Journal of Network and Systems Management (2023) 31:22
https://doi.org/10.1007/s10922-022-09717-w

Transaction Latency Within Permissionless Blockchains: Analysis, Improvement, and Security Considerations

Maher Alharby[1]

Received: 15 June 2022 / Revised: 23 November 2022 / Accepted: 26 December 2022

Abstract
Permissionless blockchains such as Ethereum offer decentralization and security. However, their performance is low compared to permissioned and traditional centralized systems, which may hinder their adoption. Transaction latency is an important performance metric that impacts the use of blockchain systems. That is, there is a need for a systematic evaluation to analyze the factors that may contribute to transaction latency. In this article, we propose a queueing model and conduct extensive simulation experiments to evaluate transaction latency within the Ethereum blockchain, considering all the contributing factors. Our simulation results suggest that transaction fees, block limit, block interval time, transaction arrival rate, and the behavior of the network nodes can all significantly contribute to the latency figures. We find the behavior of the network nodes can significantly impact the waiting time for transactions, especially since nodes are not enforced to utilize their blocks properly in the current implementation of permissionless blockchains. Finally, we suggest improvements to transaction latency by (1) increasing the block limit, (2) reducing the block interval time, and (3) encouraging nodes to fill their blocks with transactions as well as discussing the relevant security issues.

Keywords Permissionless blockchains · Ethereum · Performance · Transaction latency · Security

✉ Maher Alharby
mharby@taibahu.edu.sa

[1] Department of Computer Science, College of Computer Science and Engineering, Taibah University, Medina, Saudi Arabia

Published online: 31 December 2022

1 Introduction

Blockchain can be considered as a distributed ledger that is shared among different nodes in the network. The blockchain ledger is organized into blocks of transactions, and it is maintained by a set of nodes, referred to as miners, who continuously append new blocks of transactions. Blockchain systems can be categorized as permissionless and permissioned. Compared to permissioned blockchains, permissionless blockchain systems (e.g., Bitcoin and Ethereum) have various advantages such as decentralization and security, but their performance is low. For instance, the Ethereum network can only process about 14 transactions per second [1], while permissioned blockchains and centralized systems can process hundreds or even thousands of transactions per second [2].

There is a wide range of permissionless blockchain systems to choose from when building decentralized applications. Ethereum is currently the most common system for decentralized applications due to its smart contract engine [3]. There are currently over 3500 decentralized applications running on the Ethereum network [4]. Transaction latency is a key performance metric that impacts the selection and the use of a blockchain system, as large latency figures can hinder the adoption of such systems. There are different factors and system properties that may contribute to transaction latency. Evaluating those factors and their impact on the waiting time for transactions is essential.

In this article, we conduct an extensive simulation study to evaluate the performance of permissionless blockchains in terms of transaction latency. We consider various factors and system properties that may contribute to transaction latency, including transaction fees, block limit, block interval time, transaction arrival rate as well as the behavior of the network nodes concerning block utilization. We consider the Ethereum network as a case study for our performance evaluation. This analysis study not only sheds light on the factors that impact transaction latency but also identifies and proposes different approaches to improve transaction latency. To that end, we propose to improve transaction latency by (1) increasing the block limit or size (2) reducing the block interval time, and (3) encouraging nodes to fill their blocks with transactions as well as discussing the security considerations for those approaches.

To obtain valuable insights about transaction latency, we propose a queueing model that can be used to simulate and measure the latency of Ethereum transactions taking into consideration different factors and parameters that may contribute to the latency results. Using our proposed queueing model, we run extensive simulation experiments of different scenarios since it is not possible to derive the latency results solely by observing the real Ethereum network. To obtain representative results, we augment our model with real data that we capture from the Ethereum network.

The contributions of this work can be summarized as follows:

1. We identify all the factors and parameters that can affect transaction latency in permissionless blockchains. This is to understand the latency bottleneck problem.

We are not aware of any work that explicitly identifies the contributing factors as we did in this article.
2. We propose a queueing model that simulates and measures the latency for Ethereum transactions taking into consideration different factors that may potentially contribute to the latency results. We note that our model can be applied to permissionless blockchains in general, and it is not limited to the Ethereum blockchain.
3. We conduct extensive simulation experiments with different configurations of simulation parameters related to blocks, transactions, nodes, and other parameters. This is to understand how current and future configurations may impact transaction latency.
4. We propose different approaches to enhance transaction latency by (1) increasing the block limit or size (2) reducing the block interval time and (3) encouraging nodes to fill their blocks with transactions. These approaches can be easily adopted and implemented, but they may trigger security issues. That is, we discuss the security issues that need to be considered when applying those approaches.

The main insights and results of our latency evaluation are as follows. We find that all the contributing factors we identified can have an impact on transaction latency. Also, we observe the behavior of the network nodes can significantly impact transaction latency. When nodes fill their blocks with transactions, the overall waiting time for transactions is expected to decrease and vice versa. Within the current permissionless blockchains, nodes are not enforced to utilize their blocks properly, making transaction latency dependent on the behavior of network nodes.

Our analysis also suggests that the waiting time for transactions is expected to increase with the increase of both the transaction arrival rate and the block interval time. Increasing the block interval time can result in a high waiting time for transactions as fewer blocks can be created in the network. On the contrary, the average waiting time for transactions is expected to decrease with the increase of the block limit or when nodes utilize their blocks properly. This is because more transactions can be processed. Second, transaction fees are an important factor that can impact the waiting time for transactions. This is because the network nodes often prioritize transactions by selecting the ones that offer high fees. However, we found that the waiting time for transactions with a specific transaction fee can vary depending on other contributing factors such as transaction arrival rate and block limit.

The structure of this article is organized as follows. Section 2 provides an overview of the Ethereum blockchain, including the key concepts and the life cycle of Ethereum transactions. Section 3 identifies and discusses all the factors and system properties that can contribute to transaction latency in blockchain systems. Related work is discussed in Sect. 4. In Sect. 5, we propose a queueing model for transaction latency estimation. Section 6 describes the simulation setup and presents the main results and insights about transaction latency. We discuss the validity threats to our work and propose different approaches to improve transaction latency in Sect. 7. Finally, we conclude the article in Sect. 8.

2 Overview of the Ethereum Blockchain

Ethereum [5] is a popular permissionless blockchain that supports decentralized applications through smart contracts. Within the Ethereum blockchain, It is feasible for non-trusting entities to communicate in a secure way without involving a trusted third party [6]. This is the main feature blockchains can offer compared to a traditional centralized system. Ethereum has its currency, referred to as Ether.

Similar to the majority of permissionless blockchains, Ethereum offers a distributed ledger that is shared among all the network nodes [7]. This ledger is structured in blocks, where every block can contain a number of transactions. The Ethereum network constitutes several mining nodes whose responsibility relies on extending the blockchain ledger by creating and attaching new blocks of transactions.

2.1 Key Concepts

Unlike other permissionless blockchains such as Bitcoin, Ethereum can support smart contracts on top of the blockchain layer. Ethereum smart contracts can offer trust between non-trusted entities, and can be used to build various decentralized applications such as voting systems [8, 9], real-estate property systems [10], crowdfunding services [11], healthcare applications [12, 13] as well as providing access control and authentication services to existing system [14–17].

In this section, we explain and highlight the key concepts of the Ethereum blockchain relevant to our study. This embraces Ethereum accounts, blockchain, transactions, consensus algorithm, and incentive mechanism.

Ethereum Accounts In Ethereum, there are two types of accounts, namely, *externally owned accounts* and *contract accounts*. Every Ethereum account has a balance and an address to be called at. Contract accounts can also have associated code and storage. Different accounts can communicate with one another by creating and submitting a transaction. For example, an externally owned account can send a transaction to invoke an existing smart contract.

