A STUDY ON PERFORMANCE MODELING AND ASSURANCE OF CROSS/PERMISSIONLESS/PERMISSIONED CHAINS

By

ZUQIANG KE

Bachelor of Engineering in Computer Science and Technology
Shanghai Jiao Tong University
Shanghai, China
1987

Master of Science in Computer Science
Oklahoma State University
Stillwater, Oklahoma
2017

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
May, 2023

A STUDY ON PERFORMANCE MODELING AND

ASSURANCE OF

CROSS/PERMISSIONLESS/PERMISSIONED CHAINS

Dissertation Approved:

Dr. Nohpill Park

Dissertation Adviser

Dr. Christopher John Crick

Dr. Cong Pu

Dr. Wooyeol Choi

.

ACKNOWLEDGEMENTS

It would not have been possible to write this dissertation without the help and support of the people around me, and I only can mention some of them here.

Primarily, to my advisor Dr. N. Park for supporting and encouraging me during my Ph.D. studies here, particularly to him for allowing me to ask questions and challenging me to go beyond myself. His guidance helped me to overcome many difficulties during the dissertation research. I am grateful to him for keeping trust in my abilities and helped me with the completion of the dissertation.

Next, I would like to thank Dr. Christopher Crick, Dr. Cong Pu and Dr. Wooyeol Choi for kindly agreeing to serve on my dissertation committee. I sincerely acknowledge them for providing insightful comments on my work.

I would not have come this far without the unconditional love, support and sacrifices of my father Jintai Ke and my mother Yingzi Zheng who passed away five years ago. Despite being thousands of miles away, they gave me the strength to seek my own pathway.

Finally, this dissertation is dedicated to my wife, Hongyu Wang and my daughter Yiling Ke for always offering their love and support.

Name: Ke, Zuqiang

Date of Degree: MAY, 2023

Title of Study: A STUDY ON PERFORMANCE MODELING AND ASSURANCE OF CROSS/PERMISSIONLESS/PERMISSIONED CHAINS

Major Field: COMPUTER SCIENCE

Abstract: This research addresses and resolves the performance modeling and assurance issues across the full spectrum of blockchain protocols, from permissionless (Chapter II) and permissioned (Chapter III) to cross-chain (Chapter IV). In Chapter II, a queueing model for permissionless blockchains and validations is proposed with respect to specific yet practical characteristics of the blockchains such as Bitcoin and Ethereum, primarily in terms of the block size and its waiting time. A set of variables considered in this model lists the network traffic intensity, the maximum number of transactions in a block, the block time, and the transaction arrival rate, to mention a few. Numerical simulations are conducted, and the efficacy of the proposed model is validated in a quantitative yet practical manner versus Bitcoin and Ethereum. In Chapter III, a set of queueing models for permissioned blockchain, which is considered an emerging technology for a trustworthy decentralized network, is proposed. Hyperledger Fabric is a well-defined permissioned blockchain. It is constructed by various types of nodes, such as the nodes for endorsement, ordering, and commitment, to realize the decentralized nature of trustworthy network operations. Each type of node is characterized in terms of transaction/block queue size and waiting time, and the transaction/block arrival rates and the transaction/block service rates are considered for simulation purposes. It is taken into account how the arrival rates and the service rates co-influence the performance and how the number of channels impact the performance in order to ultimately facilitate a more dynamic way of optimization. The efficacy of the proposed models is demonstrated by the extensive numerical simulations and analyses. In Chapter IV, a cross-chain communication protocol and a m/Cox/1 queueing model-based performance model are proposed. Cross-chain communication considers two distinct types of transactions, such as an atomic swap and an inter-ledger asset transfer. They are controlled by different types of communication mechanisms, namely, Hashed Time Lock Contract (HTLC) based on a pre-image-based technique, and inter-ledger asset transfer, based on an asynchronous verification technique. In the performance model, a Poisson arrival process is assumed, and the two services for pre-commit, verify and commit are assumed to be exponential distributions. Lastly, the selection ratio of a communication protocol between HTLC and the inter-ledger asset transfer is assumed. Extensive numerical simulations are conducted to study the performance impact of changing the parameters, such as arrival rate, service rate, and the ratio of communication protocol. In this research, the proposed models provide a comprehensive yet fundamental basis to assure and ultimately optimize the design of blockchain technology-based applications in specific terms of performance.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER I

INTRODUCTION

In 2008, a paper entitled "Bitcoin: A Peer-to-Peer Electronic Cash System" [1] introduced the term "chain of blocks" into a peer-to-peer electronic cash network. The term "chain of blocks" evolved over the years into the word "blockchain" [2]. According to the National Institute of Standards and Technology (NIST) [3], "Blockchains are immutable digital ledger systems implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority." Blockchain is a decentralized model that enables peers to collaborate and build trust over a peer-to-peer business network (P2P) [69]. All the nodes in the blockchain need a voting process to make sure that a consensus is achieved on the transactions validating and block creating. After reaching consensus, a new block is appended with the hash of the previous block committed on each node, thus forming a distributed immutable digital ledger. Due to the consensus mechanisms and immutability of the ledger, blockchain can guarantee the fidelity and security of data records and eliminate the need for a third party. In more than 20 years of practice, blockchain technology has shown many benefits that the conventional database doesn't have, such as censorship, transparency, and traceability, etc. Besides cryptocurrency, blockchain technology is being used in many areas, such as improving supply chain transparency, creating loyalty programs for customers, etc. Nowadays, there are more than 1,000 active blockchain networks in the world. The blockchain technology-based applications have faced three major challenges: performance, scalability, and interoperability. It has the best result for using the different blockchains for different use cases and scenarios to meet the challenges (for more details see Appendix A.1). Therefore, it is essential to systematically analyze and evaluate the

blockchain's performance, scalability, and interoperability to advise the decision makers on the most suitable blockchain technology for their businesses and to help the software engineer develop new applications based on blockchain technology or migrate legacy applications to the blockchain network. In addition, a quantitative model that has traceability and predictability is the key foundation for establishing a trackable model that not only ensures the system operates in a safe and reliable manner but also guarantees the outputs are trustworthy and accurate by allowing for comprehensive monitoring and auditing. There are two analysis approaches that can be used to establish quantitative models and evaluate blockchain-based applications: empirical analysis [23] and analytical modeling [24]. Compared to empirical analysis, analytical modeling is a mathematical model-based approach to the study and interpretation of computer systems or networks. It has a closed-form solution to establish the relationships between actual and benchmark. The results that come from an analytical modeling can be used to answer an application scope question or make a design or development decision, and they provide a theoretical foundation for generating trackable models by using machine learning technologies.

1.1 Motivation

1.1.1 Benefits of Blockchain

Network security is one of the most important aspects that need to be considered. A higher-security network helps businesses reduce the risk of falling victim to data theft and sabotage. In a blockchain network, any valid transactions need to be agreed upon according to the consensus process, and each transaction is encrypted, immutable, and has a proper link to the prior transaction using a hashing method. Each node holds a copy of all the transactions ever performed on the network. Therefore, if any malicious actor ever wanted to make a change in the transaction, he wouldn’t be able to do so as other nodes would reject his request to write transactions to the network [5]. The consensus process and the transaction immutability make blockchain networks such as Bitcoin the most secure digital

system in the world and the most reliable monetary system ever invented. In more than a decade of practice, Bitcoin has never been hacked, and the counterfeit currency has never appeared on the network [6]. Peer-to-peer network services leverage technology to overcome the transaction costs of trust, enforcement, and information asymmetries that have traditionally been addressed by using trusted third parties. They help businesses reduce transaction costs, improve operational efficiency, and enhance data transparency. A blockchain consists of peers who are responsible for carrying out transactions and validating them to provide validation through a decentralized model. Once validated, each node keeps a copy of the transaction record. Blockchain participants can access the holdings and transactions of public addresses using a block explorer, which is used to search the blocks of a blockchain, their contents, and their relevant details. Therefore, blockchain makes the data transparent in a way that has not existed in financial systems, which is why many argue that blockchain could be used as the new standard for transparency [7]. With a shared, immutable ledger among members of a network, time-wasting record reconciliations are eliminated. It improves the execution of smart contracts [8] and then speeds up transactional performance. Also, smart contracts as an automatic procedure help execute business processes in an automated and trusted manner. Therefore, it improves the efficiency of executing transactions due to the automation of business processes. Business runs on information. The faster it's received and the more accurate it is, the better. Blockchain is ideal for delivering that information, not only because blockchain is the most secure network but also because blockchain can track orders, payments, accounts, production, and much more. Because business members share a single view of the truth, people can see all the details of a transaction from beginning to end, which gives greater confidence to create new efficiencies and opportunities [9].

1.1.2 Blockchain Challenges

Although blockchain networks provide a lot of benefits, there are concerns about whether their performance (generally measured as the average waiting time it takes for a transaction to be validated

and stored in each peer node) and scalability (the ability of that platform to support increasing loads of transactions as well as increasing the number of nodes in the network) would match up with the industry requirements. As Deloitte points out [21], "blockchain-based systems are comparatively slow." Blockchain's sluggish transaction speed is a major concern for enterprises that depend on high-performance legacy transaction processing systems. In a blockchain network, each peer node must perform computations and communicate with other peers to validate transactions, arrive at consensus, and update the state of the shared ledger. As the key process of peer-to-peer computing, communication and validation architecture, the consensus algorithms [12] deeply affect performance and scalability of blockchain and could make a bottleneck to the networks. Permissionless blockchain networks, such as Bitcoin and Ethereum [85], comprise millions of nodes and support cryptocurrency transactions. Due to the need for establishing trust between completely anonymous entities, a very computationally and time-intensive mining-based consensus mechanism is used. Thus, it takes a long time to achieve transaction finality, which results in transaction throughput in single digits [22]. This is why Bitcoin and Ethereum couldn’t obtain ideal performance in a peer-to-peer network with a huge number of nodes. On the other hand, permissioned blockchain networks that involve business-to-business and business-to-customer interactions between the partners are used to create a consortium with a limited number of nodes. Its performance can be much higher than the permissionless blockchain since a more efficient consensus algorithm can be used on a limited number of clients and nodes [22]. Meanwhile, it should be recognized that a permissioned blockchain network may lose its scalability in order to improve its efficiency. After Satoshi Nakamoto invented Bitcoin, there are at least 1,000 active blockchains with at least four types of blockchain networks in the world. The ability to establish cross-chain communication without losing performance and security is another issue that needs to be addressed. Also, it should be noted that the workload of different cross-chain protocols will not only impact the performance of cross-chain transactions but also impact the performance of participating blockchains.

1.1.3 Performance evaluation

From the above analysis, permissionless blockchain networks can gain better scalability, while permissioned blockchain networks have better performance on transaction execution. The reason seems to be that the consensus algorithms are not able to improve the performance while allowing the maximum number of nodes to participate in the business. Apparently, blockchain performance and scalability are negatively correlated. It is imperative to appropriately balance the two potentials so that the blockchains or blockchain technology-based applications can meet the business requirements. For the interoperability of blockchain networks, the performance and workload of cross-chain transactions are also traded off appropriately. It seems not easy to obtain a solution that can be accepted by both ends. Performance evaluation is defined as a formal and productive procedure to measure blockchains' performance, scalability, and interoperability based on business use cases. A quantitative performance evaluation can help decision makers and software developers systematically analyze the consensus algorithms and eventually find the balance point to improve the productivity of blockchain networks and enhance the performance of the consensus process. Performance evaluation can be classified into two general categories, namely, empirical analysis [23] and analytical modeling [24]. Empirical analysis is an evidence-based approach to the study and interpretation of information. The approach relies on benchmark, monitor, and simulation data, metrics, and results rather than theories and concepts (for more details see Appendix A.2). On the other hand, analytical modeling is a mathematical approach to business analysis that uses complex calculations that often involve numerous variables and factors. This type of analysis can be a powerful tool when seeking solutions to specific problems when used with proper technique and care [25] (for more details see Appendix A.3). Empirical analysis is a tedious and time-consuming benchmark generation and data/results collection process. It is also very expensive if it is necessary to capture all the details and then use them to generate a performance model. Moreover, empirical analysis generates a model based on benchmark data, which can only help people understand blockchain networks in their current

situation or in a test environment. It may not be possible to predict the blockchain networks very well. Compared to the empirical analysis, analytical modeling approach has the significant advantages, such as higher accuracy, more flexible, lower cost and more generalize etc. Stochastic modeling [26], which is one of the types of analytical modeling, has been widely used in artificial intelligence (AI) [27], machine learning (ML) [28], data analysis [29], and operations research [30]. The stochastic modeling represents a real-case simulation to understand the system better, study the randomness, and evaluate uncertain situations that define every possible outcome and how the system will evolve (for more details, see Appendix A.4). Hence, stochastic modeling helps professionals and investors make better management decisions and formulate their business practices to maximize profitability [31]. It also provides the theoretical basis for establishing a trackable model using machine learning technologies. In the first paper on blockchain, "Bitcoin: A Peer-to-Peer Electronic Cash System" [1], the author has already used the stochastic modeling approach to evaluate Bitcoin's security and performance. Since then, most of the research papers have used stochastic modeling to evaluate and predict blockchains, especially permissionless blockchains. All these facts have proven that stochastic modeling is the most natural approach to evaluating and predicting blockchains' performance, security, scalability, etc.

1.2 Objectives

This dissertation aims to use an analytical modeling approach to develop a set of stochastic models. These models will provide a quantitative framework [92] that helps decision makers and software developers evaluate the different configurations of blockchain networks and make trade-off decisions on the architecture design and the application development within affordable budget. This framework can also be used to estimate the performance impacts due to potential architectural changes in blockchain networks that the software engineers are considering for future releases. In this research, a stochastic model for evaluating and predicting permissionless blockchains' performance and scalability and a set of stochastic models to evaluate and predict the key performance indicators of

permissioned blockchains, as well as a stochastic model that is based on a m/Cox/1 queueing model for evaluating and predicting the performance of cross-chain asset transfer, will be proposed and validated based on an analytical modeling approach. Based on the stochastic models, several numerical simulations [32] will be conducted to reveal various preliminary performances of permissionless, permissioned, and cross-chain blockchains against reported real experimental data. Ultimately, these stochastic models can establish a theoretical foundation to provide the system analysis tools for the software developers to enhance the consensus algorithms and provide IT administrators with the management tools for maintaining the blockchain networks with higher efficiency and ultimately higher performance to combat the scalability issue.

1.3 Methodology

The primary contents of this dissertation are a literature review and analytical modeling, followed by numerical simulation and quantitative analysis (QA) [33], with the results of the analytical modeling, at the end, giving the conclusions of the analysis and the improvement suggestions. Stochastic modeling and numerical simulation are the key components of this research. Constraint identification and an initial proposal through a quantitative research approach [34] are the very first steps toward stochastic models and numerical simulations. This study will first review the performance bottlenecks and factors affecting scalability and interoperability. Based on this understanding, an initial idea will be developed to categorize the bottlenecks and the limitations for the purpose of analytical modeling. In the second stage, existing performance, scalability, and interoperability evaluation modeling methods will be identified based on a comprehensive review of current industry practices and academic research. Once the assumptions, parameters, and modeling techniques are identified, a conceptual model will be outlined. Meanwhile, an adequate amount of evidence will be provided to show the true assumptions and parameters. Based on the conceptual models, a stochastic modeling process is explored. The following steps show the critical procedure of the model generation:

- Designing a Markov chain or semi-Markov chain [35, 36] based on the transition probability with two constraints: it should be stochastic and irreducible.

- Solving the balance equations [37] or matrix-quadratic equation of the quasi-birth-and-death process (QBD) [38] and then forming a queue length distribution.

- Using the Markov chain or semi-Markov chain, formulate the equation for the average queue length.

- Using Little Law and the queue length, one can formulate the equation of average waiting time, or one can use Laplace–Stieltjes transform [39], which converts the queue length distribution to the job departure distribution, and then use the departure distribution to generate the equation of average waiting time.

In the third stage, after the equations for average queue length and average waiting time have been generated, a set of published experiment data will be used as the parameters. The numerical simulations will be conducted based on the parameters that came from the daily report of the real blockchain (Bitcoin and Ethereum main chains) and published experiment data, and then a set of charts will be drawn as the results of the numerical simulation. A quantitative analysis will be presented based on these results. Finally, a reasonable conclusion and future work will be given. Since this dissertation is an extension of the previous works, it will be conducted between November 2022 and April 2023.

1.4 Outline of Dissertation

This dissertation is organized as follows: In Chapter II, the performance modeling and numerical simulation for permissionless blockchains and their validation are discussed. The chapter presents a queueing model for mining-based public blockchains and validations with respect to specific yet practical characteristics of public blockchains such as Bitcoin and Ethereum, primarily in terms of

transaction queue size and block waiting time, as an alternative solution to the conventional industrial networks for the trustworthiness it offers. It is an extended version of our published paper [40].

In Chapter III, the performance modeling and numerical simulation for permissioned blockchains and their validation are discussed. This chapter presents quantitative models for the performance of endorsement, ordering, and commitment by primary types of nodes in permissioned blockchains with specific reference to Hyperledger Fabric [79]. It also provides numerical simulations based on the published data and provides a quantitative basis to assure and optimize the design of each type of node and, ultimately, the overall performance of the permissioned blockchain. This chapter is an extended version of our published paper [41].

In Chapter IV, two types of cross-chain communication protocols for isomorphic and heterogeneous blockchains and a performance model based on a m/Cox/1 queueing model are discussed. This chapter establishes a sound theoretical foundation to identify the interrelation and impacts between various design variables on the performance, which ultimately will reveal an optimal solution to a high-performance design of a cross-chain application that crosses the isomorphic and heterogeneous chains. This chapter is an extended version of our published paper [42].

Finally, chapter V concludes the dissertation and briefly describes how the models developed in this study can be integrated as a tool that designers, developers, and operators of blockchain networks could use to optimize system performance. It also outlines future avenues for research on performance aspects of blockchain networks.

CHAPTER II

ANALYTICAL MODELS FOR PERMISSIONLESS BLOCKCHAIN

Abstract

This research presents a queueing model for analyzing and evaluating permissionless blockchains with respect to specific yet practical characteristics of the blockchains such as Bitcoin and Ethereum, primarily in terms of transaction queue length and the waiting time, as an alternative solution to conventional computer networks for the trustworthiness it offers. A set of variables considered in this model lists the network traffic intensity, the maximum number of transactions in a block, the block time, and the transaction arrival rate, to mention a few. The proposed model provides a comprehensive yet fundamental basis to assure and ultimately optimize the design of blockchain technology-based applications in specific terms of performance. Numerical simulations have been conducted, and the efficacy of the proposed model is validated in a quantitative yet practical manner versus Bitcoin and Ethereum.

2.1 Introduction

Blockchain is a technology that operates an incorruptible digital ledger of transactions that can be programmed to record not just financial transactions but virtually everything of value [46]. Among many mining-based free joint blockchains, as a well-known cryptocurrency, Bitcoin [1] has gained a lot of attention, along with Ethereum [45, 47], and Ethereum provides additional features such as smart contracts and the EVM (Ethereum Virtual Machine). Today, these two blockchains represent the mainstream of permissionless blockchains that are open to anyone, where users can remain

anonymous, and no one entity controls the blockchains. To understand what these permissionless blockchains can be used for and how to improve their performance, security, and scalability, more adequate and quantitative performance models are urgently desired and even mandated.

