1$, otherwise) and the corresponding probabilities of dropping estimated by the model $Q = (q_1, q_2, \ldots, q_R)$, Brier score can be computed as:

$$BS_{\text{model}} = \frac{1}{R}\sum_{i=1}^{R}(q_i - o_i)^2 ,$$

that can be interpreted as an averaged mean square error of all forecasts. For instance, if we estimate that a transaction will be dropped with a probability close to 1 and it is actually dropped in the real system, then the Brier Score is 0, that is the best score achievable. Otherwise, if it is not dropped, then the Brier Score is 1, the worst score achievable.

Then, we consider two simple probabilistic predictors as reference models. Rand is a simple *random predictor* without any knowledge of the system representing the predictor with no skill, i.e.:

$$BS_{\text{rand}} = \frac{1}{R}\sum_{i=1}^{R}(q_{rnd} - o_i)^2 ,$$

where $q_{rnd} = rnd(0,1)$, i.e., a random number between 0 and 1.

On the other hand, Oracle is an ideal predictor that knows a priori the fraction of transactions of class c that will be dropped and assigns this value as the probability of dropping to all transactions of class c, i.e.:

$$BS_{\text{oracle}} = \frac{1}{R}\sum_{i=1}^{R}(\bar{o} - o_i)^2 ,$$

where $\bar{o} = \frac{1}{R}\sum_{i=1}^{R} o_i$.

Although the Oracle predictor is aware of the total dropping probability, it does not know which transactions are dropped and applies this value for all of them. Notice that Oracle is not implementable in practice since it uses information available a posteriori, but can be approximated by assuming that, for sufficiently long periods, in case of similar workloads, the fraction of

[16] 1 satoshi is 10^{-8} bitcoin, equivalent to 0.0002642 USD at the time of writing.

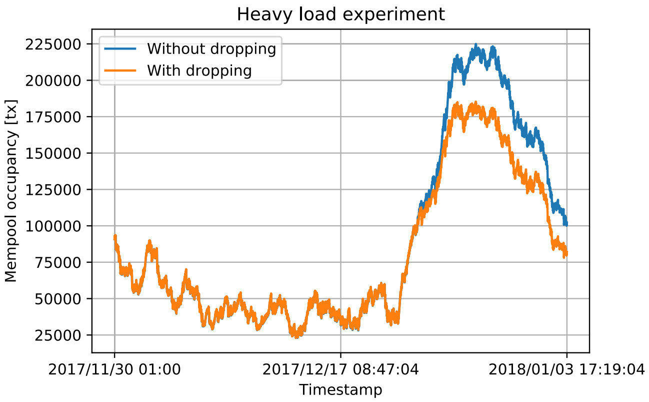

Fig. 8. Mempool occupancy in a system with and without dropping measured in number of transactions under heavy load.

transactions offering a certain fee that are dropped does not vary much. Therefore, the dropping probability used by Oracle could be inferred by historical data. Next, since Oracle outperforms Rand the former will be the term of comparison for our model.

To compare our score with the reference one we use the corresponding skill score metric, namely Brier Skill Score (BSS) that in our scenario is defined as:

$$BSS = 1 - \frac{BS_{\text{model}}}{BS_{\text{oracle}}}.$$

Positive values indicate superior performance in comparison to the reference baseline, where a score of 1 represents the optimal performance, while scores of 0 or below signify poorer performance.

We consider two scenarios for our test: *heavy load*, where the arrival intensity is higher than the maximum service capacity and *moderate load*, where the stability is satisfied but we are close to the saturation point.

B. Heavy Load Conditions

During heavy load periods, we observe many droppings of the cheapest transactions. If we consider a class with very low fee, the experiment would show 100% of dropping probability, with a perfect accuracy of our model.

To make the scenario more challenging for the model we study the class of transactions offering between 1 and 12 satoshis per byte. From the dataset, we collected 100 samples uniformly distributed in the time interval between 2017/11/30 and 2018/01/03 according to the methodology described in Section IV-A.