Ethereum Blockchain The Ethereum blockchain consists of several blocks that are linked with one another. Every new block created is linked with the previous one, forming a chain of blocks. Every block has a header and a body. The block header contains various information relevant to the block, for instance, block Gas limit, block Used Gas, and other data. The block Gas Limit indicates the maximum amount of gas units to be consumed by all transactions embraced in the block, and it is mainly designed to prevent denial of service attacks by limiting the possible computation. This is especially true as Ethereum can support the creation and execution of complex smart contracts. The block Used Gas reflects the actual amount of gas units consumed by all transactions embraced in the block. The block is considered full when the block Used Gas is equal to the block Gas Limit.

Ethereum Consensus Algorithm Ethereum currently uses Proof-of-Work (PoW) Algorithm to maintain and update the blockchain state. The consensus algorithm is augmented with the longest-chain role to decide on which blockchain branch to

follow when conflicts arise. It is worth noting that Ethereum and other blockchains are considering the move to the Proof of Stake (PoS) algorithm since PoW is not an energy-efficient algorithm. Unlike Bitcoin, the Ethereum consensus algorithm allows referencing stale blocks (blocks that are not considered as part of the main chain after resolving the conflicts) in future blocks and rewarding the nodes that created them. Such blocks are referred to as uncle blocks in Ethereum, and they do not constitute the state of the blockchain ledger.

Ethereum Transactions In Ethereum, transactions play a significant role in updating the state of the Ethereum blockchain. The balance of an Ethereum account or the value of a smart contract parameter can only be changed through transactions. Transactions are executed and confirmed by embracing them in a block which can then be attached to the Ethereum global ledger.

Ethereum transactions can be categorized into two types, namely, contract and transfer transactions. A contract transaction is to create and deploy a new smart contract to the Ethereum ledger. After creating the contract, it is possible to send a transaction to the contract address in order to execute specific functions of that contract. A transfer transaction is simply to move Ether, the Ethereum currency, from one account to another.

Different attributes form the Ethereum transaction. In this section, we only highlight the attributes relevant to our study such as gas attributes (Gas Price, Gas Limit, and Used Gas). *Gas Price* indicates the amount of Ether to be paid for each unit of gas the transaction consumes. *Gas Limit* indicates the number of gas units the transaction can consume at most, and it is mainly used to assure transaction termination. Both Gas Price and Gas Limit can be set by the submitter of the transaction. *Used Gas* indicates the total amount of gas units consumed by the transaction during its execution. Transfer transactions consume 21,000 units of gas. The amount of Used Gas for contract transactions varies depending on transaction complexity. Transactions that require more computational effort result in more consumption of gas units. The submitter of the transaction is required to pay a transaction fee to execute their transactions. The transaction fee is the multiplication of Used Gas and Gas Price, and it is to compensate miners for their effort.

Ethereum has a built-in virtual machine that is responsible for executing smart contract transactions [5]. The Ethereum Virtual Machine (EVM) resides at every node in the network, and it can support various instructions such as `ADD` and `PUSH`. Every instruction that is executed in the EVM has a particular gas cost.

Ethereum Incentive Mechanism The key idea of the incentive mechanism is to provide rewards for the network nodes as compensation for their effort spent toward maintaining the blockchain ledger. In Ethereum, when miners create new valid blocks and successfully attach them to the ledger, miners would receive a fixed block reward (currently, it is 2 Ether per block) and the fees associated with all transactions embraced in the block. In addition, miners are given a small amount of reward for every uncle block that is either created or referenced in a valid block [5].

To assure the security of the Ethereum network, Ethereum incorporates the *Gas model* in the design of its incentive mechanism. This is to control the possible computation for transactions and to provide incentives for miners for their effort. Ethereum has a rich programming language that enables the writing of complex

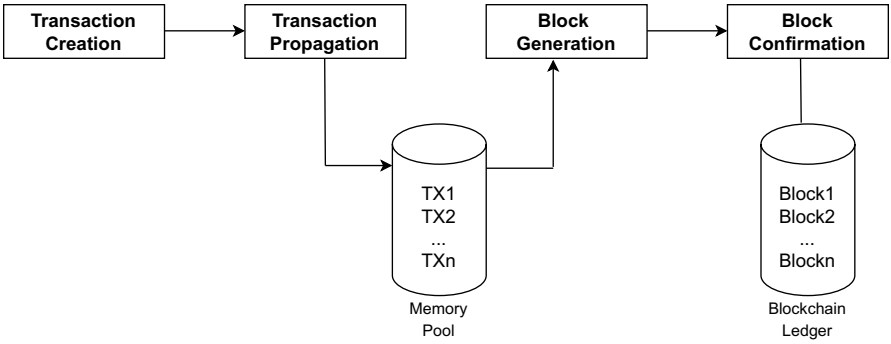

Fig. 1 The life cycle of Ethereum transactions

smart contracts which may, for instance, contain infinite loops. That is, the Gas model is important to ensure transaction termination, thus avoiding denial of service attacks. Every smart contract instruction has a predefined gas cost that is set by the Ethereum foundation based on the storage and computation overhead needed to run that instruction on the EVM [5]. For instance, the gas cost of `ADD` instruction is three units of gas. During transaction execution, the EVM calculates the amount of Used Gas for all instructions used by the transaction and then charges the submitter of the transaction accordingly.

2.2 Transaction Life Cycle

Figure 1 illustrates the life cycle of transactions within the Ethereum blockchain system.

Transaction Creation Any node in the blockchain network can create a transaction either to transfer digital assets or to invoke an existing smart contract. The number of transactions that can be created by all the network peers can vary from time to time. For instance, the average daily transactions in Ethereum are currently just over one million transactions. This number is expected to increase when the number of network nodes increases or when the number of deployed smart contracts increases.

Transaction Propagation Once a transaction is created by a network node, the transaction is expected to be sent to other nodes in the system to get it confirmed and added to the blockchain ledger. Every node in the network maintains a memory pool that acts as a placeholder for the newly created transactions. Such transactions remain in the pool as unconfirmed transactions, and they are only removed from the pool once they have been appended to the blockchain ledger. The size of the memory pool is constantly changing according to the number of transactions entering and leaving the pool.

Block Generation In blockchain systems, transactions are only confirmed by adding them to a particular block. A group of nodes, often called miners, is constantly appending new blocks of transactions. To generate a new block, miners first select a set of unconfirmed transactions from the memory pool to execute and embrace in the block. The block is then constructed by participating in a specific

consensus mechanism such as Proof-of-Work (PoW) or Proof-of-Stake (PoS). The constructed block is then forwarded to other peers in the network. Upon the reception of the newly created block, nodes have to verify the block and its embedded transactions.

Block Confirmation Once the block is verified and accepted by most of the network nodes, it will be added to the blockchain ledger. At this stage, the block and its embedded transactions are confirmed, and miners can then start building new blocks on top of it.

3 Latency Factors

In Sect. 2.2, we outline the life cycle of blockchain transactions from the creation phase to the phase where the transaction is confirmed and appended to the blockchain ledger. To model and understand transaction latency within the blockchain context, we first need to identify all factors and parameters that can affect transaction latency. Having included all the possible factors can help us understand the latency bottleneck problem properly.

In this section, we identify the various factors that could contribute to transaction latency in permissionless blockchains. We also outline and discuss the reason why these factors (parameters) can contribute to transaction latency.