Many blockchain technology-based applications have been developed ever since the inception of blockchain technology based on decentralized, digitalized, and distributed ledger technologies. These technologies provide desirable features by taking advantage of the decentralized communication architecture. However, architecture requires a time-consuming process to realize and fulfill the security, scalability, and integrity requirements. In most permissionless blockchain networks, the majority of consensus processes are considered and required as an excessively time- and energy-consuming procedure [51]. In fact, the effectiveness and efficiency of the applications are determined by the computing power of the consensus process and the energy consumption. In a mining-based consensus process, the miners who play the central role of the process are desirably equipped with hardware resources such as inexpensive computers and disks, and on the software side, an effective and efficient mining algorithm is required, which is supposed to consume less amount of resources for computation and ultimately optimize the cost for block creation and distribution. As such, an adequate and quantitative model to evaluate key performance indicators and parameters of blockchain technology is urgently sought to optimize the computing resources in a proactive manner, with which the design and development of consensus algorithms, blockchain architecture, and operating systems could be guided and improved upon.

A quantitative model and analysis on the performance of permissionless blockchain consensus process such as Proof-of-Work (PoW) [73] are to be addressed and resolved in this research, with respect to extensive and practical set of design and performance related variables. There have been few adequate yet practical stochastic models targeting the specific behavior of the permissionless blockchain consensus process found to the best of the authors' knowledge. In [53], a stochastic model has been developed to evolve the dynamics of block generation and analyze the impact of block

dissemination. By employing a combination of analytical calculations and simulation experiments to investigate both stationary and transient performance features, the results demonstrate a close agreement with measurements on a wide-area network and provide useful insight in addressing the security issues. But the research only focusses on analysis the block generation and dissemination, it is not able to provide the analysis of the performance on transaction level. In [52], it has been analyzed how Bitcoin uses a multi-hop broadcast to propagate the blocks through the network to update the ledger replicas. By using the gathered information, the research verifies the conjecture that block propagation delay in the network is the primary cause for blockchain forks and implements some changes to the Bitcoin protocol to reduce the risk of the forks. Since the study is about solving security issues such as block forks, it couldn't address the performance issues. In [55], a matrix-analytic approach is used for modeling transaction confirmation times and providing analytical expressions for the average number of transactions in queue and block. The research provides a more general model to analyze blockchain technologies and motivates a series of promising future research on the development of blockchain technologies. Although the model looks more general, it seems very difficult to evaluate by simulation. In [54], the authors develop and present a refined mathematical model for block arrivals, focusing on both the block arrivals during a period of constant difficulty and how the difficulty level evolves over time. Based on blockchain block arrival data and a stochastic analysis of the block arrival process, the model demonstrates that the process is not a homogeneous Poisson process which was suggested in the original Bitcoin paper. In [63], an analytical model is implemented to analyze the block mining process as a Poisson process with time-dependent intensity and derive predictions about block times for various hash-rate scenarios by analyzing Bitcoin's method to update the "network difficulty" as a mechanism to keep block times stable. Based on the analysis results, a new method is proposed to update the difficulty. The proposed method performs much better at ensuring stable average block times over longer periods of time.

In this research, a queueing model is proposed to trace the operations and their performance of transactions in permissionless blockchains in general. The model can be used to observe the realistic behaviors of mining-based consensus processes. The parametric simulations show that the model is accurate when it comes to the real-world scenarios. Moreover, the model can be used to perform extensive parametric studies to identify the influence factors of a node in the networks, which can ultimately help to improve the performance and reduce the energy consumption.

This research is organized as follows: the details of the proposed model and the performance measures will be presented after preliminaries and review; they will be followed by numerical simulation results to validate the efficacy; then the last section will conclude and discuss the work.

2.2 Preliminaries and Review

A permissionless blockchain is realized by the integration of several technologies, such as distributed contents and storage, cryptographic hashes, asymmetric digital signatures, and decentralized consensus algorithms. Permissionless blockchain works based on the principle of distrust yet through peer-to-peer communication. In order to keep all the network participants in synchronization and agreement on a task of interest, distributed consensus algorithms play a pivotal role in generation and verification of the transactions and the blocks. Security is another important purpose of using consensus algorithms. The algorithm must eliminate the possibility that a single entity can control the network to prevent malicious transactions and certain network errors such as double spending on the blockchain. Proof-of-Work (PoW) consensus algorithm is the most popular among the many mining-based permissionless blockchain consensus algorithms. The reputation has been gained for its capability to achieve the Byzantine Fault Tolerance (BFT) [75] objectives and to ensure honesty in the decentralized peer-to-peer networks without revealing identities.

Proof of Work consensus algorithm ensures that the blockchain is secure and decentralized, as no single entity can control the network. The computational power required for mining also plays a

deterrent role against malicious entities who may try to compromise the network. The mechanisms of the algorithm are listed as follows:

- Mining: A miner uses computational power to solve a complex mathematical problem to create a new block in the blockchain. This process is called mining. The miner who solves the problem first gets to add the new block to the blockchain and is rewarded with a set amount of cryptocurrency.
- Difficulty adjustment: The difficulty of the problem is adjusted periodically to maintain a consistent rate of block creation. This adjustment is done by changing the target value of the mathematical problem. If the target value is set too low, the problem is too easy to solve, and blocks will be created too quickly. If the target value is set too high, the problem is too hard to solve, and blocks will be created too slowly.
- Proof of work: To add a new block to the blockchain, the miner needs to provide proof of work. This proof is the solution to the mathematical problem. Once the proof is verified, the new block is added to the blockchain, and the miner is rewarded with cryptocurrency.
- Consensus: Proof of Work algorithm ensures that all nodes on the network agree on the state of permissionless blockchain. Nodes can verify that a block has been created through 'proof of work' and that the transactions in the block are valid.

Proof-of-Work algorithm functions as the methodology to approve or decline transactions in the permissionless blockchain. The process of transaction and block creation is computationally intensive and requires a significant amount of computational power and energy consumption. Figure 1 shows the process by which the transactions and blocks are created and ultimately chained together to create the blockchain.

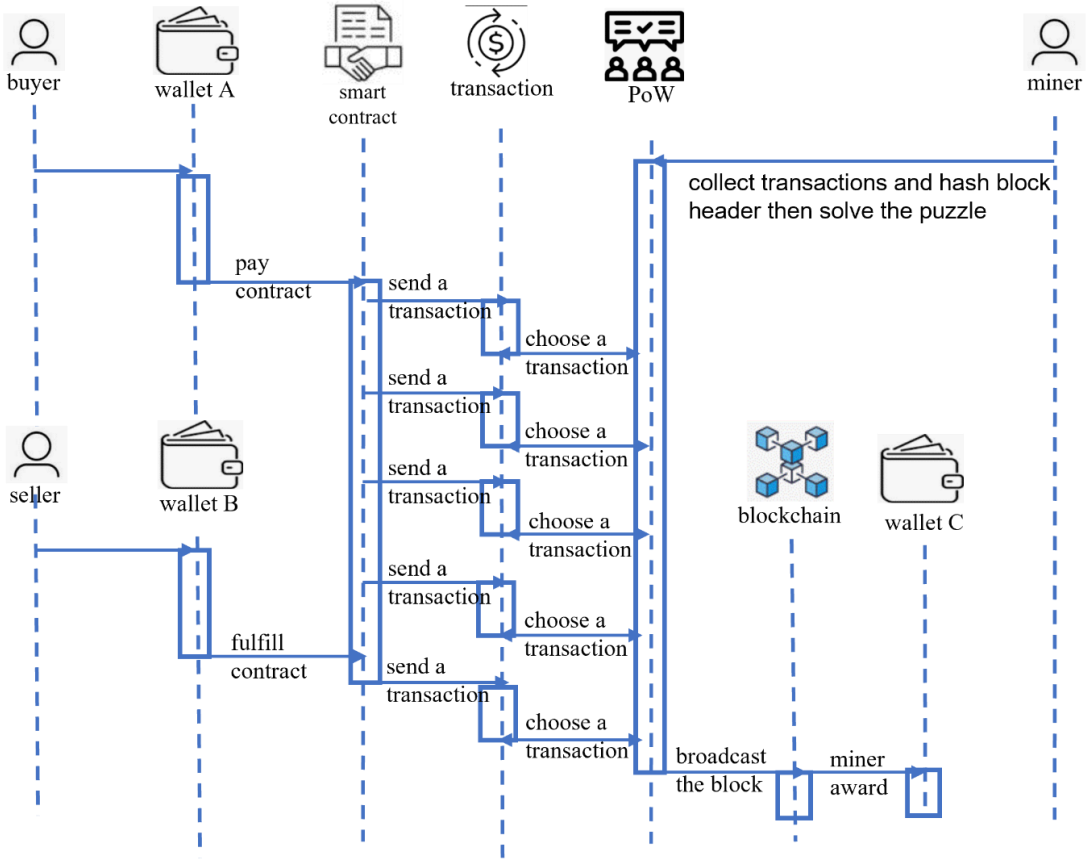

Figure 1. The process of Proof-of-Work [93]

- A user sends a transaction that is requested and authenticated to the blockchain network.
- A miner chooses a set of pending transactions to include in a new block in the blockchain.
- The miner hashes the block header, which includes the transactions data, a timestamp, and a reference to the previous block in the blockchain.
- The hash output is a random string of characters. The miner then compares the hash output to a target value set by the difficulty level of the blockchain network.
- If the hash output is less than the target value, the miner has successfully solved the 'proof of work'. If the hash output is greater than the target value, the miner must try again with a different nonce by adding a random value to the block header to create a new hash output.
- The miner continues this process of hashing the block header and adjusting the nonce until he/she finds a hash output that satisfies the target value.

- Once the miner finds a successful hash output, he/she broadcasts the block to the network as proof of work. Other nodes on the network can then verify the 'proof of work' and add the new block to their copy of the blockchain.
- The miner who successfully created the new block is rewarded with a certain amount of cryptocurrency as an incentive for contributing their computational power to the network.
- After the new block is distributed to all nodes across the network, the transaction is completed.

Proof-of-Work algorithm is a proven and secure consensus mechanism that has been used successfully in several major permissionless blockchains due to its significant benefits such as security, decentralization, and miner incentivization. However, the biggest disadvantage of the algorithm, which requires significant computational power and energy consumption to solve mathematical problems in authenticating blockchain transactions, precents the algorithm from broader acceptance in the blockchain network. In order to eliminate the worries, an adequate and quantitative model that can evaluate and assure key performance indicators and parameters of the algorithm is exigently desired.

2.3 Proposed Model and Analysis

This research specifically aims to evaluate the performance of the procedure for transaction validation and block creation in a mining-based consensus algorithm (Proof-of-Work), such as the number of newly generated blocks, the number of waiting transactions, and the overall waiting time. The proposed model tracks the transactions validation and new block generation process from the point of a transaction's arrival to the point at which a new block is generated and broadcasted into the network. Figure 2 shows the flow of the process that is considered the basis for the primary assumptions to be made for the proposed model. The process of the transaction validation and new block generation is assumed as following: The transactions arrive at the system at a fixed rate of λ with Poisson distribution and are stored in a queue. In the queue, the arriving transactions are collected into a pre-

block based on the limit of block time, the limit of block size, or the amount of gas in a pre-block. The sequence of transactions is ordered FIFO (first-in-First-Out) in a pre-block. Then, the pre-blocks are to be confirmed by a mining process in an interval of, namely, the block time. As soon as a mining process is completed, a new block is added to the local chain and disseminated across the blockchain network, and then the mining node goes back to take another pre-block from the queue and waits until the next round of mining comes up.

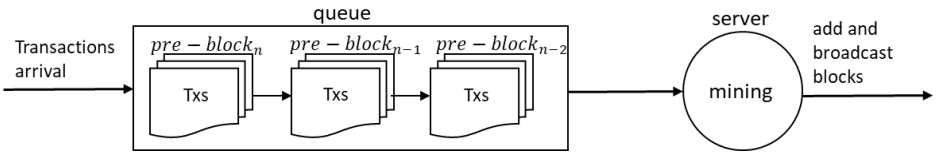

Figure 2. The process of block creation and distribution

$A_{j,k}$ is defined as a probability for j transaction arrivals during a time period t_k, and t_k involves several of the intervals $\Delta t = t_k - t_{k-1}$ with p probability and the assumption that at most one transaction arrives during this time interval. Since the transaction arrival is assumed as Poisson distribution, $A_{j,k}$ is a binomial probability which is j transaction arrivals during a time period t_k consisting of k time points given by:

$$A_{j,k} = \begin{cases} \binom{k}{j} p^j q^{k-j} & 0 \le j \le k \\ 0 & otherwise \end{cases} \quad (1)$$

where $q = 1 - p$ is probability of 0 transaction arrival. Since this is a binomial distribution [98], the expected number of transactions arrival $E(A_n) = kp = \lambda t_k = \lambda k \Delta t$, then get the transaction arrival rate:

$$\lambda = \frac{E(A_n)}{t_k} = \frac{kp}{k\Delta t} = \frac{p}{\Delta t} \quad (2)$$

The block time defines the time that it takes to validate the transactions and generate a new block (mining process). The expected block time is set at a constant to make sure the miners cannot impact the security of the network by adding more computational power. The average block time of the mining process is evaluated after a certain number of blocks, and if it is greater than the expected block time, then the difficulty level of Proof-of-Work algorithms will be reduced, and if it is less than the expected block time, then the difficulty level will be increased [65]. Therefore, without loss of generality, the time of mining a block (the block time) for all the nodes can be assumed as a constant that equals the expected block time ($E(U) = k\Delta t$), then the probability distribution of the mining process that contains k time points S_k, is given:

$$S_k = \begin{cases} 1 & if\ k\ is\ a\ constant \\ 0 & otherwise \end{cases} \tag{3}$$

, and the probability function k_j of the number of transaction arrivals during a mining process is given by:

$$k_j = \begin{cases} \sum_{k=0}^{\infty} A_{j,k} S_k & 0 \le j \le k \\ 0 & j < 0\ or\ j > k \end{cases} \tag{4}$$

after getting generating function of k_j and then substituting $k = \frac{E(U)}{\Delta t}$ and $S_k = 1$ in the function of k_j, obtain:

$$K(z) = \sum_{j=0}^{\infty} k_j z^j = \sum_{k=0}^{\infty} S_k (pz + q)^k = (pz + q)^{\frac{E(U)}{\Delta t}} \tag{5}$$

Following permissionless blockchain's consensus protocols (i.e., Proof-of-Work or Proof-of-Stake (PoS) [74]), find that the interval time between the adjacent mining process (the block time) is independent of each other. Therefore, there is an imbedded Markov chain within the time point of

starting nth block mining to starting the $(n+1)$th block mining. Therefore, obtain the state transition diagram as following:

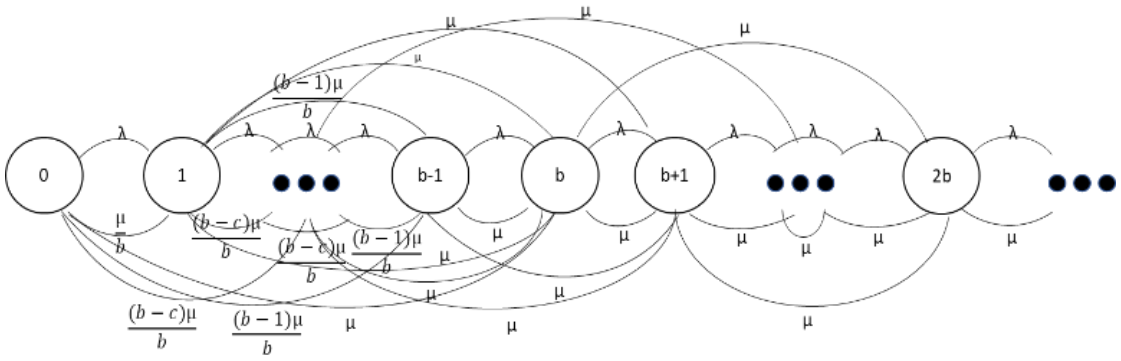

Figure 3. The rate diagram of the block-mining queue

$P_{i,j}^{(n)}$ is defined as a transition probability. It is associated with number of transactions i at the moment just before starting the nth mining process and transferring to number of transactions j at the moment just before start $(n+1)$th mining process, after nth block has been generated; and b is the maximum number of transactions in a block, then get the transition probability as following:

$$p_{i,j} = \lim_{n\to\infty} P_{i,j}^{(n)} = \begin{cases} k_j & 0 \le i \le b \\ k_{j-i+b} & b \le i \le j+b \\ 0 & i > j+b \end{cases} \tag{6}$$

when $0 \le i \le b$, the transition from i to j can be explained as i transactions are going to be validated; when $b \le i \le j+b$, the transition means that b transactions are going to be validated, but $i-b$ transactions still stay in the queue. In order to let the system be in a steady state, it is necessary to assume the network traffic intensity $\rho = \frac{\lambda E(U)}{b} < 1$. From equation (6), obtain an imbedded Markov chain probability of number of transactions (j) in the queue at the moment just before start $(n+1)$th mining process as following:

$$\pi_j = \sum_{i=0}^{\infty} \pi_i \, p_{i,j} = k_j \sum_{i=0}^{b-1} \pi_i + k_{j-i+b} \sum_{i=b}^{\infty} \pi_i \tag{7}$$

multiplying by z^j and summing over j, obtain the generating function $P(z)$:

$$P(z) = \sum_{j=0}^{\infty} \pi_j z^j = \sum_{j=0}^{\infty} \left\{ k_j \sum_{i=0}^{b-1} \pi_i + k_{j-i+b} \sum_{i=b}^{\infty} \pi_i \right\} z^j$$

$$= \frac{\sum_{j=0}^{\infty} k_j z^j \sum_{i=0}^{b-1} \pi_i z^b + \sum_{j=0}^{\infty} \sum_{i=b}^{\infty} \pi_i k_{j-i+b} z^{j+b}}{z^b}$$

$$= \frac{\sum_{j=0}^{\infty} k_j z^j \sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right) + \sum_{j=0}^{\infty} k_j z^j \sum_{i=0}^{\infty} \pi_i z^i}{z^b}$$

$$= \frac{K(z)\left(\sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right) + P(z)\right)}{z^b},$$

$$P(z) = \frac{\sum_{i=0}^{b-1} \pi_i (z^b - z^i)}{\frac{z^b}{K(z)} - 1} \tag{8}$$

In order to let $\sum_{j=0}^{\infty} \pi_j = 1$, must have $P(1-) = 1$, which, when applied to equation (8), and then give the condition:

Since $P(1-) = 1$,

$$\frac{z^b}{K(z)} - 1 = \sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right)$$

Derive both parts:

$$\frac{\left[z^b - (pz+q)^{\frac{E(U)}{\Delta t}}\right]' \left((pz+q)^{\frac{E(U)}{\Delta t}}\right) - \left[(pz+q)^{\frac{E(U)}{\Delta t}}\right]' \left(z^b - (pz+q)^{\frac{E(U)}{\Delta t}}\right)}{\left((pz+q)^{\frac{E(U)}{\Delta t}}\right)^2}$$

$$= \left[\sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right)\right]'$$

$$\frac{(bz^{b-1} - (pz+q)^{\frac{E(U)}{\Delta t}-1}(\frac{pE(U)}{\Delta t}))\left((pz+q)^{\frac{E(U)}{\Delta t}}\right) - \left[(pz+q)^{\frac{E(U)}{\Delta t}}\right]'(z^b - (pz+q)^{\frac{E(U)}{\Delta t}})}{((pz+q)^{\frac{E(U)}{\Delta t}})^2}$$

$$= \sum_{i=0}^{b-1} \pi_i \left(bz^{b-1} - iz^{i-1}\right)$$

Substitute $z = 1$ into above equation, get:

$$b - \frac{pE(U)}{\Delta t} = \sum_{j=0}^{b-1}(b-j)\pi_j$$

Substitute $E(U) = \frac{b}{\mu}, \rho = \frac{\lambda}{\mu}$ and $p = \frac{\lambda}{\Delta t}$, get:

$$\sum_{j=0}^{b-1}(b-j)\pi_j = b(1-\rho)$$

$$\sum_{j=0}^{b-1}(b-j)\pi_j = \sum_{j=0}^{b-1} b\pi_j - \sum_{j=0}^{b-1} j\pi_j = b(1-\rho), \qquad where\ \mu = \frac{b}{E(U)}\ and\ \rho = \frac{\lambda}{\mu} \quad (9)$$

According to Norman T.J. Bailey [48], if $K(z)$ in equation (8) is regular for $|z| < 1 + \delta$ (where $\delta > 0$ is sufficiently small) and if the first b probabilities π_j $(0 \le j \le b-1)$ can be chosen so that zeros of the numerator and denominator in this region coincide, then $P(z)$ will be regular for $|z| < 1 + \delta$ and will possess a power-series expansion absolutely convergent at $z = 1$, whose coefficients satisfy equation (7), hence,

$$z^b = K(z) = (pz+q)^{\frac{E(U)}{\Delta t}} \quad (10)$$

$$\sum_{j=0}^{b-1} \pi_j (z_i^b - z_i^j) = 0 \quad (11)$$

Equation (10) has simple roots $z_i, 1 \leq i \leq b-1$, other than $z_i = 1$ within the unit circle when $1 > \rho > p$; and equation (8) is completely determined by taking equation (9) and equation (10), and then equation (11) determines the solutions of the probabilities $\pi_0, \pi_1, \pi_2 \ldots \pi_{b-1}$.