Fig. 8 shows the occupancy of Bitcoin Mempool during the defined days. We notice that the populations at the infinite and finite Mempools are basically overlapped at the beginning of the observation period, but then a sudden increase in the traffic intensity leads to a high number of droppings.

Table II shows the results of this experiment. In heavy load, around 60% of the transactions in the considered time interval are dropped despite their offered fee. We may notice that the model that we propose outperforms both Rand, the classifier with no skill, and Oracle which is based on the assumption of the knowledge of the probability of dropping for the dataset.

TABLE II
BRIER SCORES FOR HEAVY AND MODERATE LOADS

	Heavy load	Moderate load
Transaction class	$[1, 12] sat/B$	$[1, 5] sat/B$
Fraction confirmed	0.39	0.64
Fraction dropped	0.61	0.36
BS_{model}	0.134	0.161
BS_{rand}	0.465	0.431
BS_{oracle}	0.242	0.232
BSS	**0.447**	**0.306**

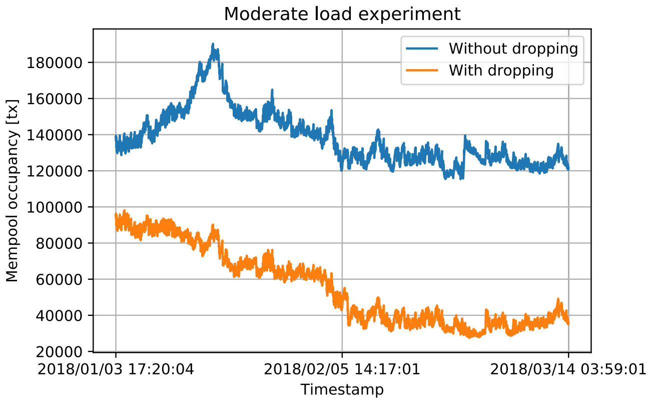

Fig. 9. Mempool occupancy in a system with and without dropping measured in number of transactions under moderate load.

C. Moderate Load Conditions

To study the moderate load condition, we consider a cheaper class of transactions, i.e., that includes transactions offering fees between 1 and 5 satoshis per byte. Fig. 9 shows the traces of the Mempool occupation during the time interval between 2018/01/03 and 2018/03/14, i.e., immediately following the heavy load condition previously studied. In Fig. 9 the two lines do not cross each other. In particular, in the latter portion of the trace, which includes the months of February and March 2018, the infinite Mempool trace is roughly a vertical translation of the finite Mempool trace, indicating a relatively low occurrence of droppings within this timeframe (differently from Fig. 8).

Table II shows the results of this experiment. Although there seems to be not much difference in the number of droppings between moderate load and heavy load, the reader should consider that we are studying a class of cheaper transactions. We may notice, also in this case, that the model that we propose outperforms the two reference classifiers.

D. Reliability Analysis as Function of the Mempool State

Reliability becomes crucial especially when the perceived arrival intensity λ_f is higher than the maximum system service capacity $B\mu$. Let $\rho = \lambda_f/(B\mu)$ be the perceived load factor of the system. Recall that, by increasing the offered fee per byte f, the perceived arrival rate decreases and hence also the perceived load factor. Fig. 10 shows the impact of this fee modulation as function of the occupancy of the Mempool. The vertical lines show the first two multiples of the block capacity. It is interesting to observe that the dropping probability appears to be convex only when the transaction may be included in the first block. If this is impossible, then the dropping probability becomes concave.

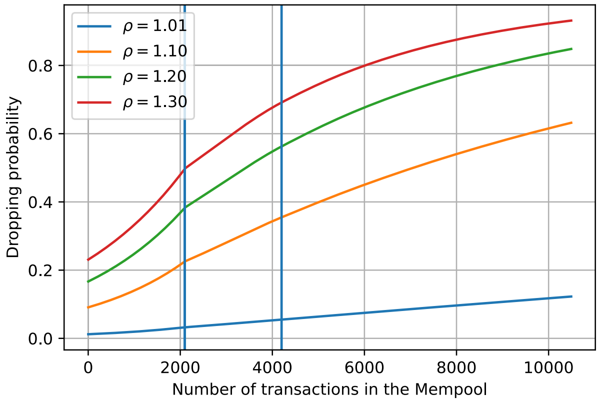

Fig. 10. Dropping probability for different load factors.