- **Transaction Fee** The fee-related parameters (e.g., Gas Price in the Ethereum network) are the most obvious and reported contributing factor to transaction latency [18]. Miners often prioritize transactions by selecting transactions that offer the highest fee to embrace in their forthcoming blocks. That is, the higher the Gas Price offered by a transaction, the more likely that the transaction is going to be added to the next block, resulting in a lower waiting time. We note that the Gas Price value for a transaction can be set by the transaction submitter.
- **Block Gas Limit** This parameter reflects the number of transactions that can be included in a single block, and it is called block size in other blockchain systems such as Bitcoin. The larger the block limit the more transactions can be processed at once and vice versa.
- **Block Interval Time** This parameter indicates the arrival rate for blocks in the blockchain network. Reducing the block interval time means that more blocks will be created, thus reducing the latency for the network transactions. For instance, the block interval time of Bitcoin is 600 s, which means about 144 blocks are created per day. In Ethereum, the block interval time is set at about 13 s, resulting in 6646 blocks per day.
- **Transaction Arrival Rate** This parameter indicates the number of transactions to be created per second in the blockchain network. The arrival rate of transactions is expected to increase with the growth of blockchain adoption. According to Etherscan,[1] the number of transactions arrived per second in the network

[1] https://etherscan.io/.

ranges from 1000 and 2500 transactions. This number is by far larger than what the Ethereum network can handle at the moment (about 14 transactions per second). Later in Sect. 6 we will show how increasing the arrival rate of transactions can impact the waiting time for transactions.

- **Behavior of Network Nodes** This is also an important factor in transaction latency, that has not been well studied in the literature. The blockchain network relies on a number of mining nodes to continuously create and append new blocks of transactions to the blockchain ledger. Such nodes have the right to select which transactions to include in their forthcoming blocks. That means there is no guarantee that transactions would be processed in a queue-based fashion. Furthermore, nodes can decide on the number of transactions to include in their blocks. That is, nodes can generate blocks that only contain a small number of transactions or even without any transactions [19]. When nodes do not utilize their block properly (e.g., not filling their blocks with transactions), the waiting time for transactions is expected to go up.

4 Related Work

In this section, we discuss the current state-of-the-art literature related to the performance evaluation of permissionless blockchains concerning transaction latency. We also discuss how our work differs and contributes to the existing literature. A comparison summary of related work is presented in Table 1.

Regarding transaction latency analysis of the Bitcoin network, there are various research efforts to evaluate the factors contributing to transaction latency. In [20], the authors analyze the confirmation time for over 50 million Bitcoin transactions according to the fee offered by those transactions. Their results suggest that transactions that do not offer fees can have a longer waiting time. Their results also demonstrate the lack of correlation between transaction confirmation time and the amount of fee offered by transactions. Easley et al. [19] evaluate the impact of transaction fees on the behavior of the Bitcoin network nodes by developing a game-theory model. They found that transaction fees can motivate the network nodes to process transactions, thus reducing the waiting time for transactions. In [21], the authors evaluate and analyze how the block size can impact the confirmation time of Bitcoin transactions using queueing models. In [22], the authors evaluate the effect of block interval time on transaction latency using architectural and simulation models. Kawase et al. [23] conduct simulation experiments to evaluate the impact of the transaction arrival rate and block size on the latency of Bitcoin transactions. Wilhelmi et al. [24] analyze the impact of block size on transaction latency using simulation techniques.

Concerning the Ethereum network, there are a limited number of studies that evaluate transaction latency. Spain et al. [18] conduct an analysis of transaction latency on the Ethereum blockchain. They observe a weak correlation between transaction fees and the latency results and found that this may be attributed to network congestion. However, the data set used in this analysis study is limited as it only covers a period of 24 days in 2017. In [25, 26], the authors analyze the correlation between

Table 1 A comparison summary of related work

References	Blockchain system	Evaluation method	Latency factor				
			Transaction fees	Block limit	Block interval time	Transaction arrival rate	Behavior of nodes
[19]	Bitcoin	Game theory models	✓	✗	✗	✗	✗
[20]	Bitcoin	Analysis of past data	✓	✗	✗	✗	✗
[21]	Bitcoin	Queueing models	✗	✓	✗	✗	✗
[22]	Bitcoin	Architectural and simulation models	✗	✗	✓	✗	✗
[23]	Bitcoin	Simulation	✗	✓	✗	✓	✗
[24]	Bitcoin	Simulation	✗	✓	✗	✗	✗
[18]	Ethereum	Analysis of past data	✓	✗	✗	✓	✗
[25]	Ethereum	Correlation analysis	✓	✗	✗	✗	✗
[26]	Ethereum	Analysis of past data	✓	✗	✗	✗	✗
[27]	Ethereum	Simulation	✓	✗	✗	✗	✗
Our work	Ethereum	Queueing models and simulation	✓	✓	✓	✓	✓

transaction fees and the confirmation time for Ethereum transactions. Their empirical results suggest the lack of such correlations. Similarly, Guo et al. [27] analyze the effect of various transaction fees on transaction latency within the Ethereum network using simulation models.

Besides these studies, there are various tools (e.g., EthGasStation[2]) that are designed to predict the waiting time for Ethereum transactions based on the given Gas Price value. Such tools are meant to help users set the appropriate Gas Price value according to the waiting time they can tolerate. Although they seem to be beneficial assistant tools, several studies (for instance, [28, 29]) have evaluated their accuracy and found that the margin error of these tools can exceed 25%. The authors of [28, 29] stated the reason for such inaccuracy is that several factors may impact the value of Gas Price, including the number of network miners and the number of pending transactions.

Based on the existing studies, we can see most of the research efforts are dedicated to the Bitcoin network, compared to the Ethereum network. We also notice that most of the current literature focuses on the impact of transaction fees on the time it takes to confirm transactions. That is, there is a lack of research studies that consider various factors that may contribute to transaction latency in blockchain systems. To the best of our knowledge, we provide the first analysis study that accounts for the behavior of the network nodes and its impacts on transaction latency. Besides, we propose different ways to improve transaction latency and consider the security issues that may arise when applying those proposals.

5 Proposed Latency Model

In this section, we propose a model that simulates and measures the latency for Ethereum transactions taking into consideration different factors that may potentially contribute to the latency results. We consider the Ethereum blockchain as a case study of our model in this article, although it can be applied to other permissionless blockchains such as Bitcoin.

Before we dive into the proposed model, we first state and discuss the model assumptions that will hold throughout the study. We consider a transaction selection strategy based on the value of Gas Price for all participating nodes. When a particular node is about to create a new block, it will sort pending transactions and then select the ones that offer the highest Gas Price. The reason why we assume this selection strategy is that nodes are often rational in that they aim to maximize their revenue. That is, we do not consider other transaction selection strategies that can be adopted by the network nodes. Knowing which strategies are currently used by the network nodes is not possible as nodes can be malicious, faulty, or rational, and their strategies may differ from time to time. Secondly, as a first approximation, we do not consider transaction propagation delay (the time duration from creating a transaction to having it placed in the memory pool) as we assume that other nodes would

[2] https://ethgasstation.info/

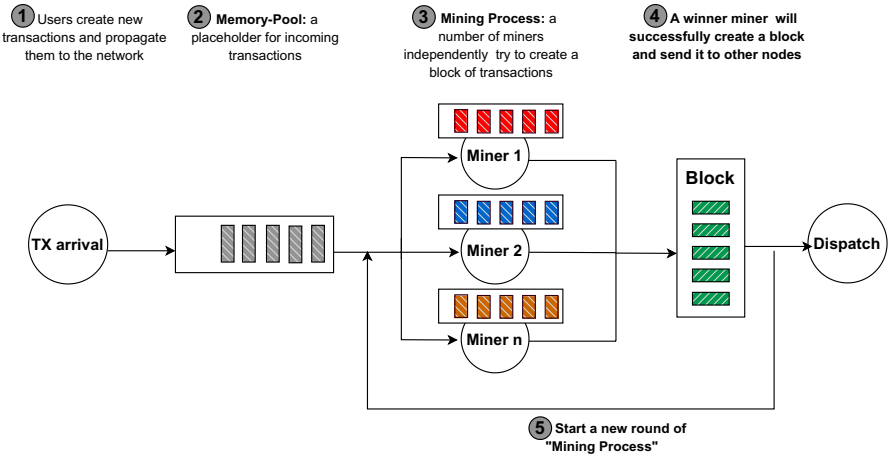

Fig. 2 Latency box plot results for different transaction arrival rates

immediately receive the newly created transactions. The time delay for transaction propagation depends on the underlying network broadcast protocol and the size of transactions. However, recent studies such as [30] have shown that the propagation delay for Bitcoin and Ethereum transactions is negligible, at 0.42 and 2.3 s, respectively. With the improvement of network broadcasting protocols, the propagation delay for transactions is expected to decrease further. However, this delay can be added as a separate parameter to the model if needed.