2.4 Performance Measures

Two important performance measures are discussed in this section: the average number of transactions in the queue and the average transaction confirmation time (waiting time), which is the duration from the time a transaction is issued by a user until it is validated by the blockchain. They can be used in numerical simulation to reveal various preliminary performances of permissionless blockchains.

Since the numerator of equation (8) is a polynomial of degree b, it can be written as:

$$\sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right) = \sum_{i=0}^{b-1} \pi_i (z-1) \prod_{j=1}^{b-1} \left(z - z_j\right) \tag{12}$$

Removing π_i from $b+1$ equations in equations (9), (11) and let $y = \sum_{i=0}^{b-1} \pi_i$, gives:

$$\begin{vmatrix} 1 & 1 & \cdots & 1 & y \\ b & b-1 & \cdots & 1 & b(1-\rho) \\ z_1^b - 1 & z_1^b - z_1 & \vdots & z_1^b - z_1^{b-1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ z_{b-1}^b - 1 & z_{b-1}^b - z_{b-1} & \cdots & z_{b-1}^b - z_{b-1}^{b-1} & 0 \end{vmatrix} \begin{bmatrix} \pi_0 \\ \vdots \\ \pi_{b-1} \\ -1 \end{bmatrix} = 0$$

Subtracting each of the 2nd to the bth columns from the columns on its left and then taking out factor $\prod_{j=1}^{b-1}(z_j - 1)$, leads to:

$$\begin{vmatrix} 0 & 0 & \cdots & 0 & y \\ 1 & 1 & \cdots & 1 & b(1-\rho) \\ 1 & z_1 & \cdots & z_1^{b-1} & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & z_{b-1} & \cdots & z_{b-1}^{b-1} & 0 \end{vmatrix} \begin{bmatrix} \pi_0 \\ \vdots \\ \pi_{b-1} \\ -1 \end{bmatrix} = 0$$

, hence:

$$y\begin{vmatrix}1 & 1 & \cdots & 1\\ 1 & z_1 & \vdots & z_1^{b-1}\\ \vdots & \vdots & \vdots & \vdots\\ 1 & z_{b-1} & \cdots & z_{b-1}^{b-1}\end{vmatrix} = (-1)^{b+1}b(1-\rho)\begin{vmatrix}1 & z_1 & \cdots & z_1^{b-2}\\ 1 & z_2 & \vdots & z_2^{b-2}\\ \vdots & \vdots & \vdots & \vdots\\ 1 & z_{b-1} & \cdots & z_{b-1}^{b-2}\end{vmatrix}$$

Cancelling the common factors, gives:

$$y = \sum_{i=0}^{b-1}\pi_i = \frac{b(1-\rho)}{\prod_{j=1}^{b-1}(1-z_j)} \tag{13}$$

Substituting equations (5), (12) and (13) into equation (8), obtain:

$$P(z) = \frac{b(1-\rho)(z-1)\prod_{j=1}^{b-1}\left(\frac{(z-z_j)}{(1-z_j)}\right)}{\frac{z^b}{(pz-q)^{E(U)/\Delta t}} - 1} \tag{14}$$

According to M.L. Chaudhry [49], differentiating equation (14) with respect to z and its limit as $z \to 1-$, obtain a result which becomes indeterminate even after using L'Hospital's rule [66] once, so using L'Hospital's rule a second time, and then obtain the mean number of transactions in the queue just before starting the new mining process:

$$L^+ = \frac{\phi^{(2)}(1) - \psi^{(2)}(1)}{2\psi^{(1)}(1)} = \sum_{j=1}^{b-1}\frac{1}{1-z_j} + b\rho + \frac{\lambda^2((E(U))^2 - E(U)\Delta t) - b(b-1)}{2b(1-\rho)} \tag{15}$$

where $= \frac{\lambda}{\mu}$, $\mu = \frac{b}{E(U)}$, z_j are roots of equation (10) and $1 > \rho > p$; $\phi(z), \psi(z)$ are numerator and denominator of equation (14) respectively.

Since the inter-occurrence of mining process $U_n = \sigma_n' - \sigma_{n-1}', n = 1,2,3 \ldots, \sigma_0' = 0$ between successive minding process occurrences are positive independent, it can be thought of this process $\{N^a(t),\ t \geq 0\}$ as a typical renewal process [67]. Let $r(t)$ be the time measured from t, the instant of starting to observe the process to the begin of next minding process at $W_{N^a(t)+1}$, then obtain:

$$r(t) = W_{N^a(t)+1} - t, \quad t > 0 \tag{16}$$

Using the renewal theorem [67] about expected residual life time [17], obtain mean remaining time W_r for the new arrivals that are waiting to be mined at next minding process:

$$W_r = \lim_{t\to\infty} E\big(r(t)\big) = \frac{E(U^2)}{2E(U)} = \frac{E(U)}{2} + \frac{Var(U)}{2E(U)} = \frac{E(U)}{2} \tag{17}$$

, where $Var(U) = 0$, since the inter-occurrence $U_n = k\Delta t$ is deterministic.

Define L^- as the mean number of transactions in the queue just after starting a mining process, then get the total amount of $L^- E(U)$ units of waiting for L^- transactions that will be mined in the mining processes:

$$W_q^- = \int_0^\infty L^- \, u dS(u) \tag{18}$$

Since, the mean of waiting to be mined for new arrivals is W_r, then get the total amount of the units of waiting for new arrivals ($\lambda E(U) = \lambda u$) during next mining process:

$$W_a = \int_0^\infty W_r \lambda u dS(u) = \int_0^\infty \frac{\lambda u^2}{2} dS(u) \tag{19}$$

, where $dS(u)$ $(0 \le u < \infty)$ is density of probability distribution of the block mining time. Since sum of the total amount of the units of waiting is $W_q^- + W_a = \int_0^\infty (L^- u + \frac{\lambda u^2}{2}) dS(u)$, then get the mean number of transactions in queue:

$$L_q = \frac{W_q^- + W_a}{E(U)} = \frac{L^- E(U) + \frac{\lambda((E(U))^2 + Var(U))}{2}}{E(U)} \tag{20}$$

Since the mean number of arrivals during a block time equals to $\lambda E(U)$, the mean number of transactions in the queue just after starting a mining process should be: $L^- = L^+ - \lambda E(U)$. Since the

average mining time is deterministic, $Var(U) = 0$. According to equation (9), have $\lambda E(U) = b\rho$, then get the mean waiting time W_q by utilizing Little's law [68], and get the mean number of transactions in queue L_q by substituting equation (15) and $Var(U) = 0$ into equation (20):

$$W_q = \frac{L_q}{\lambda} = \frac{(L^+ - \lambda E(U))E(U) + \frac{\lambda(E(U))^2}{2}}{\lambda E(U)} = \frac{L^+ - \frac{b\rho}{2}}{\lambda} \tag{21}$$

$$L_q = \sum_{j=1}^{b-1} \frac{1}{1 - z_j} + \frac{b\rho}{2} + \frac{\lambda^2 \left(\left(E(U) \right)^2 - E(U)\Delta t \right) - b(b-1)}{2b(1-\rho)} \tag{22}$$

Since the average waiting time of system $W = W_q + E(U)$, obtain W:

$$W = \frac{1}{\lambda} \left[\sum_{j=1}^{b-1} \frac{1}{1 - z_j} + \frac{3b\rho}{2} + \frac{\lambda^2 \left(\left(E(U) \right)^2 - E(U)\Delta t \right) - b(b-1)}{2b(1-\rho)} \right] \tag{23}$$

2.5 Numerical Simulation and Results

The primary objective of the simulation is to reveal various preliminary performance of permissionless blockchain of interest such as the average queue length and the transaction waiting time.

The following graphs (Figure 4 and Figure 5) show $L_{Bitcoin}$ and $W_{Bitcoin}$ (from equations (22), (23) respectively) as the results of the queue length and the waiting time respectively, given the traffic density ρ, and under the conditions of the average block-mining time $E(U)$, the average transaction arrival rate λ and the time interval Δt.

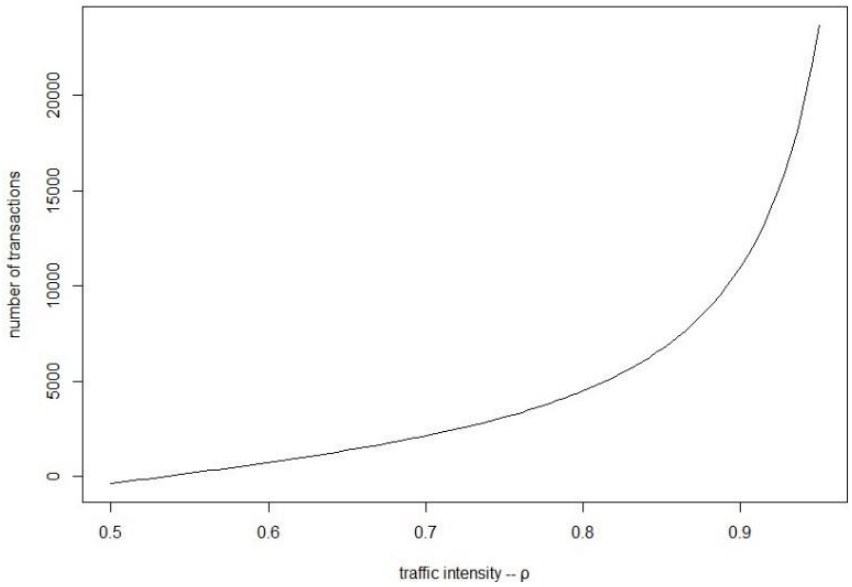

Figure 4 Average Queue Length by increasing traffic intensity (Bitcoin)

Figure 4. shows the length of queue in Bitcoin is around *10k* transactions in the queue when traffic intensity $\rho = \frac{\lambda E(U)}{b} = \frac{3 \times 600}{2000} = 0.9$, under the condition the block size $b = 2000$ [57], arrival rate = 3/s and expected block time =10 minutes [58]. This outcome is identical with the expectation that the average number of transactions in mempool is around 10k [61]. Figure 4 also shows the lower ρ the better the performance due to reducing the queue length. From the equation of ρ, shows that the bigger b the smaller ρ will have, when λ and $E(U)$ are constants. Therefore, this figure implies that the mining process can be improved by increasing the block size b.

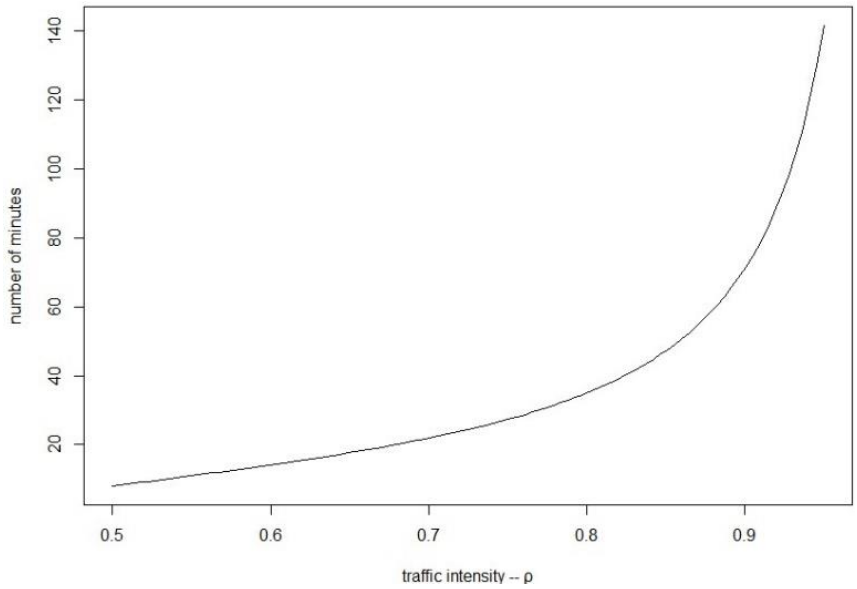

Figure 5. Average Waiting Time by increasing traffic intensity (Bitcoin)

Figure 5. shows the average waiting time is around 20 minutes if the traffic intensity $\rho = 0.7$. This result seems to reflect the idea of the argument [50] that explained why it always needs more than 20 minutes for a transaction to be mined into a particular block, rather than 10 minutes which equals to the expected block time. This diagram also shows that the waiting time can be increased by growing the traffic intensity ρ.

The following graphs (Figure 6 and Figure 7) show $L_{Bitcoin}$ and $W_{Bitcoin}$ (from equations (22), (23) respectively) as the results of the queue length and the waiting time respectively, given the block size b, and under the conditions of the average block-mining time $E(U)$, the average transactions arrival rate λ and the time interval Δt.

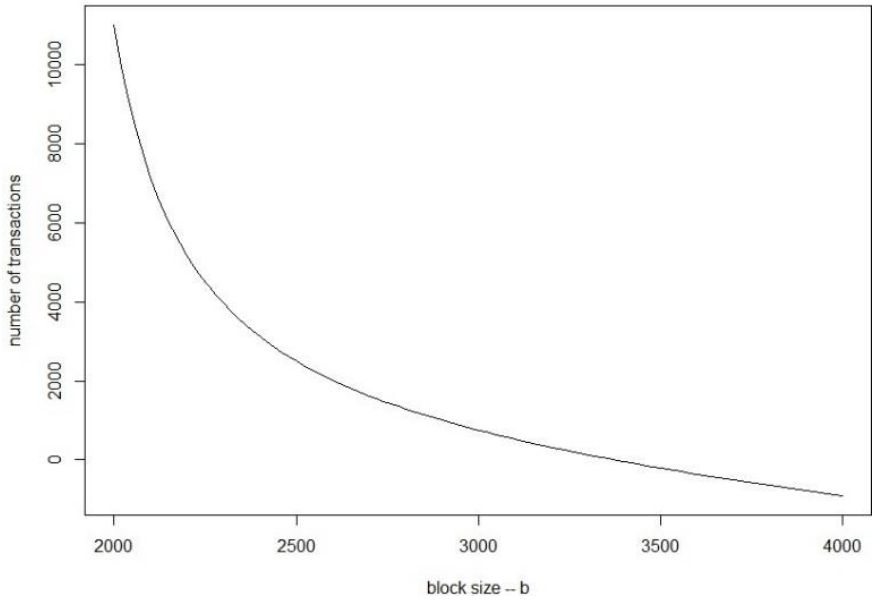

Figure 6 Average Queue Length by increasing block size (Bitcoin)

Figure 6 shows the queue length of Bitcoin is reduced rapidly while the size of block is growing bigger. By increasing the block size from 2k to 4k, the average number of transactions in queue is reduced from 10k to 0. This result seems to answer why the Bitcoin community liked to expand the block size from 1MB to 2MB [62].

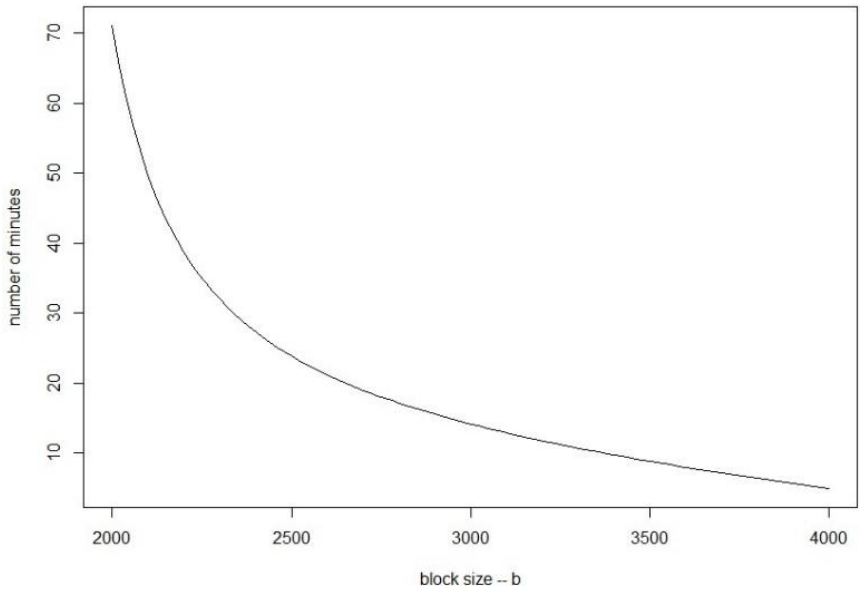

Figure 7. Average Waiting Time by increasing block size (Bitcoin)

Figure 7 shows that increasing the size of the block from 2k to 4k can improve mining process performance, and then reduce the average waiting time of Bitcoin from an hour to 10 minutes. Since increasing block size b can low the traffic intensity ρ if λ and $E(U)$ are constants, The results in the diagram consistent with the performance tracking in Figure 8.

The following graphs (Figure 8 and Figure 9) show $L_{Ethereum}$ and $W_{Ethereum}$ (from equations (22), (23) respectively) as the results of the queue length and the waiting time respectively, given the traffic density ρ, under the conditions of the average block mining time $E(U)$, the average transaction arrival rate λ and the time interval Δt.

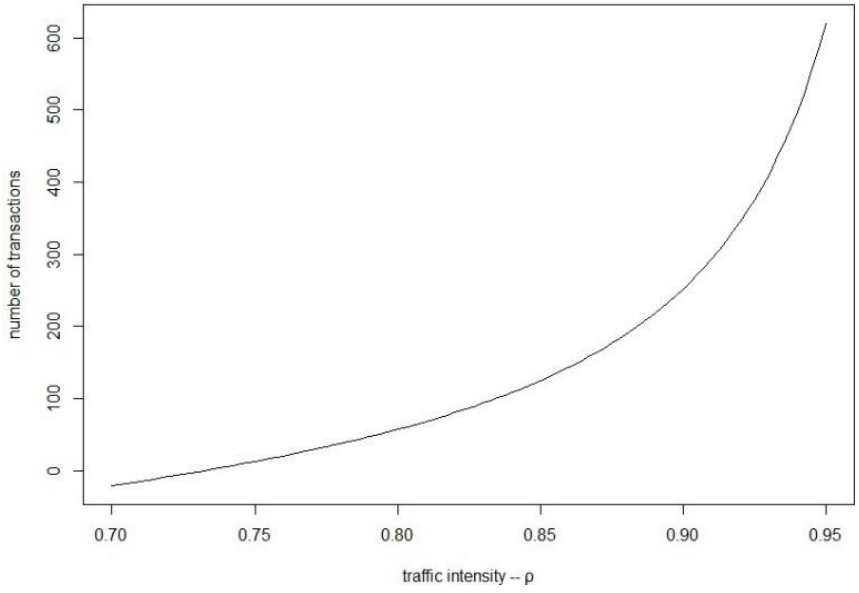

Figure 8 Average Queue Length by increasing traffic intensity (Ethereum)

Figure 8 shows that the queue length being increased by the traffic intensity growing. When traffic intensity $\rho = 0.9$, the average number of transactions in the queue is around 250. This result shows Ethereum network may be more efficient than Bitcoin due to the outcome that the arriving transaction does not need to wait to next mining process at the traffic intensity that Bitcoin does. Since λ and

$E(U)$ are constants, the traffic intensity determinates the block size b due to $\rho = \frac{\lambda E(U)}{b}$. Therefore, Figure 8 implies that the smaller block size the lower the mining performance.