V. Related Work

The quantitative analysis of PoW blockchains has attracted significant interest from researches and practitioners [4], [15], [17], [30]. However, most of the modeling efforts have been devoted to the prediction of transaction confirmation times given a certain offered fee per byte, while reliability analysis has remained an open problem.

In the field of performance analysis of blockchains, queueing models have been widely applied. In [15], [17], the authors investigate the queueing process underlying the Mempool with attention to the relation between the fee per byte offered by a transaction and its expected confirmation time. The resulting model is a queue with a scheduling of strict priorities based on the outcomes of the auction run by miners. In [4], the authors refine the model by considering a more accurate block creation policy. [21] introduces a simple analytical model that is based on key Mempool characteristics such as occupancy, transaction generation rate, and fee distribution, and estimates the optimal transaction fee for a given confirmation delay.

In a similar vein, [30] combines machine learning and queuing theory to study the confirmation process of transactions. Specifically, this allows them to study delays in transaction confirmation in blockchains. In contrast to the aforementioned models, which utilize queueing theory to infer delays, our study introduces a ruin model to assess the probability of transaction drops.

Game theory has also been used to study the strategic behavior among miners and users. A game theoretical framework of the auction mechanism behind PoW systems, relating it with confirmation delays, is proposed in [19]. In [3] the authors consider a game between miners that need to decide how to spend their resources across multiple ledgers. In this work, in contrast, we consider a single ledger, wherein users may strategically determine transaction fees to achieve the desired level of reliability and miners prioritize the confirmation of transactions with higher fees.

In [22], [23], the authors study the behavior of Bitcoin in periods of heavy load. In particular, they explore the unfair behavior of certain mining pools that may violate the blockchain neutrality. Instead of following the fee per byte auction outcome, they use alternative approaches, e.g., confirming transactions that are of their selfish or vested interest, or receiving dark-fee payments via opaque (non-public) side-channels. While those perturbations impact delay sensitive transactions, we envision that they do not significantly change the perspective of delay insensitive transactions that are mostly interested into the confirmation probability rather than their delays, as considered in this work. A detailed analysis of the implications of those perturbations on the outcome of the proposed model is out of our scope, and is left as subject for future work.

In [24] the authors concentrate their efforts on analyzing the Bitcoin Mempool, aiming to investigate the correlation between Mempool dynamics and subsequent surges in trading volumes. This investigation is rooted in projecting prior cash flow growth onto Mempool expansion. As a result, they state that when employed as a price indicator, the Mempool exhibits mixed outcomes, with a predominant presence of uncertainty in fluctuation of price.

Concerning the challenge of data sharing within IoT device networks, the authors of [31] suggest a novel feeless transaction processing mechanism. Their approach aims to serve as an alternative to cryptocurrency-oriented blockchains. Despite the fact that in this work we focus on cryptocurrency-oriented blockchains, we envision that alternative technologies, such as those proposed in [31], will still benefit from the model proposed here, given the persistence of finite Mempools in various blockchain systems.

In [33], the authors use the mean time to ruin of the Cramér-Lundberg model to evaluate the confirmation time in the Bitcoin network. While computing the mean time to ruin, the authors take into account the Mempool state at the instant of the arrival of a transaction. The Cramér-Lundberg model is also at the base of the analysis proposed in [18] to estimate the probability that a transaction with a certain fee is confirmed before a certain number of blocks. The author derives bounds for this metric using a diffusion approximation. Our work differs from [18], [33] in a number of aspects. First, the Cramér-Lundberg model assumes a constant flow of arrivals while we consider a random arrival process. Second, our metric of interest is the confirmation probability, accounting for a finite Mempool, while [18], [33] consider the confirmation time, measured in blocks [18] or seconds [33].