In this article, we adopt a queueing model with a priority mechanism to study transaction latency, as depicted in Fig. 2. We consider a memory pool as a single queue with multiple servers or users. The purpose of the memory pool is to hold unconfirmed transactions. Users of the blockchain system can create new transactions and then propagate them to the network to be confirmed and appended to the blockchain ledger. We model transaction arrival in the system as a Poisson process with a configurable rate λ. Newly arrived transactions are considered as unconfirmed transactions, and they have to wait in the queue until they have been selected and added to a forthcoming block.

We consider a set of miners in the system to act as servers to confirm incoming transactions by adding them to the blockchain ledger. Such miners try to create a block of transactions by selecting a set of the transactions waiting in the queue. Miners adopt a priority mechanism for which transactions to select and include in their forthcoming blocks. To model this priority mechanism, we consider the decision miners take to sort transactions in the queue and select a subset of those transactions to include in their blocks. For each miner, we could initiate a different transaction selection criteria, but as discussed earlier in this section, we decide to adopt the same strategy for all miners in this study. Since each block has a limit on the number of transactions it can contain, miners would include transactions with the highest fee (in terms of Gas Price value) first until the block is full or no transactions waiting in the queue.

It is worth noting that miners in the system work independently from each other. They compete against each other for providing the service of transaction confirmations. In particular, they have to perform specific mathematical tasks (e.g., finding a hash value that contains the right number of zeros in the prefix). The miner who successfully finds the right hash value for the block is eligible to append his block to the global blockchain ledger. At that stage, the included transactions would be confirmed.

The service time for our queueing model is defined as the time elapsed between two consecutive blocks, which we referred to as block interval time *Bt*. That means the service is scheduled to occur at certain times (e.g., every 13 s in the case of the Ethereum network).

6 Simulation Setup and Results

We implemented our latency model as a discrete-event simulator in Python and made it publicly available[3].

Studying transaction latency bottlenecks is neither possible nor practical on real blockchain systems. Simulation models provide a convenient and reasonable way to understand transaction latency bottlenecks without the need for interrupting the running system. With simulations, it is possible to derive useful insights about transaction latency through "What-if questions" [31]. For example, how latency can be impacted when the arrival rate for transactions is increased in the network or how the behavior of the participating nodes can result in higher latency figures.

6.1 Configuration of Simulation Parameters

To conduct simulation experiments that investigate transaction latency in Ethereum, various parameters have to be gathered and fed into the simulator. Part of these parameters can be directly collected from the Ethereum network, for instance, through Etherscan explorer. Other parameters can be set by the user of the simulator to see how different values for those parameters can impact transaction latency.

Table 2 shows the main parameters that need to be configured before running the simulator. We organized these parameters into four groups, namely, block parameters, transaction parameters, node parameters, and simulation parameters.

Block Parameters Three different parameters related to blockchain blocks need to be configured. In order to obtain realistic simulation results, we collect the data for these parameters from Etherscan explorer. The first parameter is the block interval time (B_t), which indicates how often blocks are created in the network (e.g., the elapsed time in seconds between two consecutive blocks). Based on Etherscan explorer, the current value of B_t in Ethereum is 13.18 s. The second parameter is

[3] https://github.com/maher243/Transaction-Latency.

Table 2 Simulation parameters

Type	Parameter	Description
Blocks	Block interval time (B_t)	Average elapsed time in seconds between two consecutive blocks.
	Block limit (B_l)	Block capacity that reflects how many transactions can fit into a single block
	Block utilization (B_u)	Fullness of the block to be created
Transactions	tx_{α}	Arrival rate of transactions in the system
	tx_{ug}	The used gas value of a transaction
	tx_{gl}	The gas limit value of a transaction
	tx_{gp}	The gas price value of a transaction
Nodes	N_n	Total number of nodes in the network
	H_i	The hash power of the ith node
Simulation	Sim_t	Length of the simulation time in seconds
	Sim_r	Number of simulation runs

block limit (B_l), which indicates the capacity or size of the block. The number of transactions that can fit in a block is directly influenced by this parameter. The current value of B_l in Ethereum is 30 million units of gas. The last parameter is block utilization (B_u), which indicates how full the block is to be created. The value of B_u is dependent on the node creating the block, and it can range from 0% (empty block) to 100% (full block).

Transaction Parameters We collect the data for transaction information such as transaction arrival rate and gas-related parameters. Transaction arrival rate (tx_{α}) indicates the number of transactions per second that can be created in the network. The value of tx_{α} is continuously changing over time depending on how congested the network is. We collect the data for this parameter from Etherscan explorer. To obtain gas-related parameters, we implement a python script that utilizes Etherscan APIs to retrieve transaction information. We retrieve the data for the latest 100,000 transactions and then fit the appropriate distributions. This would allow us to generate a sample from the fitted distributions every time a new transaction is about to be created in the simulator.

Node Parameters Two main parameters related to the participating nodes need to be configured. The first parameter is the number of nodes in the network (N_n). The second one is the hash power of each participating node. The hash power parameter is essential in the PoW consensus algorithm as it indicates how often a particular node creates a block in the network. The larger the hash power value for a node the more likely that node will generate the next block. We note that this parameter only applies to mining nodes that participate in the process of creating blocks. For ordinary nodes that are not participating in the mining process, we can set their hash powers to zero.

Simulation Parameters These parameters are mainly to set the length of the simulation time (Sim_t) and the number of simulation runs (Sim_r). Increasing the

length of simulation time can result in more blocks and confirmed transactions and vice versa.

6.2 Simulation Results

In this section, we present the main findings regarding transaction latency in the Ethereum blockchain. In particular, we show how different factors can contribute to transaction latency. These factors include transaction arrival rate, block limit, block interval time, block utilization level as well as transaction fee. Our main metric of interest is transaction latency, which indicates the amount of elapsed time in seconds needed for a transaction from its creation to the time when the transaction is embraced in a block.

We summarize the main findings that follow from our discussion upfront:

- Transaction arrival rate (the number of transactions created per second in the network) can significantly impact transaction latency. When the network is congested with transactions, the overall waiting time for individual transactions is expected to increase.
- Transaction latency can be reduced by adjusting the blockchain system properties such as block limit and block interval time. Increasing the block limit can reduce the overall latency for transactions as more transactions can be processed per single block. Reducing the block interval time can also result in a lower waiting time for transactions as more blocks can be created in the network.
- Another factor that impacts transaction latency is block utilization. When blocks are filled up with transactions, the overall waiting time for transactions is expected to decrease. That is, the behavior of the block creators can play a significant role in transaction latency, and thus nodes should be encouraged to utilize their blocks properly to assure less waiting time for transactions.
- As it is commonly known, transaction fees are one of the most important factors that impact transaction latency. When a node is about to create a block, it often selects pending transactions that offer the highest fee from its pool to maximize its revenue. However, the waiting time for transactions with a specific transaction fee can vary depending on other contributing factors such as transaction arrival rate and block interval time.