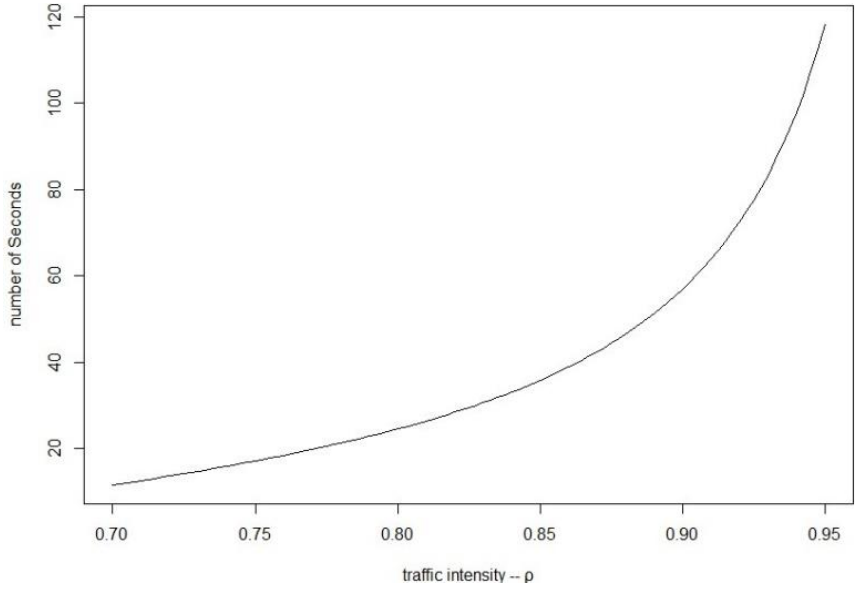

Figure 9 Average Waiting Time by increasing traffic intensity (Ethereum)

Figure 9 shows that the expected waiting time is around 50s when traffic intensity $\rho = \frac{6\times15}{100} = 0.9$. This conclusion seems to reflect why Ethereum takes about 53 seconds on average to be confirmed by a miner and included in the blockchain [56] under the condition that the arrival rate is 6/s, the expected bock time is 15s [59] and the block size is 100, This suggests that Ethereum should take in the range of 14 to 20 seconds, which equals the expected waiting time of the transaction confirmation.

The following graphs (Figure 10 and Figure 11) show $L_{Ethereum}$ and $W_{Ethereum}$ (from equations (22), (23) respectively) as the results of the queue length and the waiting time respectively, given the block size b (maximum number of transactions in a block), and under the conditions of the average block-mining time $E(U)$, the average transactions arrival rate λ and the time interval Δt.

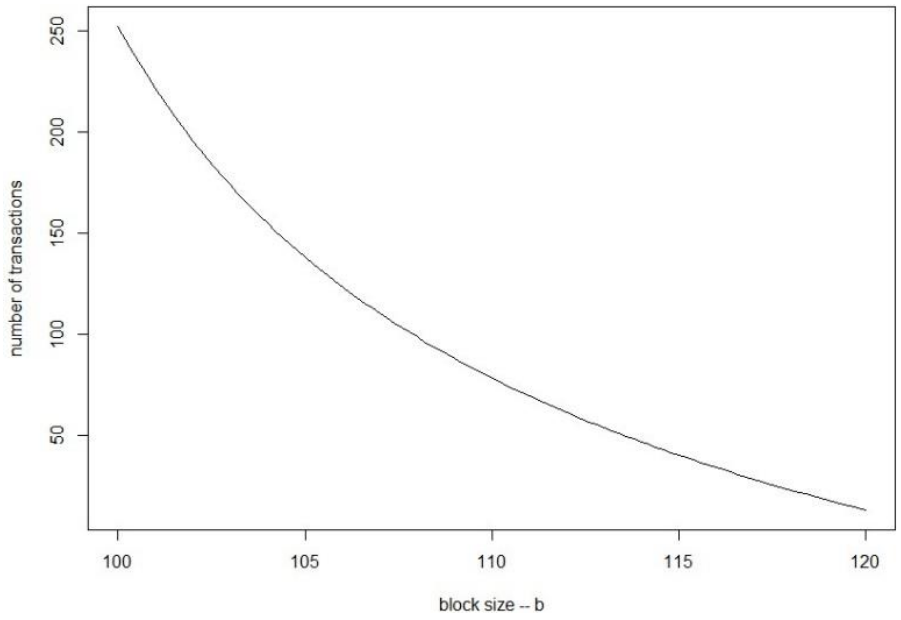

Figure 10 Average Queue Length by increasing block size (Ethereum)

Figure 10 shows the queue length of Ethereum is reduced rapidly while the size of the block is growing. After increasing the block size from 100 to 120, the average size of the queue dropped from 250 to 20. This result implies that the performance of the mining is almost linearly improved by increasing the block size.

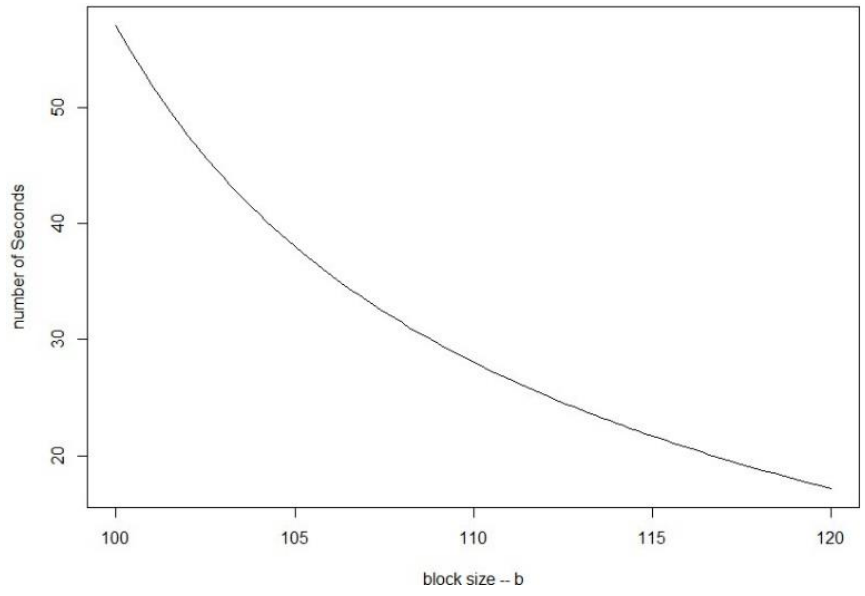

Figure 11 Average Waiting Time by increase block size (Ethereum)

Figure 11 shows that the average waiting time decreased from more than 50 minutes to less than 20 minutes by increasing the block size from 100 to 120. This result shows that increasing the size of the block can dramatically improve mining performance and reduce the waiting time for transaction confirmation.

2.6 Conclusions

This research has presented a queueing model to evaluate the performance of mining-based permissionless blockchains with respect to the number of transactions in the queue and the overall waiting time for a transaction confirmation, along with a practical validation work. A set of variables considered in this model and simulated include the number of transactions in queue, the waiting time for a transaction confirmation, as well as block mining distribution, along with a few constants such as transaction arrival rate and block size. Parametric simulations have been conducted, which demonstrate that the results are in great agreement. By giving different traffic intensities ρ in an ascending order, the queue length and waiting time increase very quickly. However, by giving different block size b in an ascending order, the queue length and the waiting time decreased very drastically. These trends imply that traffic intensity and block size are the two major factors that can affect the performance of permissionless blockchain-based applications. This conclusion encourages us to lower the system loading rate ρ by reducing the block time during the consensus process (i.e., Proof-of-Work) through the algorithm improvement to mitigate the increasing difficulty otherwise. It is also suggested to reduce the wait time by increasing the block size. However, these two suggestions are likely to encounter a technical challenge: how to maintain permissionless blockchains that are both decentralized and secure under the pressure of stringent performance and scalability requirements? Furthermore, besides the simulations that have been conducted in this research, which have shown the impacts of the traffic intensity ρ and block size, the proposed model can also be used to test the impacts of changing arrival rate and the block-time. Besides Bitcoin and Ethereum, the proposed model is readily extensible to other mining-based permissionless blockchains. In

conclusion, the proposed model provides a comprehensive yet theoretical basis to assure and ultimately optimize the design of permissionless blockchain-based applications with respect to cost and performance.

CHAPTER III

ANALYTICAL MODELS FOR PERMISSIONED BLOCKCHAIN

Abstract

The performance of a permissioned blockchain, namely Hyperledger Fabric [79], is modeled and analyzed in a quantitative manner in this research. Various types of nodes contribute to the performance of Hyperledger Fabric, and each of those is modeled and tracked along the operational flow of the permissioned blockchain. There are nodes for endorsement, ordering, and commitment to realize the decentralized network operations. A quantitative model for each type of node is proposed along with the numerical analysis. Each type of node is characterized in terms of transaction/block queue size and waiting time, and the transaction/block arrival rates and the transaction/block service rates are considered for simulation purposes. The analysis is conducted in this research to particularly demonstrate how the arrival rates and the service rates co-influence the performance and how the number of channels impacts the performance, in order to ultimately facilitate a more dynamic way of optimization that takes the co-relation across different types of nodes into account. The major contribution of this research to the field of computer science is the creation of a series of queuing models to evaluate the performance of different types of nodes, including: Chaincode Execution and Endorsement, Block Creation and Delivery, Transaction Validation and Block Committing, and Transaction Processing.

3.1 Introduction

Blockchain is an immutable digital ledger system implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority [3]. A permissioned blockchain is a distributed digital database that is shared among the nodes of a peer-to-peer network [69]. It stores transactional data (transactions) in a digital ledger (block). A transaction is a transfer of value on the permissioned blockchain. It is organized in the form of a block and distributed to every node in the network. Whenever a transaction is added to the network, every node validates and processes it individually [70]. A block is basically a collection of transactions. Each block consists of a block header with certain data, the hash value of the current block, and the hash value of the previous block. Whenever a block is created, it automatically attaches a hash value to it. Any changes that are made to the block can be identified using hash values [70]. The decision to add a block to the chain is made by consensus in a permissioned blockchain network. Consensus is a group decision-making process in which participants develop and decide on proposals with the aim, or requirement, of acceptance by all. The focus on establishing agreement with at least the majority or the super-majority and avoiding unproductive opinions differentiates consensus from unanimity, which requires all participants to support a decision [71]. A consensus is achieved through a consensus algorithm, which is a core part of permissioned blockchain. The algorithm ensures that whatever local copy each entity has is synchronized with others and is the latest one [72]. Byzantine Fault Tolerance is the consensus algorithm that has been widely used in permissioned blockchains.

Nowadays, permissioned blockchain has become an incredibly prominent and promising technology altogether. Proponents of permissioned blockchain suggest that the technology has virtually unlimited applications across any number of different fields such as banks, energy, the internet of things (IoT) [78], health, media, and more [76]. One of the most prominent permissioned blockchain technology-based applications is Automating trust. It is one of eight

emerging technologies that people need to know. Automating trust that combines the blockchain with artificial intelligence (AI) [27] and IoT has the potential to ensure the authenticity of data, verify identities, and enable secure multiparty transactions. The convergence of these technologies eliminates time-consuming and expensive manual efforts, automates trust between partners, and brings traceability to supply chains [77]. Automating trust is supposed to use a permissioned blockchain network in which every node needs to be authenticated as a member of the group. In the network, there are some restrictions on the node's ability to participate in the transaction and validation processes. Permissioned blockchain such as Hyperledger Fabic [79] enable trusted parties to send transactions in a peer-to-peer fashion.

Permissioned blockchain such as Hyperledger Fabric allow parties to quickly settle transactions, resulting in faster movement of goods and services without the high-energy-consuming consensus. In addition, it can exercise extensive access control mechanisms to limit the extent of the block-accesses and transaction issuing. The administrators can participate in the blockchain, which would not be allowed in a permissionless blockchain. Due to the benefits of higher performance and being more manageable, permissioned blockchain is more suitable for enterprise applications that require authenticated participants and quick service processes. Enterprise-level applications should have the ability to handle a huge volume of transactions at low latency [80] in general. Hence, there is a concern about the performance of permissioned blockchain, and in order to understand what permissioned blockchain can be used for and how to improve its performance, an adequate quantitative performance model is exigently demanded.

A quantitative model and analysis of the performance of the permissioned blockchain are addressed and resolved in this research with respect to an extensive and practical set of design and performance-related variables. To the best of the authors' knowledge, there have been few adequate yet practical performance models, particularly targeting the specific behavior of permissioned blockchains. In [80], a comprehensive empirical study has been conducted to

characterize the performance of Hyperledger Fabric and identify potential bottlenecks to get a better understanding of the system. The paper also introduced some optimization techniques such as aggressive caching, paralyzing verification, and bulk read/write of the ledgers. It identifies the impact of various configuration parameters and provides various guidelines for configuring and optimizing Hyperledger Fabric v1.0. However, compared to analytical modeling, empirical performance evaluation is expensive, tedious, and time-consuming. By using stochastic reward nets (SRN), Sukhwani et al. [81] proposed a performance model for Hyperledger Fabric in which the potential for performance bottleneck spots in networks with a large number of nodes has been studied. The analysis results showed that the endorsement process is significantly affected by the number of peers and policies. The results also showed that the transaction validation check (VSCC) is the most time-consuming process. The paper suggested using parallel computation to improve VSCC performance. In the paper, many real experiment values have been collected and validated by analysis of variance (ANOVA). However, it didn't provide details about how the mean queue length, utilization, and throughput are calculated. In [82], a comparison has been made between proof-of-work-based blockchains such as Bitcoin and those based on byzantine fault tolerance such as Hyperledger Fabric in terms of scalability and performance, in which it was reported that the performance of the byzantine fault tolerance-based blockchains (permissioned blockchains) is better compared to that of the proof-of-work-based blockchains (permissionless blockchains). On the other hand, permissionless blockchains provide better scalability than permissioned blockchains. The authors provided suggestions on how to improve the blockchain's scalability but didn't provide the empirical evidence that improvements can be achieved. In [83], an application as a standalone benchmark has been developed to simulate database accesses in Hyperledger Fabric, and it was identified that a major performance bottleneck is the data compression of the databases that store the latest key-value pairs in the ledger data, indexes to transactions, and the updated history. The paper provided three opportunities to achieve better performance in Hyperledger Fabric and gave benchmarks to show

the improved performance. However, it provided the only comprehensive empirical evaluation. In [84], the authors proposed the first benchmarking framework, called Blockbench, for evaluating permissioned blockchains. By using Blockbench, they conducted a comprehensive analysis for the performance comparisons of three major permissioned blockchains, namely Ethereum, Parity [89], and Hyperledger Fabric. The paper also demonstrated several bottlenecks and design trade-offs at different layers of the software stack. Blockbench is a benchmarking framework that measures the workloads of the data processing to evaluate the blockchain's performance. However, it couldn't provide any performance prediction or optimization. In [90], Kocsis et al. proposed a performance evaluation model for blockchain technology, which was used to evaluate Hyperledger Fabric v0.6. The main purpose of the model is to evaluate the software design in its early stages, where changing the requirements of the software will not have a big impact on the overall cost and time. The model seems like a good tool for software design and development, but it may not be able to provide a reference for system configuration and optimization. Pu et al. [91] proposed a Generalized Stochastic Petri Nets (GSPN) model for a blockchain system based on Hyperledger Fabric v1.2. It analyzes the impact of different configurations of ordering services on system performance to find the bottleneck. The model shows the transaction flow in detail and provides a simulation-based approach to obtaining the system latency and throughput. The paper proposed a mathematical configuration selection approach to determine the best configuration for ordering services. However, the model couldn't provide the configuration for committing services for optimizing VSCC and the ledger updating process. In [92], Wang et al. presented a performance study and bottleneck analysis of Hyperledger Fabric. The paper studied the performance characterization of each phase of the transaction life cycle and the different ordering services, including Solo, Kafka, and Raft. The experimental result showed that the validate phase was likely to be the system bottleneck. The result also demonstrated that there were no significant performance differences among the three ordering services. The paper showed evidence that the bottleneck of Hyperledger Fabric is in the validate phase rather than the ordering phase, but it

only provided empirical evaluation and a few necessary analyses and couldn’t give suggestions for optimizing the validating performance.

In this context, a quantitative model to trace the operations and their performance of transactions and blocks in a permissioned blockchain has been proposed. This model can be used to simulate the realistic behaviors of the transaction execution, block generation, and verification processes in a permissioned blockchain. The parametric simulations show that the model is effective when it is validated against real-world scenarios. Moreover, the model can be used to perform extensive parametric studies to identify the influence factors of the system of concern and can ultimately help to improve permissioned blockchains' performance, availability, and security.

This chapter is organized as follows: preliminaries will be presented in the following section with reviews; the details of the proposed models with analysis will be presented; then, extensive numerical simulations and their results will be presented in order to validate the efficacy of the proposed models; and lastly, the research will end with conclusions and the future work.

3.2 Preliminaries and Review

Hyperledger Fabric is a platform for permissioned blockchain that has various unique features that are suitable for high-volume enterprise-level business applications. It deploys transactions by executing the smart contracts (e.g., Chaincode) on the blockchain [93]. A Hyperledger Fabric is operated by coordinating different types of nodes, such as endorsing nodes, ordering nodes, and committing nodes, along with clients. Each of these nodes holds an identity in the network space managed by a Membership Service Provider (MSP). Except for the ordering nodes, the other two types of nodes maintain a state database that tracks the current value of all of the assets listed on the ledger. Two types of state database are supported, such as CouchDB externally and GoLeveldb internally [93]. In the Hyperledger Fabric architecture, a channel is a private subnet for communication between two or more specific network members for the purpose of conducting

private and confidential transactions. Each transaction on the network is supposed to be executed on a channel, where each party must be authenticated and authorized to be executed on that channel. Each peer that joins a channel has its own identity, provided by a MSP [93].

Hyperledger Fabric employs, namely, an execute-order-validate architecture [94]. Figure 12 shows the logical view of the architecture. In the architecture, a client initiates a transaction proposal by signing it with his or her credential and then invoking a specific Chaincode, i.e., the Endorsement System Chaincode (ESCC), to endorse the transaction in the endorsing nodes.