Finally, and most importantly, we should notice that all models mentioned so far assume an infinite Mempool size and hence can be used only in condition of stability, i.e., when the intensity of the arrivals is lower than the service capacity. This is not always the case for important blockchains like Bitcoin that can experience long periods of heavy load. To the best of our knowledge, the model proposed in this paper is the first considering a finite Mempool and hence capable of studying the heavy load conditions that cause transaction evictions.

VI. Conclusion

In this article, we have proposed a model to predict the confirmation or dropping probabilities of transactions in PoW blockchains. The model exploits the auction-based mechanism underlying the confirmation process of these blockchains to derive the dropping probabilities from the offered fees. The analysis relies on the theory of difference equations and its main step is the computation of the roots of a certain characteristic polynomial.

The advantage of this white box model relies on the fact that no historical data are necessary to train the model. The intensive use of historical data for training purposes is a major drawback of machine learning models applied in this context, since dropped transactions do not leave traces in the blockchain logs. Despite the fact that for Bitcoin we have a dataset that allows us to infer these events, for other blockchains analogous historical data are unavailable.

To the best of our knowledge, reliability analysis of the transaction confirmation process in PoW blockchains is a novel aspect in the field of blockchain studies. However, the increasing popularity of these distributed ledgers combined with their limited throughput, due to their intrinsic security design, results in a delicate balance between reducing the application running costs and maintaining a certain level of reliability. Our contribution is a first step toward an automatic optimization of this trade-off. Furthermore, we demonstrate that in contrast to random and oracle predictors, our model exhibits greater precision in its outcomes. Moreover, it can be used to estimate the likelihood of transaction droppings resulting from heightened transaction competition in the Mempool, offering valuable insights for both end users and network developers. While the former can effectively optimize their costs of eventual confirmation when the confirmation time is not a priority, for the latter, our model can be a useful tool to evaluate the dropping probability for the system under heavy load. Therefore, the model may support a fee estimation tool for delay-tolerant transactions whose interest is just that of being confirmed.

Future works include and integration of this model with workload predictors (see, e.g., [20]) to allow for long term classification of transactions. Moreover, we plan to investigate models with more general arrival processes.

Appendix A
Proof of Theorem 1

Clearly, $p_0 = \frac{T_0}{T_K} = 0$ and $p_K = \frac{T_K}{T_K} = 1$. We first consider $1 \leq i \leq B$. In this case, since $m_j = \left\lfloor \dfrac{j-1}{B+1} \right\rfloor = 0$ for $1 \leq j \leq B+1$, we have

$$p_i = \tfrac{T_i}{T_K} = \frac{\frac{1}{\alpha^{i-1}} \sum_{l=0}^{m_i} \beta^l \binom{l(B+1)-i}{l}}{T_K} = \frac{\dfrac{1}{\alpha^{i-1}}}{T_K} = \frac{\alpha \dfrac{1}{\alpha^i}}{T_K} = \alpha\, p_{i+1}.$$

Let us now consider the general case $1 \leq i \leq K$. Let $Z(B+1) < i \leq (Z+1)(B+1)$ for some $Z \geq 0$, i.e., $Z = m_i$. We prove that (3) is a solution of equation $p_i = (1-\alpha)\, p_{i-B} + \alpha\, p_{i+1}$. Indeed, if $p_i = T_i/T_K$, we have:

$$\begin{aligned}
&(1-\alpha)\, T_K\, p_{i-B} + \alpha\, T_K\, p_{i+1} \\
&\quad = (1-\alpha) \frac{1}{\alpha^{i-B-1}} \sum_{l=0}^{Z-1} \beta^l \binom{l(B+1)-i+B}{l} \\
&\qquad + \alpha \frac{1}{\alpha^i} \sum_{l=0}^{Z} \beta^l \binom{l(B+1)-i-1}{l}.
\end{aligned}$$