6.2.1 Impact of Transaction Arrival Rate

Figure 3 shows the latency results in seconds for transactions, taking into account the arrival rate for transactions in the network. This box plot figure can help us see the minimum, maximum, median, first quartile, and third quartile results for transaction latency. We set the block limit and the block interval time as currently configured in Ethereum (30 million units of gas and 13.48 s, respectively). We assume blocks are fully utilized with transactions to simulate the best-case scenario.

From Fig. 3 we can observe the following insights. First and most importantly, we observe a strong positive correlation between transaction arrival rate and latency.

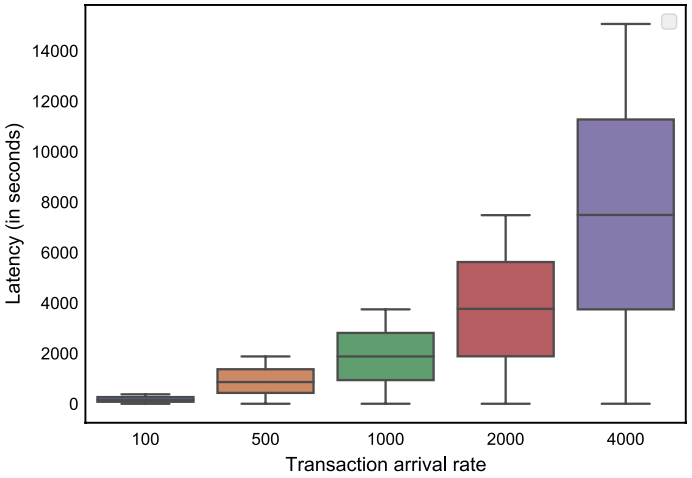

Fig. 3 Latency box plot results for different transaction arrival rates

Increasing the arrival rate for transactions can result in higher latency and vice versa. For instance, the average transaction latency can be increased from 158 to 7500 s when the arrival rate for transactions is increased from 100 to 4000 transactions per second. This is somehow expected since in blockchains each block has a limit on how many transactions it can include as well as blocks arrive at certain times. When we push more transactions into the network than its capacity this would result in more delays in processing those transactions. Secondly, we observe different waiting times for transactions that fall under a specific configuration of arrival rate. This is because transactions in our simulation have different Gas Price values, thus the waiting time for transactions is significantly different from one another. For example, the latency for transactions can range from 0 to 15,000 s in the case of a 4000 transaction arrival rate. Furthermore, we can observe that the maximum waiting time for transactions can also increase when the transaction arrival rate increases. For example, the maximum waiting time for transactions can be increased from 400 to over 15,000 s when the arrival rate is increased from 100 to 4000 transactions per second. The minimum waiting time for transactions is almost the same for all the different configurations of transaction arrival rates. This is because transactions with high fees are given priority by the network nodes, thus they are processed as soon as they appear in the network, regardless of the arrival rate of transactions.

6.2.2 Impact of Block Limit

Figure 4 shows the latency results in seconds for transactions, for different configurations of block gas limits. We set the arrival rate for transactions at 2000. The block interval time is set as currently configured in the Ethereum network, at 13.48 s. We assume blocks are fully utilized with transactions.

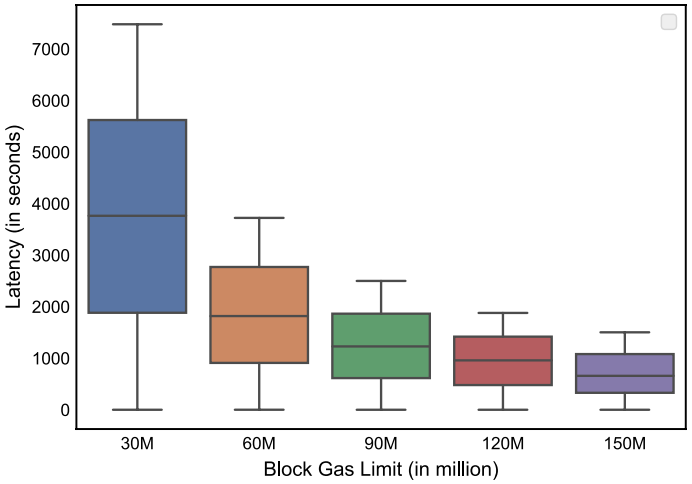

Fig. 4 Latency box plot results for different block limits

From Fig. 4 we can observe a strong negative correlation between block limit and transaction latency. Increasing the block limit can result in a reduction in transaction latency and vice versa. For instance, the average waiting time for transactions can be reduced by 82% when the block limit is increased from 30 to 150 million units of gas. This is because increasing the block limit means more transactions can fit into a single block, thus resulting in a reduction of transaction latency. Secondly, we can observe that the maximum waiting time for transactions can also decrease when the block limit increases. For example, the maximum waiting time for transactions is over 7000 s when the block limit is 30 million units of gas, while it is about 1500 s for the case when the block limit is 150 million units of gas. Similar to the results of the transaction arrival rate presented in Sect. 6.2.1, we observe different waiting times for transactions that fall under a specific configuration of block limit as well as we observe almost the same minimum waiting time for transactions across the different configurations of block limits.

6.2.3 Impact of Block Interval Time

Figure 5 shows the latency results in seconds for transactions, for different configurations of block interval times. We set the arrival rate for transactions at 2000. The block limit is set as currently configured in the Ethereum network, at 30 million units of gas. We assume blocks are fully utilized with transactions.

From Fig. 5 we can observe a strong positive correlation between block limit and transaction latency. Increasing the block interval time can increase transaction latency and vice versa. For instance, the average waiting time for transactions can be increased by 196% when the block interval time is increased from 5