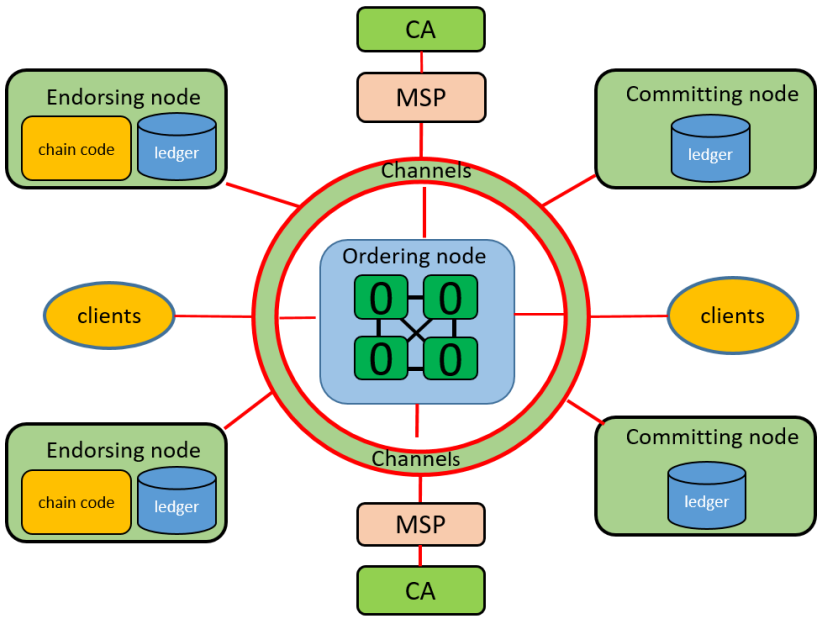

Figure 12. High level diagram of Hyperledger Fabric architecture

The endorsing node initially simulates the transactions and verifies if the client is authorized to invoke transactions on the channel by executing ESCC, and then creates a read/write set for the ledger. At the end of ESCC, the endorsing node signs off the transaction and returns it to the client. Once the client receives enough signatures to satisfy the endorsement policy, the client can submit the transaction, namely, a well-formed transaction [12], to the ordering node for consensus. Hyperledger Fabric logically separates the consensus activities from ordering services

and validating services to make sure that the Hyperledger Fabric architecture can work with any Hyperledger consensus module. The ordering service is responsible for establishing and tracking the total order of all well-formed transactions in the blockchain and ensuring a consistent view of all the transactions across all nodes. Typically, the ordering service consists of multiple nodes for scalability and resilience purposes and coordinates a protocol to reach consensus based on the total order in place. The ordering node is supposed to receive only well-formed transaction responses from the client and process the transactions by checking the current set of policies to make sure whether the participant has administrative rights, and accordingly cut block transactions by the block size limit or the block time limit, and then send the block to all the committing nodes on the channel for validating the transactions. The committing node does not carry any Chaincode, receives the block that came from the ordering nodes, and verifies the ordering node's signature on the block. Each valid block is decoded, and all the transactions go through the Validation System ChainCode (VSCC) evaluation procedure to validate the transactions against the endorsement policy specified for the Chaincode. If the endorsement policy is not satisfied, then that transaction is marked invalid. After VSCC validation, the committing node invokes Multi-Version Concurrency Control (MVCC) validation to ensure that the version of keys read by each transaction during the endorsement phase is identical to the current state in a local ledger at the time of the commit. MVCC is a mechanism to avoid the double-spending problem in Hyperledger Fabric. If the read-set version does not match the recorded version, the transaction is marked as invalid. After VSCC and MVCC validations, a ledger update is invoked in order to append the new block to the local ledger by updating the database either externally or internally.

Hyperledger Fabric employs several different consensus mechanisms as well as implementation approaches to ensure modularity. Most Hyperledger consensus modules use a voting-based approach for consensus that provides fault tolerance within seconds. Consensus in Hyperledger

Fabric can be broken down into three phases: endorsement, ordering, and validation. Figure 13 depicts the process of reaching consensus, which involves three primary phases as follows:

- Endorsement Phase—Simulating the transaction on selected peers and collecting the state changes.
- Ordering Phase—Ordering the transactions and cutting them into a block through a consensus protocol; and
- Validation & Committing Phase—Validating all the transactions in the block and then committing it to the ledger [93].

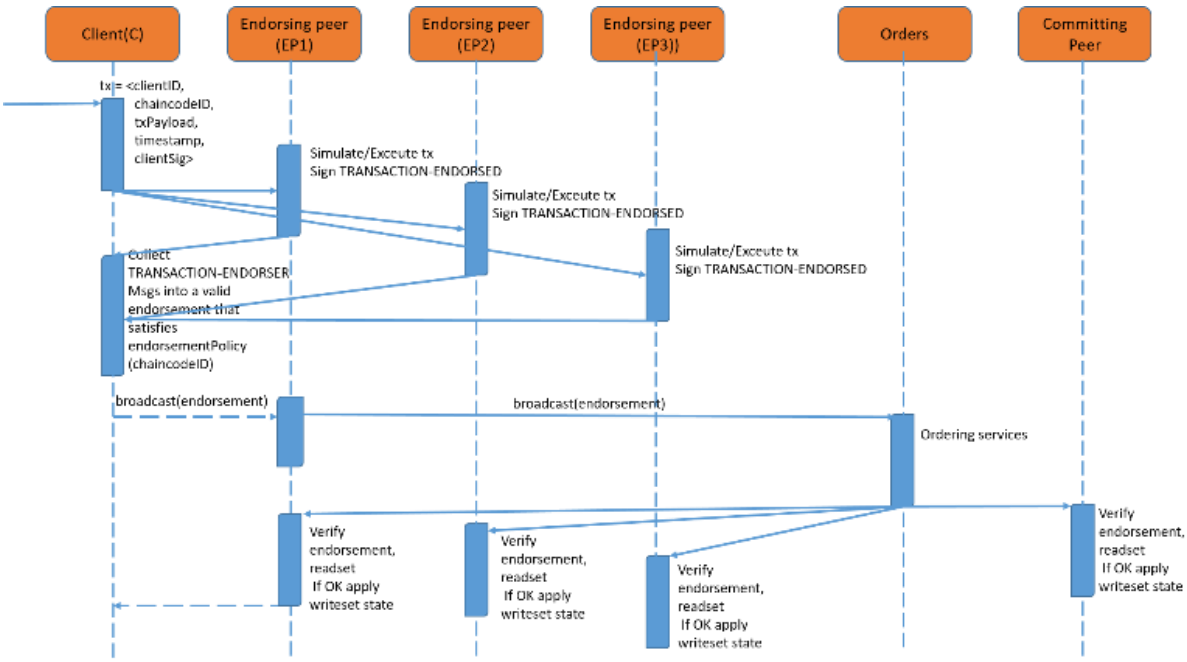

Figure 13. Transaction flow

A queuing network-based model is proposed. as a preliminary study, and three different types of queues are used to evaluate the performance of Hyperledger Fabric: process (or transaction flow), the number of newly generated blocks, the number of waiting transactions, and the overall waiting time. The proposed model tracks the block generation process from the point of a transaction proposal's arrival through the point when the transaction is committed. Figure 14

shows the consensus process that is considered in this work as the basis for the primary assumptions to be made for the proposed model.

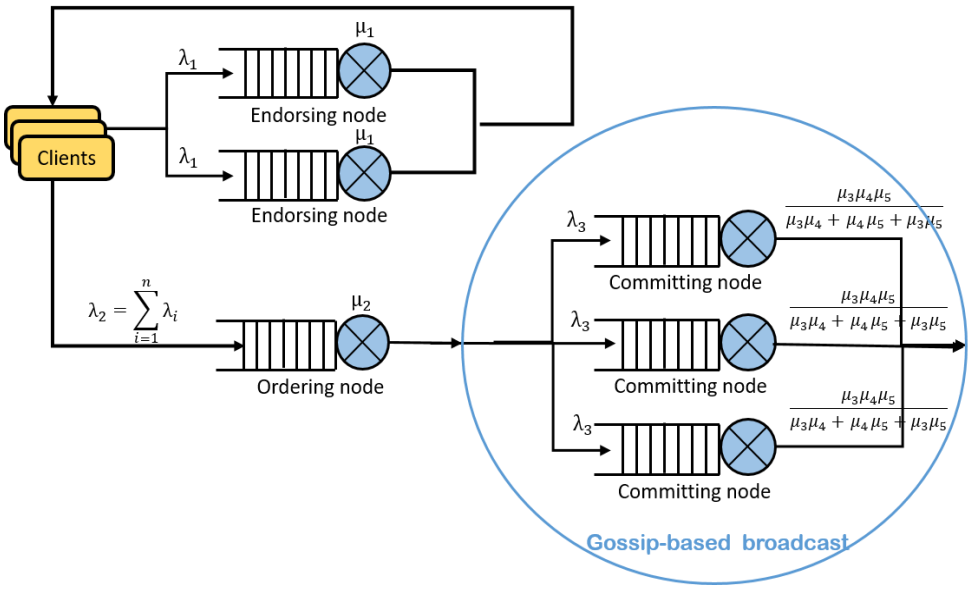

Figure 14. Queuing model of Hyperledger Fabric consensus process

To begin, it is assumed that transaction proposals enter the system via a Poisson process at a rate of λ_1 and are queued in an endorsing node. The endorsement service simulates and verifies the transaction by executing ESCC, which is assumed to be in an exponential distribution at a rate of μ_1. After inspecting and collecting enough proposals to fill its capacity, responding to them, and verifying that the endorsements are correct, the client broadcasts the transaction over the network at a rate of λ_i. Following that, it is assumed that valid and legitimate transactions [80] arrive at the ordering node via a sum of several Poisson processes at a rate of $\lambda_2 = \sum_{i=0}^{n} \lambda_i$ and are stored in the ordering node's queue. The order service is assumed to be in a geometric distribution at a rate of μ_2 in an ordering node. The service time starts with the transactions waiting in the queue till they hit the block timeout or the block size limit, the transactions being ordered, and a block of the pending transactions being created with a digital signature. After the block is created, the ordering nodes immediately deliver the block to the committing nodes using a gossip data

dissemination protocol [93]. The blocks arrive at the committing nodes at a rate λ_3 with a special message transmission method that a message broadcast operates by nodes receiving messages from other nodes on the channel and then forwarding these messages to a number of randomly selected nodes on the channel [93]. After a new block arrives at the committing node, the block will either stay in the queue if the queue is not empty or be served immediately by VSCC validation, MVCC validation, and ledger updating. The service time of two validations and the ledger updating are assumed to be in exponential distributions at three different rates of μ_3, μ_4, μ_5 respectively. After all the transactions in the block are validated by VSCC and MVCC, a new block will be added to the local ledger by the ledger updating, and then the committing nodes notify the clients. Without losing generality, it is also assumed that the discipline of all queues in the proposed models is first-come, first-served (FCFS).

3.3 Model Analysis and Performance Measure

3.3.1 Queueing Model of Transaction Simulation

As a client sends a transaction proposal to invoke ESCC, which is installed on the endorsing node by an administrator, it initializes and manages ledger state and executes a smart contract via a transaction. For every ESCC, there is an endorsement policy to specify the endorsing rules. During the process of executing ESCC, the endorsing node recodes all the state changes into a writeset and collects all keys accessed by the transaction and their versions into a readset. At the end of ESCC, the endorsing node puts the result of the readset and the writeset together and returns it to the client. After collecting a sufficient number of endorsements, the client integrates them into a well-formed transaction. In order to build a queueing model to simulate ESCC executing and the endorsement, It is assumed that all transactions arriving at the endorsing nodes follow a Poisson process at a rate of λ_1. to build a queueing model to simulate ESCC execution and endorsement. When a transaction proposal arrives at the nodes, an ESCC is executed by the

transaction at the rate of μ_1. Because the time required to execute ESCC is solely determined by the current states of the nodes, the ESCC execution time follows an exponential distribution. Since the transaction arrival interval and the time of ESCC executing are of exponential distribution, the stochastic process [95] for the number of transactions $\{N_n(t)\}$ is a homogeneous Markov chain. If the traffic intensity $\rho = \frac{\lambda_1}{\mu_1} < 1$. By using the M/M/1 queue [96] to represent the number of transactions in an endorsing node, the probability of the number of transactions (j) in endorsing node can be obtained [97]:

$$\pi_j = \pi_0 \rho^j, \qquad where\ \rho = \frac{\lambda_1}{\mu_1} < 1, \qquad \pi_0 = 1 - \rho \tag{1}$$

, and the numbers of the transactions in the system:

$$L = \sum_{j=0}^{\infty} j\pi_j = (1 - \rho)\rho \sum_{j=1}^{\infty} j\rho^{j-1} = (1 - \rho)\rho \frac{d}{d\rho}(\sum_{j=0}^{\infty} \rho^j) = \frac{\rho}{(1-\rho)} = \frac{\lambda_1}{\mu_1 - \lambda_1} \tag{2}$$

Using Little's Law, get the waiting time in endorsing node:

$$W = \frac{L}{\lambda_1} = \frac{1}{\mu_1 - \lambda_1} \tag{3}$$

, and the waiting time in the queue:

$$W_q = W - \frac{1}{\mu_1} = \frac{\lambda_1}{(\mu_1 - \lambda_1)\mu_1} \tag{4}$$

, as well as the number of the transactions in the queue:

$$L_q = \frac{\lambda_1^2}{(\mu_1 - \lambda_1)\mu_1} \tag{5}$$

3.3.2 Queueing Model of Block Creation and Delivery

After the client collects enough responses from the endorsing nodes and then verifies that the endorsements are the same, it broadcasts a well-formed transaction to the ordering nodes [93]. Therefore, it is reasonable to assume that all the arrivals of well-formed transactions are independent and have a constant probability. Since the ordering node will receive multiple well-formed transactions from multiple clients at the same time, the process of the transactions arriving at the ordering node is a sum of Poisson processes. In this section, a discrete probability is used to represent transaction arrival rather than continuous probability, so the transaction arrival can be a binomial distribution [98]. Let's $A_{j,k}$ be a probability for *j* transactions arrival during a time period t_k, then t_k involves several of the intervals $\Delta t = t_k - t_{k-1}$ with p_a probability. Hence, get the probability of *j* transaction arrivals:

$$A_{j,k} = \begin{cases} \binom{k}{j} p_a^j (1-p_a)^{k-j} & 0 \le j \le k \\ 0 & otherwise \end{cases} \quad k = 0,1,2,\ldots \tag{6}$$

Since the mean of the binomial process is $kp_a = \lambda_2 k\Delta t$ where $\lambda_2 = \sum_{i=0}^{n} \lambda_i$, get $p_a = \lambda_2 \Delta t$.

In order to create a semi-Markova chain, it is necessary to define a new stochastic process$\{ON_n^+\}$ that is defined as the number of transactions in the ordering node immediately after the nth block departure. The stochastic process is given by:

$$ON_{n+1}^+ = (ON_n^+ - 1)^+ + tx\_a_{n+1} \tag{7}$$

, where tx_a_{n+1} is the number of transactions arriving during the $(n+1)$st block departure period.

In an ordering node, the order service collects the transactions from the queue in the node if the queue is not empty and establishes and delivers a new block to the committing node. At the same

time, the node tracks the total order of all the transactions to ensure a consistent view across all the nodes until the number of transactions hits the block size or the duration reaches the timeout. The number of transactions in a block depends on the channel configuration parameters that are related to the desired size and maximum elapsed duration for a block [12]. In other words, the time it takes to generate a new block depends on the number of well-formed transactions in the queue or the block time (the maximum elapsed duration). Therefore, the block creating, and delivery interval has a memoryless character, and it can be sure that the distribution of the block creating, and delivery interval is a geometrical distribution [99] for a discrete probability (or an exponential distribution for a continuous probability). Let's S_k be a probability that a block creating and delivery time interval *V* contains *k* time marks ($V = \sum_{k=0}^{\infty} kS_k\Delta t = b\mu_2\Delta t$, where b is the block size and μ_2 is the rate of block creating and delivery) and a block is created and delivered at the beginning of $(k+1)$st Δt, then get:

$$S_k = P(V = k+1) = (1-p_s)^k p_s \quad k = 0,1,2,\ldots \tag{8}$$

, where $p_s$the probability that a block is created and delivered at the moment. Since the mean of the geometric process is $\frac{1-p_s}{p_s} = b\mu_2$, the probability of block generation $p_s = \frac{1}{1+b\mu_2}$. The Markov Chain of ON_n^+ process is as following:

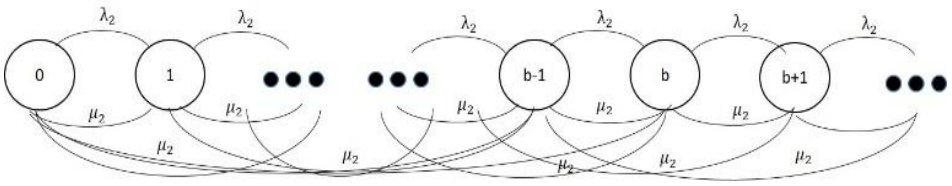

Figure 15. The rate diagram of the transaction ordering queue

In Figure 15, λ_2 is a transactions arrival rate, and $\mu_2$is a rate that an ordering node creates and deliveries the blocks. Apparently, the interval of the block creating, and delivery is independent of the transactions arriving process. Therefore, the distribution of tx_a_{n+1} is a join probability

function that counts the arrival transaction number during a block creating and delivery interval, and then the probability of the number of arrival transactions (j) in the interval is given by:

$$k_j = \sum_0^\infty A_{j,k} S_k = \begin{cases} \sum_0^\infty \binom{k}{j} p_a^j (1-p_a)^{k-j} (1-p_s)^k p_s & j \geq 0 \\ 0 & j < 0 \; or \; j > k \end{cases} \tag{9}$$

, multiplying by z^j and summing over j, obtain the generating function $K(z)$:

$$K(z) = \sum_0^\infty (1-p_a+p_a z)^k (1-p_s)^k p_s = \sum_0^\infty ((1-p_a+p_a z)(1-p_s))^k p_s$$

$$= \frac{p_s}{1-(1-p_a+p_a z)(1-p_s)} \tag{10}$$

Let's define the one-step transition probability:

$$p_{i,j} = \begin{cases} k_j & 0 \leq i \leq b \\ k_{j-i+b} & b \leq i \leq j+b \\ 0 & i > j+b \end{cases} \qquad b \text{ is block size} \tag{11}$$

when $0 \leq i \leq b$, the transition from i to j can be explained as i transactions are going to become a block; when $b \leq i \leq j+b$, the transition means that b transactions are going to become a block, but $i-b$ transactions still stay in the queue until reaching the block size or timeout. In order to let the system be in the steady state, it is necessary to assume the network traffic intensity $\rho = \frac{\lambda_2}{b\mu_2} < 1$. From equation (11), get a probability of number of transactions (j) in the queue moment immediately after creating and delivery nth block as following:

$$\pi_j = \sum_{i=0}^\infty \pi_i \, p_{i,j} = k_j \sum_{i=0}^{b-1} \pi_i + k_{j-i+b} \sum_{i=b}^\infty \pi_i \tag{12}$$

multiplying by z^j and summing over j, obtain the generating function $P(z)$:

$$P(z) = \sum_{j=0}^{\infty} \pi_j z^j = \frac{\sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right)}{\frac{z^b}{K(z)} - 1} \tag{13}$$

substitute equation (10) into equation (13), get:

$$P(z) = \frac{\sum_{i=0}^{b-1} \pi_i \left(z^b - z^i\right)}{\frac{z^b \left(1 - (1 - p_s)(1 - p_a + p_a z)\right)}{p_s} - 1} \tag{14}$$

According to M.L. Chaudhry and J.G.C. Templeton [49], equation (14) can be re-written as following:

$$P(z) = \frac{C(z-1)\prod_{j=1}^{b-1}\left(z - z_j\right)}{(z-1)\prod_{j=1}^{b}\left(z - z_j\right)} = \frac{C}{z - z_b} \tag{15}$$

, where *C* is a constant to be determined and z_b is a simple root of denominator of equation (14) which is larger than 1. Since $P(1-) = 1,\ C = 1 - z_b$ then get:

$$P(z) = \frac{z_b - 1}{z_b - z} \tag{16}$$

By differentiating equation (16) and substituting $z = 1$, get the mean of number of transactions in the ordering node immediately after the *n*th block departure:

$$E(ON_n^+) = L^+ = \frac{z_b - 1}{(z_b - z)^2} = \frac{1}{z_b - 1} \tag{17}$$

Choosing zeros numerator and denominator of equation (14), and differentiating on both sides of denominator, get:

$$z_b - 1 = \frac{p_s}{(1 - p_s)p_a}$$

then, have

$$L^{+} = \frac{(1 - p_s)p_a}{p_s} \tag{18}$$

According to Ke, Park [40], get the waiting time in queue:

$$W_q = \frac{L^{+} - \lambda_2/\mu_2 + \lambda_2 W_r}{\lambda_2} \tag{19}$$

where W_r is the expected residual lifetime that a well-formed transaction to be packaged into a block. Since the memoryless character of the block creating and delivery time interval, $W_r = \frac{1}{\mu_2}$ then get the waiting time in queue:

$$W_q = \frac{L^{+}}{\lambda_2} = \frac{1}{\lambda_2(z_b - 1)} = \frac{(1 - p_s)p_a}{\lambda_2 p_s} \tag{20}$$

, and the queue length by using Little's law as well as system waiting time:

$$L_q = \lambda_2 W_q = \frac{1}{z_b - 1} = \frac{(1 - p_s)p_a}{p_s} \tag{21}$$

$$W = W_q + \frac{1}{\mu_2} = \frac{1}{\lambda_2(z_b - 1)} + \frac{1}{\mu_2} = \frac{(1 - p_s)p_a}{\lambda_2 p_s} + \frac{1}{\mu_2} \tag{22}$$

3.3.3 Queueing Model of Block Validation and Committing

After the blocks are created and signed, the ordering node broadcasts them to the committing nodes using a gossip data dissemination protocol, besides the dedicated nodes that directly connect to the order nodes. The node receives gossip-based broadcasting messages from ordering nodes or other nodes on the channel, and then forwards these messages to a number of randomly selected committing nodes on the channel [93]. In this randomized gossip broadcasting algorithm, the Poisson model could be a reasonable first-order model for the arrival of messages at a particular node [100]. As a result, there is compelling evidence to believe that block arrival at the committing node is a Poisson process with a λ_3. rate. Without losing any generality, it is also

assumed that the size of the re-ordering buffer in the committing node is big enough to deal with all the out-of-order delivery due to the gossip data dissemination protocol.