After algebraic manipulation, as $Z \geq 0$,

$$\begin{aligned}
&\frac{1}{\alpha^{i-1}} \sum_{l=0}^{Z} \beta^l \binom{l(B+1)-i-1}{l} \\
&\quad = \frac{1}{\alpha^{i-1}} \sum_{l=1}^{Z} \beta^l \binom{l(B+1)-i-1}{l} + \frac{1}{\alpha^{i-1}} \\
&\quad = \frac{1-\alpha}{\alpha^{i-B-1}} \sum_{l=0}^{Z-1} \beta^l \binom{l(B+1)-i+B}{l+1} + \frac{1}{\alpha^{i-1}}.
\end{aligned}$$

Recall that $(n)_l$ denotes the rising factorial of n, $(n)_l = n(n+1)\cdots(n+l-1)$ and $\binom{n}{l} \triangleq \frac{(-1)^l(-n)_l}{l!}$. Then, noting that $\beta = \alpha^B(1-\alpha)$,

$$(1-\alpha)\, T_K\, p_{i-B} + \alpha\, T_K\, p_{i+1} \tag{15}$$

$$\begin{aligned}
&= \frac{1-\alpha}{\alpha^{i-B-1}} \left(\sum_{l=0}^{Z-1} \beta^l\, (-1)^l \cdot \left(\frac{(i-l(B+1)-B)_l}{l!} \right.\right. \\
&\quad \left.\left. - \frac{(i-l(B+1)-B)_{l+1}}{(l+1)!} \right) \right) + \frac{1}{\alpha^{i-1}}
\end{aligned} \tag{16}$$

$$\begin{aligned}
&= \frac{1-\alpha}{\alpha^{i-B-1}} \left(\sum_{l=0}^{Z-1} \beta^l\, (-1)^l\ \frac{(i-l(B+1)-B)_l}{l!} \right. \\
&\quad \left. \cdot \frac{l(B+1)-i+B+1}{l+1} \right) + \frac{1}{\alpha^{i-1}}
\end{aligned} \tag{17}$$

$$\begin{aligned}
&= \frac{1}{\alpha^{i-1}} \left(\sum_{l=0}^{Z-1} \beta^{l+1}\, (-1)^{l+1}\ \frac{(i-l(B+1)-B)_l}{l!} \right. \\
&\quad \left. \cdot \frac{i-(l+1)(B+1)}{l+1} \right) + \frac{1}{\alpha^{i-1}}
\end{aligned} \tag{18}$$

$$= \frac{1}{\alpha^{i-1}} \sum_{l=0}^{Z} \beta^l \binom{l(B+1)-i}{l} = T_i. \tag{19}$$

This proves the statement since, dividing both sides by T_K, we obtain $T_i/T_K = p_i$. ∎

References

[1] Bitcoin's, "'BRC-20' explosion sends users scrambling for options, including lightning," 2023. Accessed: Aug. 29, 2023. [Online]. Available: https://coindesk.com/tech/2023/05/08/bitcoins-brc-20-explosion-sends-users-scrambling-for-options-including-lightning

[2] "'Estimatesmartfee' method documentation," Accessed: Nov. 15, 2023. [Online]. Available: https://bitcoincore.org/en/doc/0.16.0/rpc/util/estimatesmartfee

[3] E. Altman et al., "Blockchain competition between miners: A game theoretic perspective," *Front. Blockchain*, vol. 2, 2019, Art. no. 26.

[4] S. Balsamo, A. Marin, I. Mitrani, and N. Rebagliati, "Prediction of the consolidation delay in blockchain-based applications," in *Proc. ACM/SPEC Int. Conf. Perform. Eng.*, 2021, pp. 81–92.

[5] M. Basile, G. Nardini, P. Perazzo, and G. Dini, "Segwit extension and improvement of the blocksim bitcoin simulator," in *Proc. IEEE Int. Conf. Blockchain*, 2022, pp. 115–123.