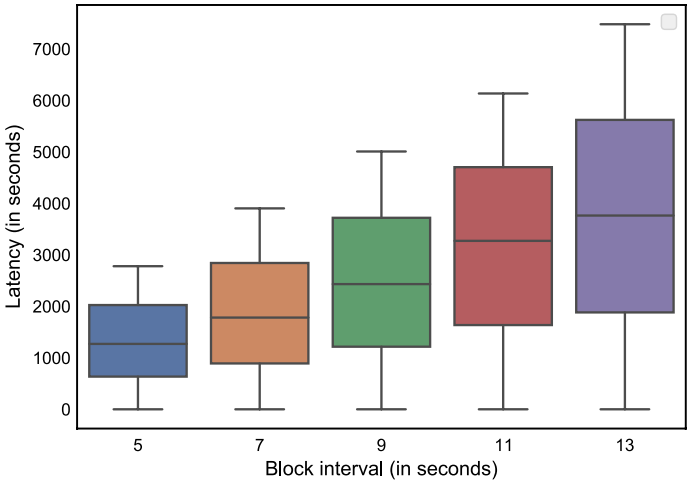

Fig. 5 Latency box plot results for different block interval times

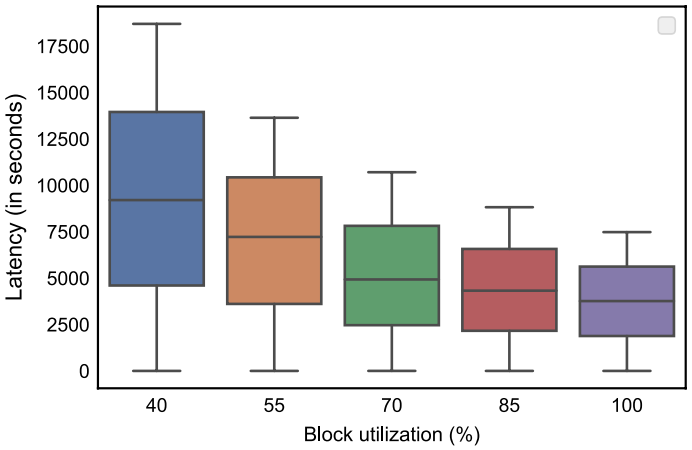

Fig. 6 Latency box plot results for different block utilization levels

to 13 s. That means decreasing the block interval time for the Ethereum network from its current value of 13 to 5 s would help reduce the average transaction latency by about two-thirds. This is because reducing the block interval time means more blocks can be created in the network, thus the processing time for transactions will be minimized. Secondly, we can observe that the maximum waiting time for transactions can also increase when the block interval time increases. For example, the maximum waiting time for transactions is about 7500 s when the block interval time is 13 s, while it is under 2800 s for the case when the block interval time is 5 s. Similar to the results presented in previous sections, we observe different waiting times for transactions that fall under a specific configuration of block interval time as well as we observe almost the

same minimum waiting time for transactions across the different configurations of block interval times.

6.2.4 Impact of Block Utilization Level

Figure 6 shows the latency results in seconds for transactions, for different configurations of block utilization levels. We set the arrival rate for transactions at 2000. The block limit and the block interval time are set as currently configured in the Ethereum network, at 30 million units of gas and 13.48 s, respectively.

From Fig. 6 we can observe a strong negative correlation between block utilization level and transaction latency. Increasing the block utilization level can result in a reduction in transaction latency and vice versa. For instance, the average waiting time for transactions can be reduced by 59% when the block utilization level is increased from 40% to 100%. This is because more transactions can be processed in a single block, and thus the latency for transactions is expected to decrease. Secondly, we can observe that the maximum waiting time for transactions can decrease when the block utilization level increases. For example, the maximum waiting time for transactions is about 18,700 s when the block utilization level is at 40%, while it is about 7500 s for the case of 100% block utilization level. Similar to the results of previous sections, we observe different waiting times for transactions that fall under a specific configuration of block utilization level as well as we observe almost the same minimum waiting time for transactions across the different configurations of block utilization levels.

To sum up, maximizing the block utilization level can help reduce the waiting time for transactions. Block utilization depends on the behavior of the block creators who are not enforced to utilize their blocks properly in the current implementation of permissionless blockchains. That is, miners should be encouraged or enforced to fill their blocks with transactions to reduce the latency for transactions.

6.2.5 Impact of Transaction Fee

In order to investigate the impact of transaction fees on the waiting time for transactions, we run different simulations to observe transaction latency based on different Gas Price values. We also consider different network parameters that may impact latency along with Gas Price.

Figure 7 shows the average latency for transactions based on the Gas Price value, for different transaction arrival rates and different block utilization levels. The five curves in each of the two plots of Fig. 7 indicate different Gas Price values for transactions. We set the block limit and the block interval time as currently configured in the Ethereum blockchain. In Fig. 7a, we consider a utilization level of 100%, which means all blocks are filled with transactions. In Fig. 7b, we consider an arrival rate of 2000 transactions per second.

Figure 8 shows the average latency for transactions based on the Gas Price value, for different transaction block limits and different block interval times. The five curves in each of the two plots of Fig. 8 indicate different Gas Price values for transactions. We set the arrival rate for transactions to 2000 transactions per second and

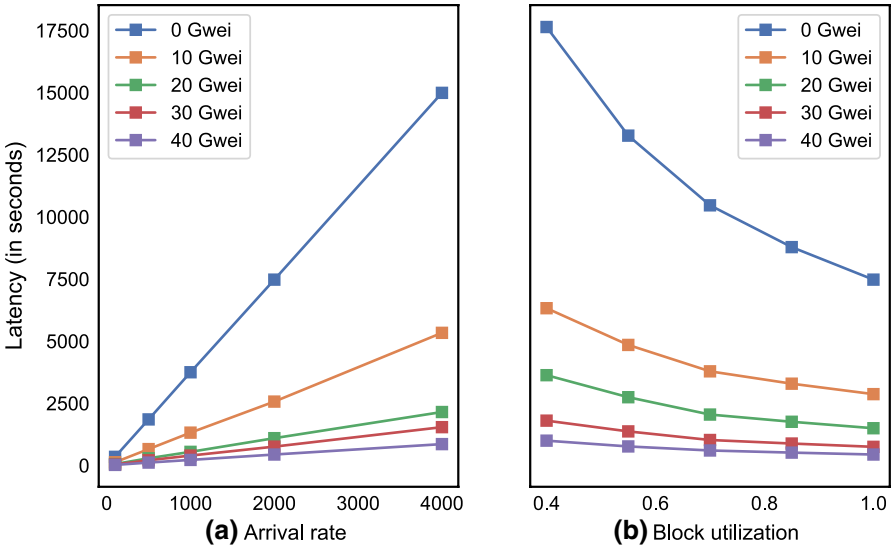

Fig. 7 Latency results for different Gas Price values (in Gwei) for **a** different transaction arrival rates and **b** different block utilization levels

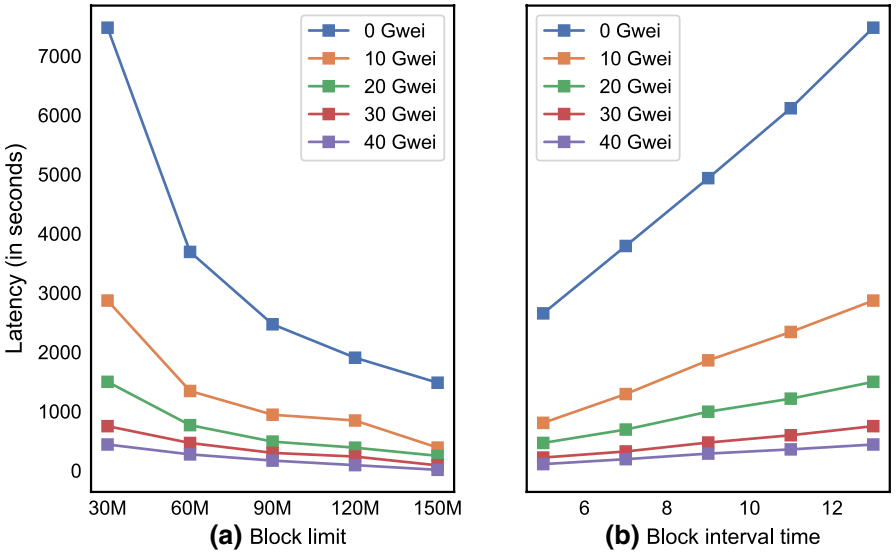

Fig. 8 Latency results for different Gas Price values (in Gwei) for **a** different transaction block limits and **b** different block interval times

the block utilization level to 100%. In Fig. 8a, we consider a block interval time of 13.48 s, as currently set in the Ethereum network. In Fig. 8b, we set the block limit to 30 million units of gas, as currently implemented in the Ethereum network.

From Figs. 7 and 8, we conclude that there is a strong negative correlation between Gas Price value and latency. The more Gas Price offered by a transaction, the less waiting time for transactions is expected and vice versa. For example, the latency for transactions with a Gas Price value of zero is about 15,000 s, while that for transactions with a Gas Price value of 40 is 860 s. This makes sense as miners are often rational (e.g., they select and include transactions that offer the highest Gas price value in their forthcoming blocks).