As the block arrives at the committing node, the transactions within the block are validated in two different ways (VSCC and MVCC) at a rate of μ_3 and μ_4 respectively, to ensure the endorsement policy is fulfilled and that there have been no changes to the ledger state for read-set variables. At the last step of the block committing, the ledger updating appends the new block to the local ledger by updating the database (GoLeveldb or CouchDB) at a rate of μ_5. Since the services of the validations (VSCC, MVCC, and ledger updating) are independent of each other, every service has a memoryless character. Given the different service rates, such as μ_3 is the number of blocks for VSCC validation, μ_4 is the number of blocks for MVCC validation and μ_5 is the number of blocks per second for ledger updating, this chain of services is a typical phase-type [101] service process. In order to get a fine-grained performance analysis, it is better to use a phase-type distribution to define the distribution of the validations and ledger updating services.

The stochastic process of the number of blocks in the system $\{CN_n(t)\}$ is not a homogeneous Markov chain because the services of the validation and ledger updating are chained together into a phase type distribution. To reveal the Markov chain that has been embedded in the process $\{CN_n(t)\}$ is a new stochastic process $\{CN_n^+\}$ must be defined. A process $\{CN_n^+\}$ is defined as the number of transactions in the committing node immediately following the commit of the nth block, and is given by:

$$CN_{n+1}^+ = (CN_n^+ - 1)^+ + b\_a_{n+1} \tag{23}$$

, where b_a_{n+1} is the number of blocks arriving during the $(n+1)$st service period. A Markov chain of stochastic process $\{CN_n^+\}$ is shown as following figure:

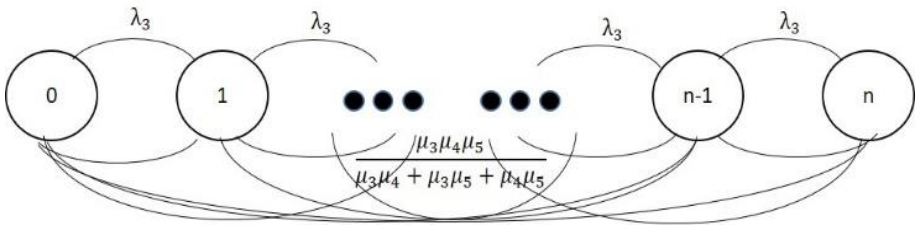

Figure 16. The rate diagram of the block committing queue.

Since the block arrival process is Poisson process with rate λ_3, the probability density function of number arrival blocks (j) at time t is given by:

$$a(j;t) = \frac{e^{-\lambda_3 t}(\lambda_3 t)^j}{j!} \tag{24}$$

The service process is a phase type distribution with three phases at rates of μ_3, μ_4 and μ_5 respectively, given the probability of completely executing all three services at time t:

$$s(t) = \mu_3\mu_4\mu_5\left(\frac{e^{-\mu_5 t}}{(\mu_3-\mu_5)(\mu_4-\mu_5)} + \frac{e^{-\mu_3 t}}{(\mu_4-\mu_3)(\mu_5-\mu_3)} + \frac{e^{-\mu_4 t}}{(\mu_3-\mu_4)(\mu_5-\mu_4)}\right),$$
$$where\ \mu_3 \neq \mu_4 \neq \mu_5 \tag{25}$$

Since the arrival process and the service process are independent of each other, the distribution of b_a_{n+1} is a join probability function that counts the number of arrival blocks during a phase-type service interval. Therefore, the probability of the number of arrival blocks (j) in the service interval is given by:

$$P_j = \int_0^\infty a(j;t)s(t)dt$$
$$= \int_0^\infty \frac{e^{-\lambda_3 t}(\lambda_3 t)^j}{j!}\mu_3\mu_4\mu_5\left(\frac{e^{-\mu_5 t}}{(\mu_3-\mu_5)(\mu_4-\mu_5)} + \frac{e^{-\mu_3 t}}{(\mu_4-\mu_3)(\mu_5-\mu_3)}\right.$$
$$\left. + \frac{e^{-\mu_4 t}}{(\mu_3-\mu_4)(\mu_5-\mu_4)}\right)dt \tag{26}$$

Let's define the one-step transition probability:

$$P_{ij} = P\{CN_n^+ = j | CN_{n-1}^+ = i\} \qquad i,j \geq 0 \; and \; n \geq 1 \tag{27}$$

is given by

$$P_{ij} = \begin{cases} P_j, & i = 0, j \geq 0 \\ P_{j-i+1}, & i > 0, j \geq i-1 \\ 0, & j < i-1, i > 0 \end{cases} \tag{28}$$

In order to let the system stay in the steady-state, it is necessary to assume the network traffic intensity $\rho = \frac{\lambda_3(\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5)}{\mu_3\mu_4\mu_5} < 1$. From equations (28) and (26), an embedded Markov chain probability of number of blocks (j) in a committing node at the moment immediately after nth block committed is obtained:

$$\pi_j = \sum_{i=0}^{\infty} \pi_i \, P_{i,j} \; = \; \sum_{i=1}^{j} \pi_i P_{j-i+1} + \pi_0 P_j \tag{29}$$

$$\begin{aligned} \pi_j = \lambda_3 \Bigg(& \sum_{i=1}^{j} \pi_i \int_0^{\infty} \frac{e^{-\lambda_3 t}(\lambda_3 t)^{j-i+1}}{(j-i+1)!} \left[\frac{\mu_3\mu_4 e^{-\mu_5 t}}{(\mu_3-\mu_5)(\mu_4-\mu_5)} + \frac{\mu_4\mu_5 e^{-\mu_3 t}}{(\mu_4-\mu_3)(\mu_5-\mu_3)} \right. \\ & \left. + \frac{\mu_3\mu_5 e^{-\mu_4 t}}{(\mu_3-\mu_4)(\mu_5-\mu_4)} \right] dt \\ & + \left(1 - \frac{\lambda_3(\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5)}{\mu_3\mu_4\mu_5} \right) \int_0^{\infty} \frac{e^{-\lambda_3 t}(\lambda_3 t)^{j}}{j!} \left[\frac{\mu_3\mu_4 e^{-\mu_5 t}}{(\mu_3-\mu_5)(\mu_4-\mu_5)} \right. \\ & \left. + \frac{\mu_4\mu_5 e^{-\mu_3 t}}{(\mu_4-\mu_3)(\mu_5-\mu_3)} + \frac{\mu_3\mu_5 e^{-\mu_4 t}}{(\mu_3-\mu_4)(\mu_5-\mu_4)} \right] dt \Bigg), \\ & where \; \mu_3 \neq \mu_4 \neq \mu_5 \end{aligned} \tag{30}$$

For more rigorous proof, see Chaudhry and Templeton [49]. Since the blocks arrival is a Poisson distribution and the service is a phase-type distribution, the model can be seen as an M/G/1 queueing model [102]. Therefore, the Pollaczek–Khinchine formula [103] can be used to state a

relationship between the queue length and the service time and then get the mean number of blocks in the queue:

$$L_q = \frac{\rho^2 + \lambda_3^2\left(\frac{1}{\mu_3^2} + \frac{1}{\mu_4^2} + \frac{1}{\mu_5^2}\right)}{2(1-\rho)}, \qquad where\ \ \rho = \frac{\lambda_3(\mu_3\mu_4 + \ \mu_4\mu_5 + \mu_3\mu_5)}{\mu_3\mu_4\mu_5} < 1 \quad (31)$$

Using Little's law, get the waiting time in queue.

$$W_q = \frac{L_q}{\lambda_3} = \frac{\rho^2 + \lambda_3^2\left(\frac{1}{\mu_3^2} + \frac{1}{\mu_4^2} + \frac{1}{\mu_5^2}\right)}{2(1-\rho)\lambda_3} \quad (32)$$

, then get the waiting time in system:

$$W = W_q + \frac{\mu_3\mu_4 + \ \mu_4\mu_5 + \mu_3\mu_5}{\mu_3\mu_4\mu_5} \quad (33)$$

3.3.4 Queueing Model of Transaction Processing

The flow of a transaction in Hyperledger Fabric through the consensus algorithm [23] consists of three phases of transaction processing. During the process of reaching consensus, the data of transaction traverses from node to node along several links on the path before the transactions are committed. Based on the queueing network model for the Hyperledger Fabric consensus process as shown in Figure 13 and the analysis of the queueing models in previous subsections 3.3.1, 3.3.2, and 3.3.3, the transactions are assumed to exponentially arrive at the queues in a Poisson process, and further, they take on new exponential services while traversing a cascade of queues in series. As all the forward transition probabilities in the queues (endorsing, ordering, and committing) are the same as the backward probabilities as shown in 3.3.1, 3.3.2, and 3.3.3, according to Burke's theorem [104], a virtual circuit path of the transaction flow in the channel can be approximately decomposed into a tandem queue with 3 M/M/1 queues as shown in Figure 17.

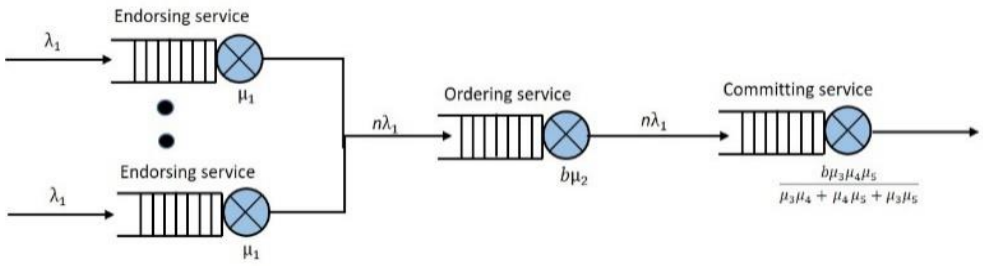

Figure 17. The queueing virtual circuit path for the transaction flow.

In the figure 17, λ_1 is the transaction arrival rate and n is number of channels, μ_1 is ESCC service rate and $\mu_1 > \lambda_1$; b is block size, μ_2 is block creating & delivery rate and $b\mu_2 > n\lambda_1$; μ_3 is block VSCC rate, μ_4 is block MVCC rate, μ_5 is the ledger updating rate and $\frac{b\mu_3\mu_4\mu_5}{\mu_3\mu_4+\mu_4\mu_5+\mu_3\mu_5} > n\lambda_1$.

Using Kleinrock Independence Approximation [105], get average number of the uncommitted transactions in a channel:

$$L_{tx} = \frac{\lambda_1}{\mu_1 - \lambda_1} + \frac{n\lambda_1}{b\mu_2 - n\lambda_1} + \frac{n\lambda_1}{\frac{b\mu_3\mu_4\mu_5}{\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5} - n\lambda_1} \tag{34}$$

Using Little's law, get the waiting time for a transaction completion:

$$W_{tx} = \frac{L_{tx}}{n\lambda_1} = \frac{1}{n(\mu_1 - \lambda_1)} + \frac{1}{b\mu_2 - n\lambda_1} + \frac{1}{\frac{b\mu_3\mu_4\mu_5}{\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5} - n\lambda_1} \tag{35}$$

3.4 Numerical Simulation and Results

The primary objective of the simulation is to reveal various preliminary performances of interest in the Hyperledger Fabric, such as the average queue length and the transaction waiting time. In this section, the effects of increasing the transaction/block arrival rate λ, and transaction/block service rate μ, and channel number are investigated.

In order to make the results of simulations more realistic, it would be feasible to consider using the results that come from the real experiment. In [81], the authors provided a testing environment, and its configuration is summarized below.

- Each node is launched as a Docker container and then connected in a network using Docker Swarm2. The containers corresponding to each organization are run on an independent physical node (‘Org0’, ‘Org1’).
- All containers corresponding to the ordering service run in a single physical node (‘Ordering Service’). 4 Kafka brokers, 3 ZooKeeper nodes.
- A Hyperledger Caliper is deployed on a separate physical node, with multiple client threads.
- Physical machines corresponding to Org0, Org1, and Caliper have 4 CPUs (1 socket, 4 core) (Intel Xeon 2.2 GHz) with 12GB RAM.
- Physical machines corresponding to the ordering service have 16 CPUs (2 sockets, 4 cores, 2 hyper-threads) Intel Xeon 2.4 GHz with 32GB RAM and 7200 rpm Hard Disk Drive.
- All physical machines are connected with a 1 Gbps switch.
- All nodes are synchronized using Network Time Protocol (NTP) service so that transaction latency can be measured across nodes.
- Communication between all nodes is configured to use Transport Layer Security (TLS).

Based on the configuration, [81] provided an application that maintains account balances for users and performs two functions, such as ’open’ and ’transfer’, as well as their test results. Function ‘open’ checks if an account exists, and if not, creates a new account and assigns it an account balance. Thus, this function performs one read and one write operation to the key-value store. Since the function ‘open’ evaluates the basic operations, the statistical data that comes from

the execution of 'open' can fairly represent a typical Hyperledger Fabric workload. Therefore, it is reasonable to choose the results of the performance evaluation for opening an account as the parameter values of our model. Table 1 summarizes the test results for function 'open':

Parameter	**Block Size**	**Mean Time (ms)**
ESCC	-	3.25
Block Creation and Delivery	40 80 120	75.74 81.60 93.56
VSCC	-	2.52
MVCC	40 80 120	2.56 5.10 7.20
Ledger Updating	40 80 120	207.80 208.30 188.40

Table 1. The parameter values for 'open' transaction [81]

3.4.1 Impact of Transaction/Block Arrival Rate

A endorsement performance simulation has been conducted by using equations (4) and (5). Figure 18 plots the tendencies of the average queue length and waiting time in the endorsing node as transaction arrival rates increase from 100 to 300 txs/s, given a fixed ESCC service rate (μ_1) that can be calculated by the mean time of ESCC in Table 1 ($\mu_1 = \frac{1000}{3.25} = 307.7\ txs/s$).

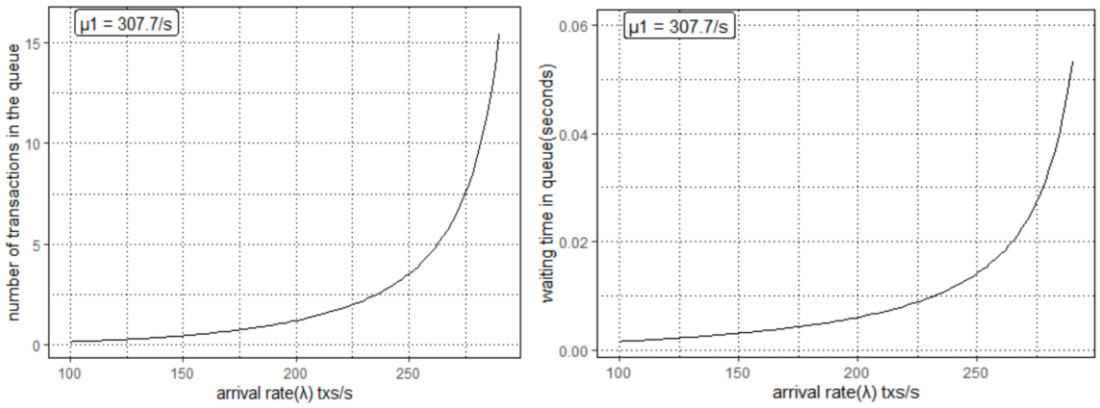

Figure 18. The length and waiting time of endorsing queue

The average queue length and waiting time grow very slowly as the transaction arrival rate increases from 100 to 250 txs/s. When the arrival rate exceeds 250 txs/s, the average queue length and waiting time rapidly increase because the traffic intensity $\rho = \frac{\lambda_1}{\mu_1}$ is closing to 1. One way to tackle this issue is to increase the computational power of endorsing nodes or improve ESCC performance. Figure 19 shows the tendencies of the average queue length and wait time in the endorsing node by increasing the transaction arrival rates from 100 to 300 txs/s, and ESCC service rate from 305 to 310 txs/s. The average queue length and waiting time grow slowly if the ESCC rate can be increased to 310 txs/s after the arrival rate is over 250 txs/s.

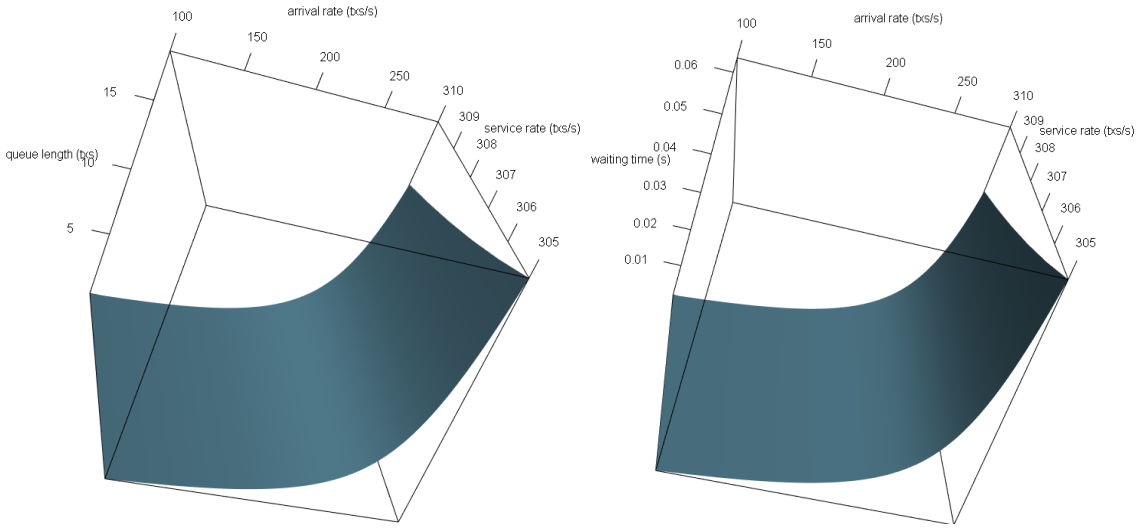

Figure 19. The length and waiting time of endorsing queue with λ_1 and μ_1 both being changed.