[6] D. A. Bini, "Numerical computation of polynomial zeros by means of Aberth's method," *Numer. Algorithms*, vol. 13, no. 2, pp. 179–200, 1996.

[7] D. A. Bini and L. Robol, “Solving secular and polynomial equations: A multiprecision algorithm,” *J. Comput. Appl. Math.*, vol. 272, pp. 276–292, 2014.

[8] J. Cui, N. Liu, Q. Zhang, D. He, C. Gu, and H. Zhong, “Efficient and anonymous cross-domain authentication for iiot based on blockchain,” *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 2, pp. 899–910, Mar./Apr. 2022.

[9] K. Dilcher, J. D. Nulton, and K. B. Stolarsky, “The zeros of a certain family of trinomials,” *Glasgow Math. J.*, vol. 34, pp. 55–74, 1992.

[10] S. Goldberg. *Introduction to Difference Equations*. Mineola, NY, USA: Dover Pub., 1986.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, 2nd ed. Berlin, Germany: Springer, 2009.

[12] J. Hernandez-Orallo, P. A. Flach, and C. Ferri, “A unified view of performance metrics: Translating threshold choice into expected classification loss,” *J. Mach. Learn. Res.*, vol. 13, pp. 2813–2869, 2012.

[13] J. Hoenicke, “Johoe bitcoin mempool,” Accessed: Aug. 29, 2023. [Online]. Available: https://jochen-hoenicke.de/queue/

[14] S. Karlin and H. M. Taylor, *A First Course in Stochastic Processes*. 2nd ed. New York, NY, USA: Academic Press, 1968.

[15] S. Kasahara and J. Kawahara, “Effect of bitcoin fee on transaction-confirmation process,” *J. Ind. Manage. Optim.*, vol. 15, no. 1, 2019, Art. no. 365.

[16] G. Katriel, “Gambler’s ruin probability - a general formula,” *Statist. Probability Lett.*, vol. 83, no. 10, pp. 2205–2210, 2013.

[17] Y. Kawase and S. Kasahara, “Priority queueing analysis of transaction-confirmation time for bitcoin,” *J. Ind. Manage. Optim.*, vol. 16, no. 3, pp. 1077–1098, 2020.

[18] R. Gündlach, M. Gijsbers, D. T. Koops, and J. Resing, “Predicting confirmation times of Bitcoin transactions,” *SIGMETRICS Perform. Eval. Rev.*, vol. 48, no. 4, pp. 16–19, 2021.

[19] J. Li, Y. Yuan, S. Wang, and F.-Y. Wang, “Transaction queuing game in Bitcoin blockchain,” in *Proc. IEEE Intell. Veh. Symp.*, 2018, pp. 114–119.

[20] I. Malakhov, C. Gaetan, A. Marin, and S. Rossi, “Workload prediction in BTC blockchain and application to the confirmation time estimation,” in *Proc. Eur. Workshop Perform. Eng.*, 2021, pp. 3–21.

[21] I. Malakhov, A. Marin, and S. Rossi, “Analysis of the confirmation time in proof-of-work blockchains,” *Future Gener. Comput. Syst.*, vol. 147, pp. 275–291, 2023.

[22] J. Messias, M. Alzayat, B. Chandrasekaran, and K. P. Gummadi, “On blockchain commit times: An analysis of how miners choose bitcoin transactions,” in *Proc. 2nd Int. Workshop Smart Data Blockchain Distrib. Ledger*, 2020, pp. 1–9.

[23] J. Messias, M. Alzayat, B. Chandrasekaran, K. P. Gummadi, P. Loiseau, and A. Mislove, “Selfish & opaque transaction ordering in the bitcoin blockchain: The case for chain neutrality,” in *Proc. 21st ACM Internet Meas. Conf.*, 2021, pp. 320–335.

[24] A. Mikhaylov, H. Dinçer, S. Yüksel, G. Pinter, and Z. A. Shaikh, “Bitcoin mempool growth and trading volumes: Integrated approach based on qrof multi-swara and aggregation operators,” *J. Innov. Knowl.*, vol. 8, no. 3, 2023, Art. no. 100378.