Interestingly, our simulation results show that although Gas Price is an important parameter that can impact the latency for transactions, other parameters such as transaction arrival rate, block limit, block interval time, and block utilization level can also contribute to latency. For example, the average waiting time for transactions with a specific Gas Price value can significantly vary depending on the arrival rate for transactions in the network. The latency for transactions with a Gas price value of zero is 100 s in the case of an arrival rate of 100 transactions, while it is about 15,000 s when the arrival rate is increased to 4000 transactions.

It is worth noting that the latency for transactions with a particular Gas Price value increases with the increase of either block interval time or transaction arrival rate. On the contrary, the latency decreases with the increase of block limit or with the increase of block utilization level. Increasing block limit and block utilization level means more transactions are embraced in a single block, and thus the waiting time for transactions is expected to decrease.

To that end, one would expect different waiting times for their transactions. The Gas Price value or transaction fee alone is not enough to determine the waiting time for transactions. When the network is congested with transactions, the submitter of the transactions needs to offer a higher Gas price value to reduce the latency. The same applies when miners do not utilize their blocks properly, when the block limit is relatively small or when the block interval time is relatively large.

7 Discussion

Our analysis of transaction latency presented in this work is not limited to the Ethereum network, but it can also be applied to other permissionless blockchains such as Bitcoin. This is because most of the parameters (such as block interval time, block utilization, number of nodes, and transaction arrival rate) are common among blockchain systems. We note the difference can be of relation to transaction information as Ethereum uses the gas model within its incentive mechanism.

In this section, we discuss the validity threats to our evaluation of transaction latency within the Ethereum network. Also, we discuss different approaches to improve transaction latency and highlight the security considerations for each of those approaches.

7.1 Validity Threats to Our Study

In this section, we discuss the validity threats to our latency model and our analysis derived from this work.

Transaction Selection Criteria In Sect. 6.2 we show the latency results for transactions assuming that all nodes in the network follow the same transaction selection strategy. In particular, we assume nodes are rational in that they would select transactions with the highest fee first to maximize their rewards. It is infeasible to know the selection strategy adopted by the network nodes in existing blockchain systems. Nodes might have different motivations for their participation such as rational or malicious intents. However, it is possible to extend our model to include different selection strategies for transactions. Also, it is possible to add different strategies for different nodes when needed.

Transaction Propagation We present the results for transaction latency without considering the delay in propagating transactions, as discussed in Sect. 5. The waiting time for transactions can be slightly increased when introducing the transaction propagation delay. In that sense, our analysis should be considered as a best-case analysis. However, the results we derived from our analysis are still sound, especially as the propagation delay is negligible. If needed, the model can be extended to count for transaction propagation delays.

7.2 Transaction Latency Improvement and Relevant Security Considerations

Based on our analysis and the results that we present in this work, we can suggest the following approaches to minimize transaction latency. For each approach, we also highlight and discuss the relevant security considerations.

Increasing Block Limit A straightforward approach to improve the performance of permissionless blockchains is through increasing the block limit as it reduces the waiting time for pending transactions. However, increasing the block limit can degrade the security level of the blockchain system. This is because large blocks require more time to be propagated and validated, thus increasing the chance of network conflicts (a condition that occurs when multiple blockchain versions exist). To that end, it is crucial to not push the block limit to a level where the security of the blockchain system is negatively impacted. Rigorous analysis and experimentation are needed to determine to which level the block limit can be increased without scarifying the security of the system.

Reduction of Block Interval Time Transaction latency can also be minimized by reducing the block interval time, as more blocks can be generated per time slot. However, the number of network conflicts is expected to increase as a result. From a security standpoint, the time between two consecutive blocks should be larger than the time needed to propagate and validate a single block. Rigorous analysis and experimentation are needed to determine the lowest block interval time that can maintain the system security. To that end, the block interval time should not be simply reduced unless we assure that the security of the blockchain system is maintained.

Table 3 Transaction latency (in seconds) for three different improvement proposals as opposed to that of the current Ethereum system

Approach	Block limit (Gas)	Block interval (s)	Latency (s)
Current Ethereum	30 million	13.18	3765
Improvement proposal 1	60 million	11	1613
Improvement proposal 2	90 million	9	810
Improvement proposal 3	120 million	7	378

Encouraging Nodes to Fill Their Blocks with Transactions Another approach to improve transaction latency is through a proper design of the integrated incentive mechanism. The current incentive mechanism of permissionless blockchains (such as Bitcoin and Ethereum) does not push nodes to properly utilize their blocks. Nodes can still gain significant rewards for a block, even when the block does not contain transactions. Regarding security, there is no direct impact on the security of blockchain systems when implementing this solution. That is, we propose to adjust the incentive mechanism of permissionless blockchains to ensure that the rewards nodes would receive are proportional to block utilization (the more transactions included in the block the more rewards are given to the nodes and vice versa). This would encourage nodes to utilize their blocks properly. The only security concern when nodes utilize their blocks is that other nodes have to verify the recipient blocks and their transactions, which may encourage various nodes to skip the verification process [32, 33]. When this happens, the blockchain system would contain unverified blocks and transactions, and as a result, the security of the system will be questionable.

Evaluation Results In Sect. 6, we show how transaction latency can be impacted by each of the contributing factors (e.g., block limit and block interval time) individually. However, when these factors are considered together the latency for transactions can significantly be reduced. Earlier in this section, we proposed improving transaction latency by (1) increasing the block limit, (2) reducing the block interval time, and (3) encouraging miners to utilize their blocks properly. To show how transaction latency can be minimized when combining all these three approaches, we conduct various simulation experiments and compare the latency figures of our proposed schemes with that of the current Ethereum system.

Table 3 presents the average latency figures for transactions (in seconds) for both the current Ethereum network and for our improvement proposals. In all experiments, we assume nodes always fill their blocks with transactions and also assume the arrival rate for transactions is fixed at 2000 transactions per second. We found that increasing the block limit to 60 million units of gas and reducing the block interval rate to 11 (the first improvement proposal) can result in a reduction of transaction latency by just over 50% as opposed to that of the existing Ethereum network. The second and third improvement proposals are to show to what extent transaction latency can be minimized when we keep changing the block limit and the block interval time parameters. In the third proposal, for instance, the average transaction latency is 378 s compared to that of the current Ethereum at 3765 s.

8 Conclusion

This article proposes a queueing model to estimate transaction latency within permissionless blockchain systems, considering various factors that affect the latency results. We identify all the factors and parameters that can affect transaction latency and then conduct extensive simulation experiments with different configurations of simulation parameters. The factors we considered in our analysis are transaction fees, block limit, block interval time, transaction arrival rate as well as the behavior of the network nodes concerning block utilization. Our simulation results suggest that all the contributing factors we identified can have an impact on transaction latency. We find the behavior of the mining nodes can significantly impact transaction latency, especially since nodes are not enforced to utilize their blocks properly in the current implementation of permissionless blockchains.

From our analysis, we can observe the following insights. The waiting time for transactions is expected to increase when the arrival rate of transactions increases or when the block interval time increases. On the contrary, the average waiting time for transactions is expected to decrease with the increase of the block limit or when nodes utilize their blocks properly. Concerning transaction fees, we found that transactions with a high fee are expected to have lower waiting times. However, this is not always true as the waiting time can also be impacted by other factors such as the number of pending transactions in the network.

We propose different approaches to improve transaction latency by (1) increasing the block limit or size (2) reducing the block interval time, and (3) encouraging nodes to fill their blocks with transactions. We discuss the security issues that need to be considered when applying those approaches to existing permissionless blockchains. In future work, we aim to research a mathematical model for transaction latency to draw a more robust and complete analysis. In addition, we aim to conduct rigorous and extensive analysis to evaluate the proposed approaches with respect to performance and security.