Figure 20 shows the results of the numerical simulation using equations (20) and (21). The figure depicts the tendencies of the average queue length and waiting time in the ordering node when the well-formed transaction arrival rates are increased from 300 to 1000 txs/s, given a fixed block creating and delivery service rate of (μ_2) that can be obtained from the mean time of the block creating & delivery (block size = 80) in Table 1 ($\mu_2 = \frac{1000}{81.60} = 12.255\ blocks/s$).

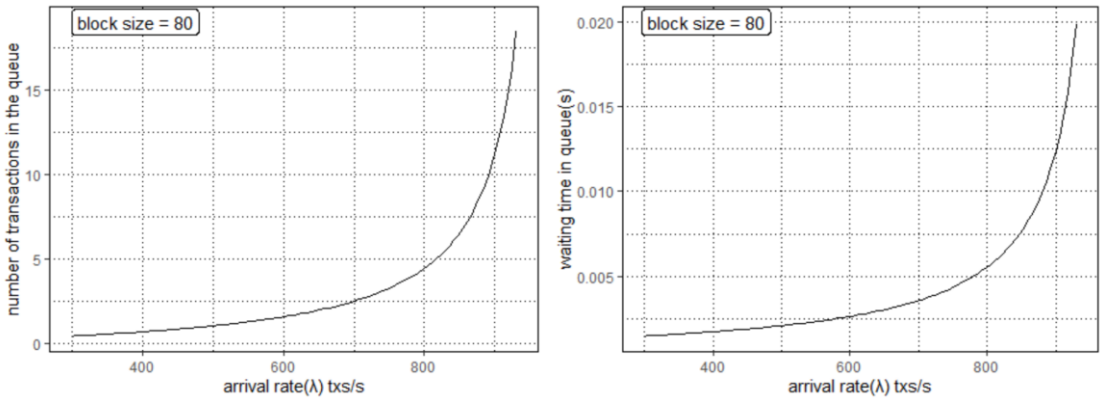

Figure 20. The length and waiting time of ordering queue

During the increase of the well-formed transaction arrival rate $\lambda_2 = \sum_{i=0}^{n} \lambda_i$ from 300/s to 800/s, the average queue length and waiting time grow very slowly. However, once the arrival rate reached 850 txs/s, the average queue length and waiting time increased dramatically because the well-formed transaction arrival rate was approaching the value of the block creation and delivery rate multiplied by the block size. ($b\mu_2 = block\ size\left(\frac{1000}{mean\ time\ (ms)}\right) = \frac{80000}{81.60} = 980.4\ txs/s$). In order to reduce the growth speed of the average queue length and waiting time after the arrival rate of over 850 txs/s, adjusting the block size from 80 to 120 is one of the suitable solutions. Figure 21 shows the tendencies of the average queue length and waiting time by increasing the value of $b\mu_2$ from 900 to 1300 txs/s. As the same time, the arrival rate is growing from 300 to

1000 txs/s. From figure 21, the growth of the average queue length and waiting time slows down dramatically if the block size has been increased.

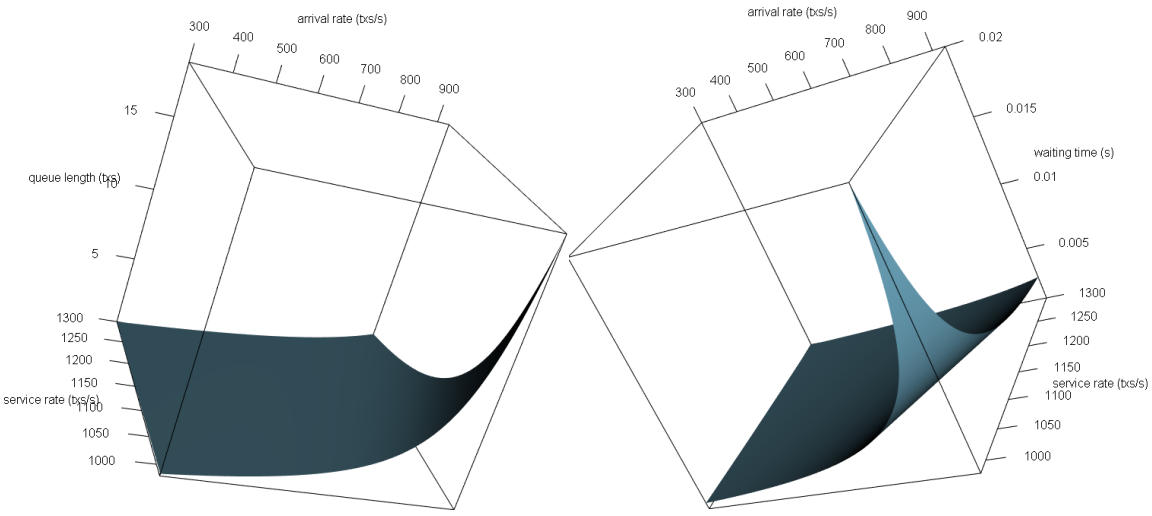

Figure 21. The length and waiting time of ordering queue with λ_2 and μ_2 both being changed.

A performance simulation has been conducted by using the equations (31) and (32). Figure 22 depicts the trends in average queue length and waiting time in committing nodes when the block arrival rates λ_3 are increased from 0 to 2.5 block/s, with fixed values of the VSCC validation rate (μ_3), MVCC validation rate (μ_4) and ledger updating rate (μ_5). VSCC validation rate ($\mu_3 = \left(\frac{1000}{mean\ time\ (ms)}\right)/block\ size = \left(\frac{1000}{2.52}\right)/80 = \frac{396.83}{80} = 4.96$ blocks/second), MVCC validation rate ($\mu_4 = \frac{1000}{mean\ time\ (ms)} = \frac{1000}{5.10} = 196.08$ blocks/second) and the ledger updating rate ($\mu_5 = \frac{1000}{208.30} = 4.801$ blocks/second) can be obtained by calculating the mean values in Table 1.

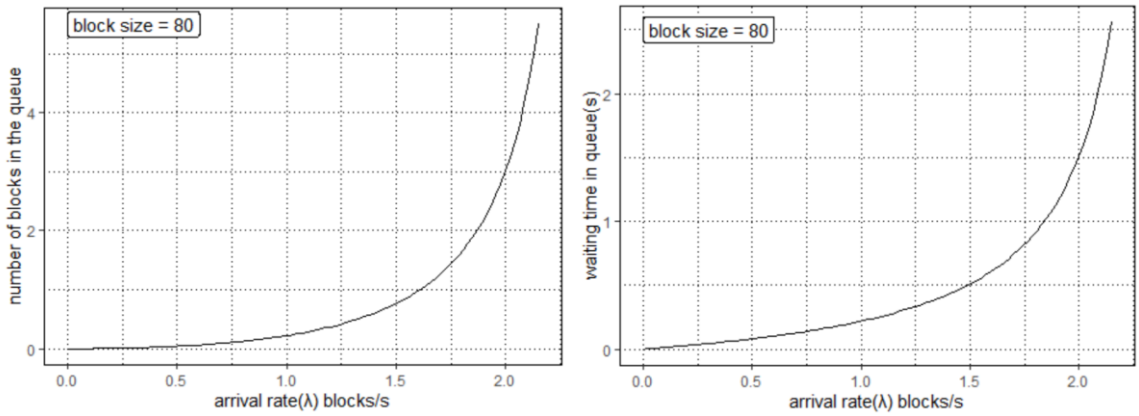

Figure 22. The length and waiting time of committing queue

From the figure, the average queue length and waiting time in committing nodes grow very slowly when the arrival rate is increasing from 0.5 to 1.6 blocks/s, but the average queue length and waiting time grow very fast when the arrival rate reaches and passes 2 blocks/s. Because the block arrival rate is more comparable to the average service rate of the three phases ($\frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4+\mu_4\mu_5+\mu_3\mu_5} = \frac{4.96\times196.08\times4.801}{4.96\times196.08+4.96\times4.801+196.08\times4.801} = 2.41\ blocks/s$), when the rate exceeds the level. One solution to mitigate this problem might be a migration from CPU to GPU and increasing the memory space of the committing nodes. Figure 23 shows that the growth of the queue length and the waiting time are getting slower and slower if the validations and the ledger updating performance can be improved from 2.41 to 3.41 blocks/s.

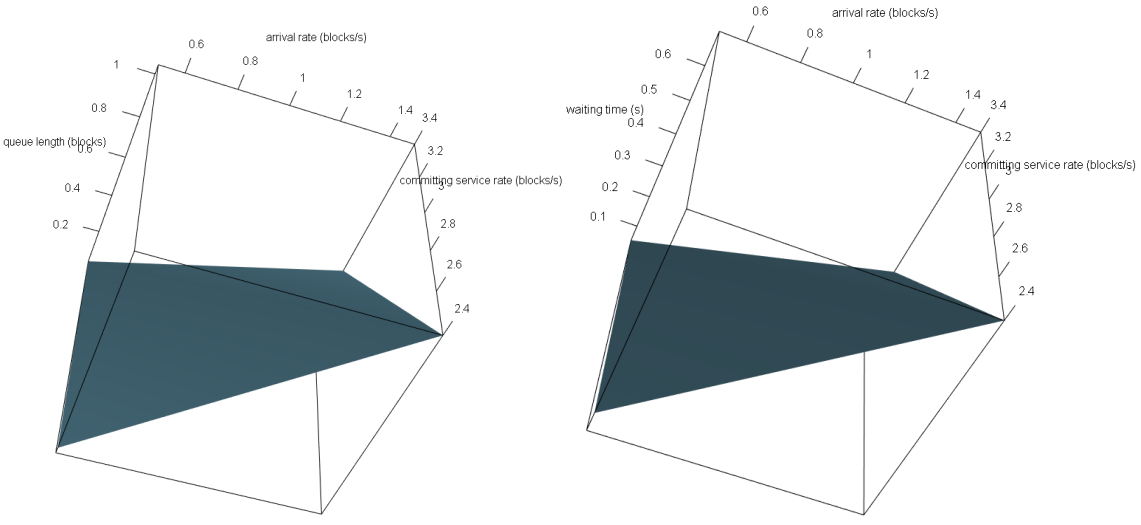

Figure 23. The length and waiting time of committing queue with λ_3 and $\frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4+\mu_4\mu_5+\mu_3\mu_5}$ both being changed.

From the plots of the simulations, it can be easily identified that a major performance bottleneck in Hyperledger Fabric occurred in the committing nodes. To go a step further, VSCC validation and the ledger updating latency discourage the rising transaction arrival rate. The complexity of the endorsement policy and the block size mainly affect the performance of VSCC validation because the VSCC validation of a transaction endorsement requires evaluation of the endorsement policy expression against the collected endorsements and checking for satisfiability, which is NP-Complete [80]. Another fact that significantly degrades the performance relates to the database access in Hyperledger Fabric. For instance, GoLevelDB is 3x faster than CouchDB [80], GoLevelDB data compression can significantly degrade the transaction throughput, and increasing the state database size from 256 to 1024 MB degrades the throughput by as much as 70% [83].

3.4.2 Impact of Block Service Rate

According to the previous study, the performance bottleneck occurred in VSCC validation and the process of ledger updating. This section will simulate the models on the committing nodes by increasing the validating rate and ledger updating rate to see which is the most cost-effective way to improve the performance.

According to Thakkar et al., there are two ways to improve the throughput of VSCC validation: using cache for MSP to avoid deserialization of the serialized identity every time, and parallel executing VSCC validation of a block to maximize the CPU utilization [12]. Figure 24 shows the trends in average queue length and waiting time from executing equations (31) and (32) by increasing the VSCC validating rate (μ_3) from 5 to 10 blocks/s, given a fixed block arrival rate

($\lambda_3 = 2\ block/s$), MVCC validating rate ($\mu_4 = 196.08\ blocks/s$), the ledger updating rate ($\mu_5 = 4.801\ blocks/s$) and block size (80).

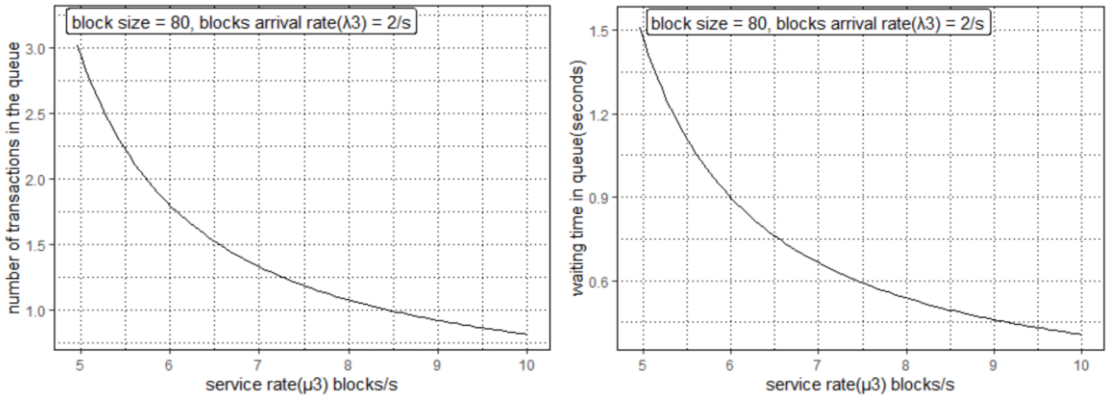

Figure 24. The length and waiting time with increasing the service rate of VSCC validation.

From figure 24, the average queue length and waiting time almost linearly decrease while the VSCC validation rate is growing before it reaches twice the starting rate. This phenomenon shows that increasing the VSCC validation rate may be a more effective way to improve the performance of transaction validation and block committing. A more dynamic simulation has been conducted. In the simulation, the VSCC validation rate will be increased from 5 to 10 blocks/s and the block arrival rate will be increased from 0 to 2 blocks/s. Figure 25 shows that the node can still gain a much better performance when the arrival rate is over 2 blocks/s if the VSCC validation rate can be increased from 4 to 10 blocks/s.

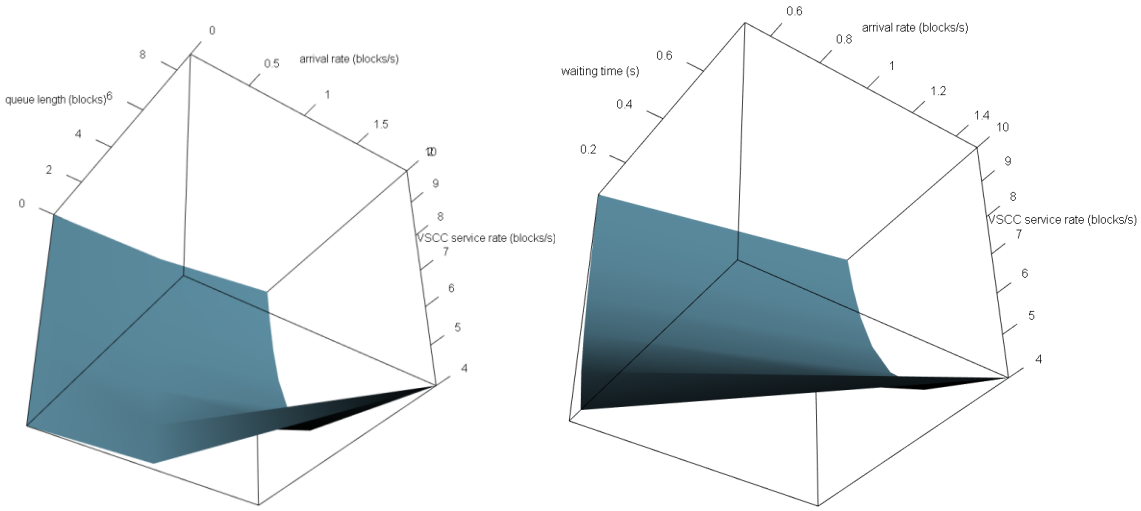

Figure 25. The length and waiting time with increasing the service rate of VSCC validation and the block arrival rate.

Thakkar et al. also proposed using bulk read/write to optimize the database I/O performance during MVCC validation [12]. Figure 26 depicts the tendencies of the average queue length and waiting time in a committing node as the MVCC validation rate (μ_4) is increased from 100 to 2000 blocks/s, given a fixed block arrival rate ($\lambda_3 = 2\ block/s$), VSCC validating rate ($\mu_3 = 4.96\ blocks/s$), the ledger updating rate ($\mu_5 = 4.801\ blocks/s$) and the block size (80).

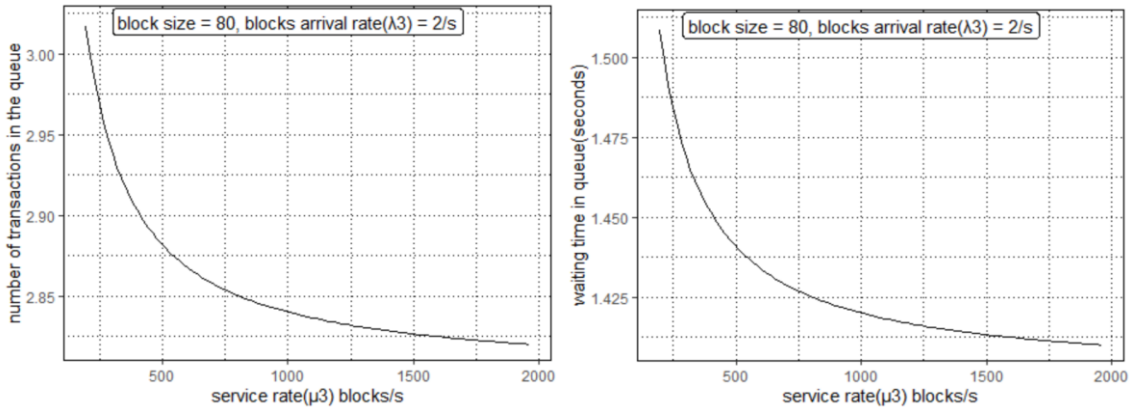

Figure 26. The length and waiting time with increasing the service rate of MVCC validation.

The results tell us that the average queue length and waiting time are reduced during the growth of the MVCC validating rate, but the reducing rate of the average queue length and waiting time is much smaller than the result that was obtained by increasing the VSCC validating rate. In other words, optimizing the performance of MVCC validation may not be the most cost-effective approach to reducing the average queue length and waiting time. Figure 27 depicts the tendencies of the average queue length and waiting time in committing nodes by increasing the MVCC validation rate (from 100 to 2000 blocks/s) and the arrival rate (from 0 to 2 blocks/s), given a fixed block size of 80, VSCC validating rate ($\mu_3 = 4.96\ blocks/s$), the ledger updating rate ($\mu_5 = 4.801\ blocks/s$). Again, the results clearly show that optimizing the performance of MVCC validation may not be a cost-effective approach to reducing the growth rate of the average queue length and waiting time because the growth rate could not gain noticeable progress by increasing the MVCC validation rate from 200 to 500 blocks/s.