[25] A. Morcos, “High level description bitcoin core’s fee estimation algorithm,” 2017. Accessed: Nov. 15, 2023. [Online]. Available: https://gist.github.com/morcos/d3637f015bc4e607e1fd10d8351e9f41

[26] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” *Decentralized Bus. Rev.*, 2008.

[27] T. Neudecker, P. Andelfinger, and H. Hartenstein, “Timing analysis for inferring the topology of the bitcoin peer-to-peer network,” in *Proc. 13th IEEE Int. Conf. Adv. Trusted Comput.*, 2016, pp. 358–367.

[28] F. W. J. Olver, D. W. Lozier, R. F. Boisvert, and C. W. Clark, *The NIST Handbook of Mathematical Functions*. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[29] J. P. Moyano, K. Schmedders, and G. P. Reich, “Urns filled with bitcoins: New perspectives on proof-of-work mining,” *SSRN Electron. J.*, Univ. Zurich, 2019, Art. no. 3399742, doi: 10.2139/ssrn.3399742.

[30] S. Ricci, E. Ferreira, D. S. Menasche, A. Ziviani, J. E. Souza, and A. B. Vieira, “Learning blockchain delays: A queueing theory approach,” *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 46, no. 3, pp. 122–125, 2019.

[31] F. Shamieh, X. Wang, and A. R. Hussein, “Transaction throughput provisioning technique for blockchain-based industrial IoT networks,” *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 3122–3134, Oct.–Dec. 2020.

[32] H. L. Skala, “An aspect of the gambler’s ruin problem,” *Int. J. Math. Educ. Sci. Technol.*, vol. 22, no. 1, pp. 51–56, 1991.

[33] I. Stoepker, R. Gündlach, and S. Kapodistria, “Robustness analysis of bitcoin confirmation times,” *SIGMETRICS Perform. Eval. Rev.*, vol. 48, no. 4, pp. 20–23, 2021.

Ivan Malakhov received the Ph.D. degree in computer science from Università Ca’ Foscari Venezia, Venice, Italy, in 2023. He is currently a Postdoctoral Researcher with the same university. In 2022, he has also been a Visiting Researcher with Newcastle University, Newcastle upon Tyne, U.K. His research interests include modeling, analysis and performance evaluation of computer systems, in particular blockchain networks under various consensus mechanisms.

Andrea Marin (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Venice, Venice, Italy, in 2009. He is an Associate Professor of computer science with the same university. He is the co-author of more than 100 technical papers in refereed international journals and conference proceedings. His current research interests include the performance and reliability evaluation of computer systems using stochastic modeling techniques.

Sabina Rossi received the Ph.D. degree in computational mathematics and informatics from the University of Padova, Padua, Italy, in 1994. She is currently a Full Professor of computer science with the University Ca’ Foscari of Venice, Venice, Italy. She has been a Visiting Professor with Universitè Paris 7, Paris, France, in 2007, and a Research Fellow with the Universitè Catholique de Louvain-la-Neuve, Ottignies-Louvain-la-Neuve, Belgium, in 1997. She is the co-author of more than 100 technical papers in refereed international journals and conference proceedings. Her current research interests include development of formal tools for the analysis and verification based on process algebraic techniques and, specifically, on stochastic process algebras.

Daniel Sadoc Menasché received the Ph.D. degree in computer science from the University of Massachusetts, Amherst, MA, USA, in 2011. He is currently an Associate Professor with the Institute of Computing, Federal University of Rio de Janeiro, Brazil. His research interests include modeling, analysis and performance evaluation of computer systems, being a recipient of best paper awards at GLOBECOM 2007, CoNEXT 2009, INFOCOM 2013 and ICGSE 2015. He is an alumnus affiliated member of the Brazilian Academy of Sciences.

Open Access funding provided by ‘Università Ca’ Foscari Venezia’ within the CRUI CARE Agreement