Acknowledgements The author would like to thank Taibah University for its supervision support. All data are provided in full in the results section of this article.

Author Contributions The author has conceived and designed the experiments, performed the experiments,analyzed the data, performed the computation work, prepared figures and/or tables,authored or reviewed drafts of the article, and approved the final draft.

Funding There is no funding for this study.

Declarations

Conflict of interest The author declares that they have no conflict of interest

Ethical Approval This article does not contain any studies with human participants and/or animals performed by any of the authors.

Consent to Participate There is no informed consent for this study.

Consent for Publication Not applicable.

References

1. Kılıç, B., Özturan, C., Sen, A.: Parallel analysis of Ethereum blockchain transaction data using cluster computing. Cluster Comput. 1–14 (2022)
2. Fan, C., Ghaemi, S., Khazaei, H., Musilek, P.: Performance evaluation of blockchain systems: a systematic survey. IEEE Access **8**, 126927–126950 (2020)
3. Oliva, G.A., Hassan, A.E., Jiang, Z.M.J.: An exploratory study of smart contracts in the Ethereum blockchain platform. Empir. Softw. Eng. **25**(3), 1864–1904 (2020)
4. Kabla, A.H.H., Anbar, M., Manickam, S., Alamiedy, T.A., Cruspe, P.B., Al-Ani, A.K., Karupayah, S.: Applicability of intrusion detection system on Ethereum attacks: a comprehensive review. IEEE Access (2022)
5. Gavin Wood: Ethereum: a secure decentralised generalised transaction Ledger Byzantium Version 7e819ec, https://ethereum.github.io/yellowpaper/paper.pdf. Yellow Paper (2019)
6. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008). http://bitcoin.org/bitcoin.pdf. White Paper
7. Biais, B., Bisiere, C., Bouvard, M., Casamatta, C.: The blockchain folk theorem Toulouse School of Economics (2018)
8. Hjálmarsson, F.., Hreiarsson, G.K., Hamdaqa, M., Hjálmtỳsson, G.: Blockchain-based e-voting system. In: 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), pp. 983–986 (2018). IEEE
9. Ayed, A.B.: A conceptual secure blockchain-based electronic voting system. Int. J. Netw. Secur. Appl. **9**(3), 01–09 (2017)
10. Karamitsos, I., Papadaki, M., Al Barghuthi, N.B., et al.: Design of the blockchain smart contract: a use case for real estate. J. Inf. Secur. **9**(03), 177 (2018)
11. Vakilinia, I., Badsha, S., Sengupta, S.: Crowdfunding the insurance of a cyber-product using blockchain. In: 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), pp. 964–970 (2018). IEEE
12. Tanwar, S., Parekh, K., Evans, R.: Blockchain-based electronic healthcare record system for healthcare 4.0 applications. J. Inf. Secur. Appl. **50**, 102407 (2020)
13. Griggs, K.N., Ossipova, O., Kohlios, C.P., Baccarini, A.N., Howson, E.A., Hayajneh, T.: Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. J. Med. Syst. **42**(7), 1–7 (2018)
14. Sukhodolskiy, I., Zapechnikov, S.: A blockchain-based access control system for cloud storage. In: 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp. 1575–1578 (2018). IEEE
15. Atlam, H.F., Alenezi, A., Walters, R.J., Wills, G.B., Daniel, J.: Developing an adaptive risk-based access control model for the internet of things. In: 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and Ieee Smart Data (SmartData), pp. 655–661 (2017). IEEE
16. Dagher, G.G., Mohler, J., Milojkovic, M., Marella, P.B.: Ancile: privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. Sustain. Cities Soc. **39**, 283–297 (2018)
17. Ding, Y., Sato, H.: Bloccess: enabling fine-grained access control based on blockchain. J. Netw. Syst. Manag. **31**(1), 1–34 (2023)
18. Spain, M., Foley, S., Gramoli, V.: The impact of Ethereum throughput and fees on transaction latency during ICOS. In: International Conference on Blockchain Economics, Security and Protocols (Tokenomics 2019). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2020)
19. Gürcan, Ö.: Multi-agent modelling of fairness for users and miners in blockchains. In: International Conference on Practical Applications of Agents and Multi-Agent Systems, pp. 92–99. Springer (2019)
20. Möser, M., Böhme, R.: Trends, tips, tolls: a longitudinal study of bitcoin transaction fees. In: International Conference on Financial Cryptography and Data Security, pp. 19–33. Springer (2015)
21. Kawase, Y., Kasahara, S.: Transaction-confirmation time for bitcoin: a queueing analytical approach to blockchain mechanism. In: Proceedings of the International Conference on Queueing Theory and Network Applications, pp. 75–88. Springer (2017)

22. Yasaweerasinghelage, R., Staples, M., Weber, I.: Predicting latency of blockchain-based systems using architectural modelling and simulation. In: Proceedings of the 2017 IEEE International Conference on Software Architecture, pp. 253–256 (2017). IEEE
23. Kawase, Y., Kasahara, S.: Priority queueing analysis of transaction-confirmation time for bitcoin. J. Ind. Manag. Optim. **16**(3), 1077 (2020)
24. Wilhelmi, F., Barrachina-Muñoz, S., Dini, P.: End-to-end latency analysis and optimal block size of proof-of-work blockchain applications (2022). arXiv:2202.01497
25. de Azevedo Sousa, J.E., Oliveira, V., Valadares, J., Dias Goncalves, G., Moraes Villela, S., Soares Bernardino, H., Borges Vieira, A.: An analysis of the fees and pending time correlation in Ethereum. Int. J. Netw. Manag. **31**(3), 2113 (2021)
26. Zhang, L., Lee, B., Ye, Y., Qiao, Y.: Evaluation of Ethereum end-to-end transaction latency. In: 2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5 (2021). IEEE
27. Guo, J., Jiang, Z., Li, L., Bian, J.: Mathematical modeling of transaction latency on Ethereum. In: 2021 IEEE International Conference on Joint Cloud Computing (JCC), pp. 34–37 (2021). IEEE
28. Pierro, G.A., Rocha, H.: The influence factors on Ethereum transaction fees. In: 2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB), pp. 24–31 (2019). IEEE
29. Pierro, G.A., Rocha, H., Tonelli, R., Ducasse, S.: Are the gas prices oracle reliable? a case study using the ethgasstation. In: 2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE), pp. 1–8 (2020). IEEE
30. Alharby, M., van Moorsel, A.: Blocksim: an extensible simulation tool for blockchain systems. Front. Blockchain **3**, 28 (2020)
31. Bertoli, M., Casale, G., Serazzi, G.: JMT: performance engineering tools for system modeling. ACM SIGMETRICS Perform. Eval. Rev. **36**(4), 10–15 (2009)
32. Pontiveros, B.B.F., Torres, C.F., State, R.: Sluggish mining: profiting from the Verifier's Dilemma (2018). https://fc19.ifca.ai/wtsc/SluggishMining.pdf
33. Alharby, M., Lunardi, R.C., Aldweesh, A., van Moorsel, A.: Data-driven model-based analysis of the Ethereum verifier's dilemma. In: 2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 209–220 (2020). IEEE

Maher Alharby received a Ph.D. degree in computer science from Newcastle University in 2020. He has been awarded the 2020 SPEC Kaivalya Dixit Distinguished Dissertation Award for his Ph.D. thesis on "Models and Simulation of Blockchain Systems". He is currently an Assistant Professor in the College of Computer Science and Engineering, Taibah University, Saudi Arabia. His research fields include blockchain technology, smart contracts, usable security, network security, and cryptography. A list of his research publications can be found at shorturl.at/bcsN4.