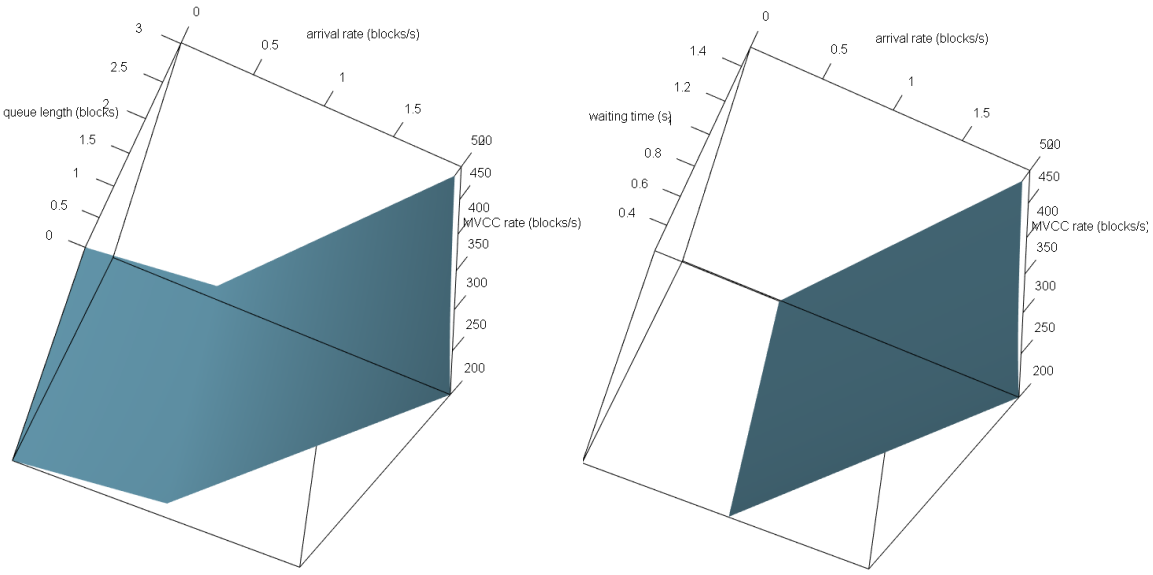

Figure 27. The length and waiting time with increasing the service rate of MVCC validation and the block arrival rate.

Nakaike et al. pointed out that a major performance bottleneck is incurred by accesses to the databases that store the ledger data [83]. The paper has given a series of approaches to improving the database's performance to increase the throughput of the ledger updating. Figure 28 depicts the tendencies of the average queue length and waiting time for committing nodes as the ledger updating rate (μ_5) is increased from 5 to 10 blocks/s, given a fixed block arrival rate ($\lambda_3 = 2\ block/s$), VSCC validating rate ($\mu_3 = 4.96\ blocks/s$), MVCC validating rate ($\mu_4 = 196.08\ blocks/s$) and the block size (80).

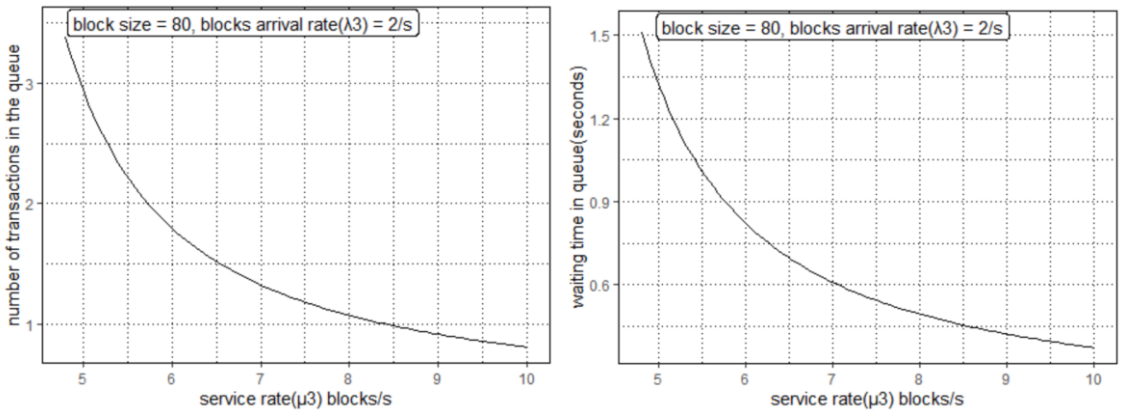

Figure 28. The length and waiting time with increasing the service rate of the ledger updating.

From figure 28, the average queue length and waiting time almost linearly decrease while the ledger updating rate is growing before the rate reaches 8.5 blocks/s. The figure also shows that the average queue length is less than 1 when the ledger updating rate reaches 10 blocks per second. This result shows that improving the database's performance is another cost-effective approach that can effectively increase the throughput of the committing node. The same as the results that would be obtained by increasing the VSCC validation rate, Figure 29 shows that the node can still get a pretty good performance if the ledger updating rate is increased from 4 to 10 blocks/s, after the arrival rate increases to 3 blocks/s.

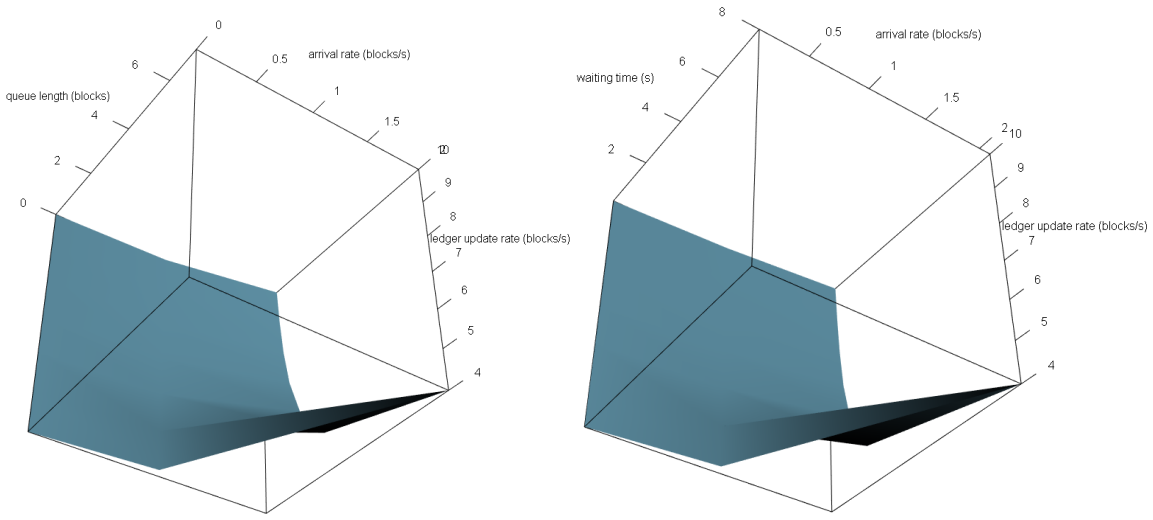

Figure 29. The length and waiting time with increasing the service rate of the ledger updating and the block arrival rate.

3.4.3 Impact of Number of Channels

Channels isolate transactions from one another. Transactions submitted to different channels are ordered, delivered, and processed independently of each other. While the number of channels to use and what channels to transact on are determined by the application and participant combinations, it has significant implications on platform performance and scalability [80]. In this section, equations (34) and (35) will be used to reveal the performance that is impacted by increasing the number of channels in Hyperledger Fabric.

Figure 30 plots the tendencies of the average queue length and waiting time in Hyperledger Fabric by increasing the number of channels from 1 to 20, given the fixed transaction arrival rate $\lambda_1 = 10\ tx/s$, ESCC rate $\mu_1 = 307.7\ txs/s$, block creating & delivery rate $\mu_2 = 980.4 txs/s$, VSCC validation rate $\mu_3 = 4.96\ blocks/s$, MVCC validation rate $\mu_4 = 196.8\ blocks/s$ and the ledger updating rate $\mu_5 = 4.801\ blocks/s$.

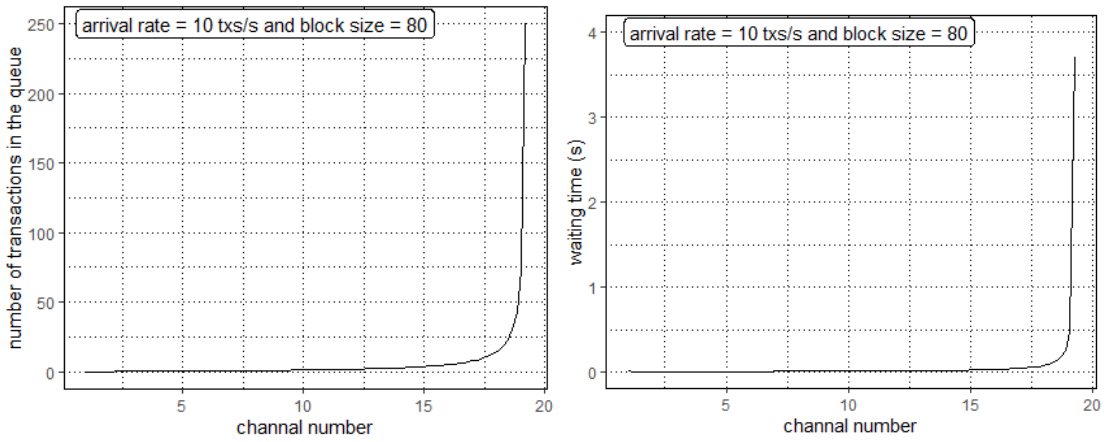

Figure 30. The queue length and waiting time of transactions in the network with increasing number of channels.

From figure 30, the average queue length and waiting time grow very slowly by increasing the number of channels from 1 to 17. After the number of channels reached 18, the average queue length and waiting time rose extremely fast because the number of channels $\times \lambda_1 = 18 \times 10 = 180\ txs/s$ is closing to $80 \times \frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5} = 192.8 txs/s$. The figure shows that the growth of the average queue length and waiting time is strongly inhibited once the number of channels exceeds 18. As the previous sub-section suggested, one way to solve this issue is to improve the throughput of VSCC validation and ledger updating by increasing the average rate of the three service rates ($\frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5}$) from 2.41 to 5 blocks/s. Figure 31 shows the tendencies of the average queue length and waiting time by increasing the average committing service rate ($\frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4 + \mu_4\mu_5 + \mu_3\mu_5}$) from 3 to 5 blocks/s, At the same time, the number of channels is growing from 0 to 30. The figure shows that increasing the VSCC validation rate and the ledger updating rate can get better performance when the number of channels exceeds 18.

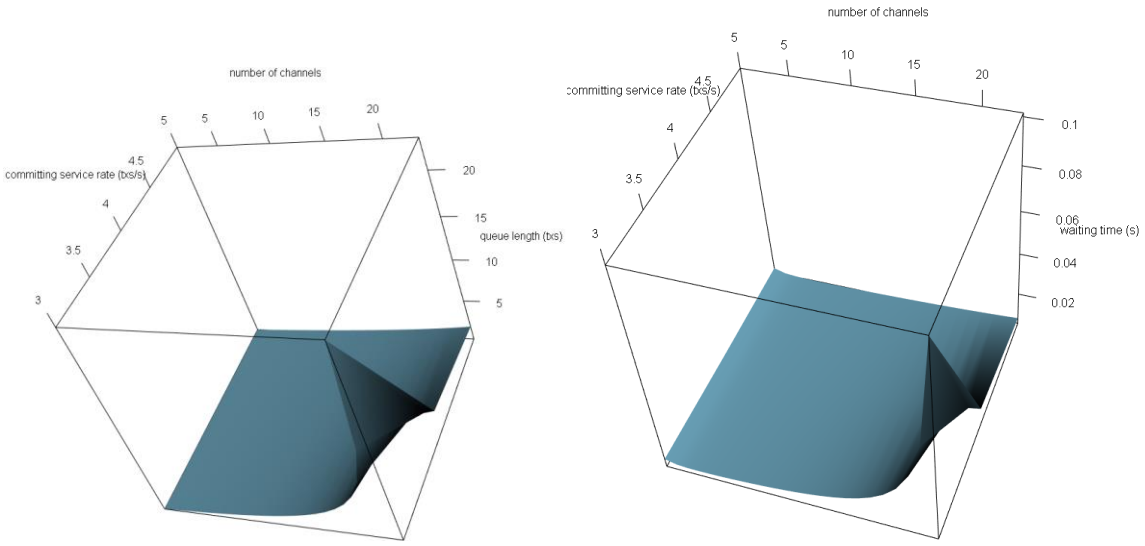

Figure 31. The queue length and waiting time of transactions in the network with increasing number of channels and the average committing service rate ($\frac{\mu_3\mu_4\mu_5}{\mu_3\mu_4+\mu_4\mu_5+\mu_3\mu_5}$).

3.5 Conclusions

This research presented quantitative models based on preliminary work in [41] to evaluate the performance of various types of nodes in a permissioned blockchain, namely, the Hyperledger Fabric. A set of variables considered in the simulation include the number of transactions and blocks in the queue and the waiting time of the transaction, along with a few constants that are referenced from published data for practical purposes, such as the mean service time and the block size. It has been observed that one performance bottleneck is located on or around the committing node and another is the number of channels, suggesting a focus area for future work as far as performance optimization is concerned. The analysis has been further extended beyond the one in [41] to demonstrate how the arrival rates and the service rates co-influence the performance in order to ultimately facilitate a more dynamic approach for the analysis and optimization. Also, the impact of the increased number of channels has been analyzed to reveal that the queue length and waiting time grow quite slowly as the number of channels is increased if the VSCC and ledger updating rates are increasing as expected. Therefore, both bottlenecks can

be solved by increasing VSCC and ledger updating rates. Extensive parametric simulations have been conducted and revealed that the results are as expected without loss of generality. Lastly, the versatility of the proposed quantitative models has been tested by extensive numerical simulations and analyses.

As part of the future work, a comprehensive empirical study of Hyperledger Fabric will be done by trying various configuration parameters and then comparing the experimental results to the numerical simulations that have been done in this research to evaluate and improve the proposed models. As [81] pointed out, performance bottlenecks, such as VSCC validation and ledger updating, are rooted in the endorsement policy and the number of peers. The future work intends to create analytical models that demonstrate how performance is directly related to the endorsement policy and the number of peers. The model will be able to provide some trade-off solutions to help with the definition of the endorsement policy for different applications case-by-case.

CHAPTER IV

PERFORMANCE MODELING AND ASSURANCE FOR CROSS-CHAIN

Abstract

Since the invention of Bitcoin in 2008, along with a rapidly increasing number of blockchain applications and users, there has been a surge in the number of blockchain networks with a variety of heterogeneous designs and functionalities. Hence, it becomes urgently necessary to allow interoperability between isomorphic and heterogeneous blockchains. To address the requisites, the research proposed a cross-blockchain communication model and a performance model based on m/Cox/1 queueing model. The cross-chain communication model considers two distinct types of communication, such as atomic swaps and inter-ledger asset transfers. In the communication model, there are two types of communication protocols that are used to control transactions across chains: the hashed time-lock contract (HTLC) [18] for crossing isomorphic chains and the inter-ledger asset transfer protocol (IATP) [19] for crossing heterogeneous chains. In proposed performance model, a Poisson arrival process is assumed at a rate of λ, and two service rates are assumed to be exponentially distributed at a rate of μ_1 (asset swap/transfer pre-commit and verify) and μ_2 (transfer commit/rollback), respectively. Lastly, the selection ratio of a communication protocols between IATP and HTLC is assumed to be at a rate of p and $(1-p)$, respectively. Extensive numerical simulations are conducted to study the impacts of communication request arrival rates, communication service rates, communication traffic rates, and the proportion of the communication protocols on L (number of transactions in the system)

and W (average waiting time in the system of transaction). This research established a sound theoretical foundation to identify the interrelation and impacts between various design variables on the performance and ultimately revealed an optimal solution to a high-performance cross chain design across isomorphic and heterogeneous blockchains.

4.1 Introduction

Since the inception of Bitcoin as the first decentralized ledger currency in 2008, the topic of blockchains (or distributed ledgers) has evolved into a well-studied field both in industry and academia [106]. In just ten years, tens of thousands of blockchains and blockchain technology-based applications have emerged. However, many blockchains exist in the market since each blockchain is isolated from one another. Individual blockchains have been designed with particular use cases in mind, so they feature specific strengths, limitations, and different degrees of decentralization [107]. The fact that blockchains operate in isolation has mostly made it impossible for people to enjoy the full benefits of ledger technology. Cross-chain technology seeks to solve the inability of different blockchains to communicate with one another by enabling interoperability between blockchains, thus making it easy for them to share information [108].

Cross-chain interoperability allows two different blockchains to transfer value or data and communicate between them. It involves the use of various methods to transfer value or data between different blockchains, and the choice of method depends on the specific use case and the blockchains involved. The mechanisms involved in transferring value or data over blockchains are summarized as Figure 32:

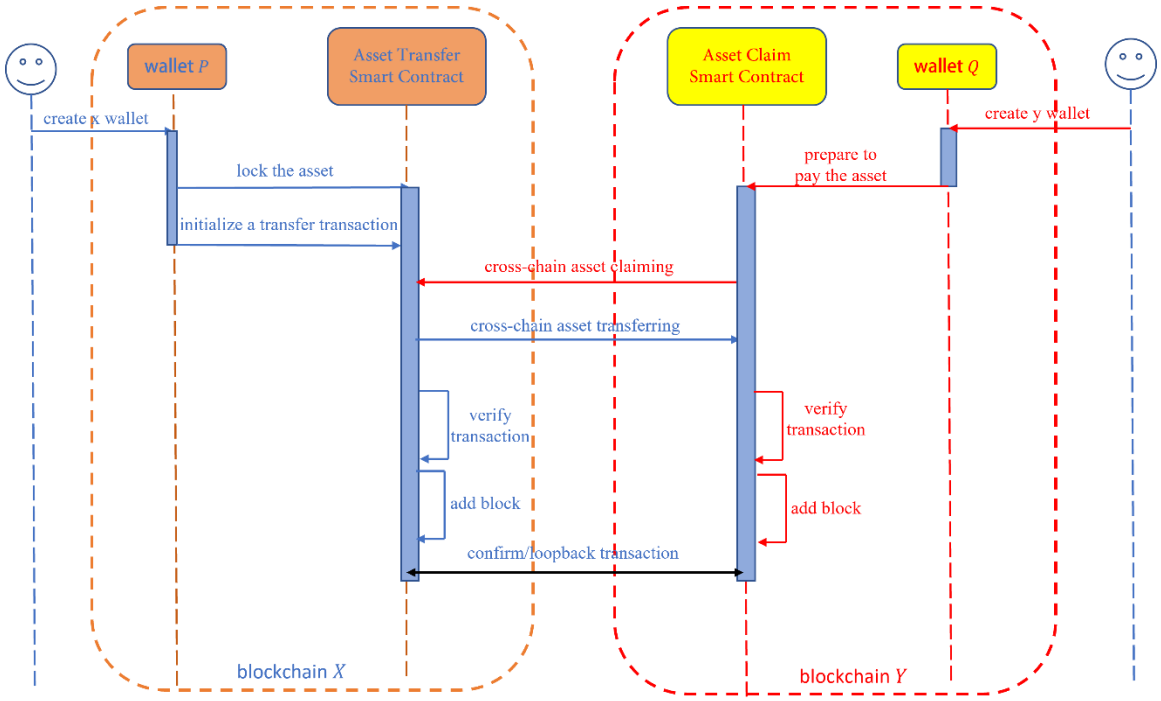

Figure 32. Mechanisms of cross-chain interoperability

- Wallet Creation: A user creates a wallet on the blockchain network they want to use, which generates a unique address that can send and receive cryptocurrency.

- Transaction Initiation: The user initiates a transaction by creating a digital signature that verifies their ownership of the funds they want to send.

- Cross-Chain Interoperability: To transfer value between different blockchains based on the cross-chain protocols, such as atomic swaps, inter-ledger.

- Transaction Verification: The transaction is broadcast to the network, and nodes on both blockchains validate and verify the transaction's details and confirm the user's ownership of the funds.

- Block Addition: Once the transaction is verified, it is added to a new block of transactions on the originating blockchain and then added to the receiving blockchain through a cross-chain interoperability solution.

- Confirmation: The transaction is considered complete once it has been added to both blockchains and several subsequent blocks have been added on top of it, confirming its validity.

- Fee Payment: In most cases, users need to pay a small fee to miners or validators for processing their transaction and adding it to the blockchain.

There are two participants involved in cross-chain communication: a source blockchain and a target blockchain. The source blockchain is the blockchain on which the transaction is initiated to be executed on a target blockchain. While the general purpose of interoperability comes down to a blockchain exposing its internal state to another, over-the-chain asset transfers rely on a cross-chain communication protocol that facilitates interoperability between different blockchain networks and enables the exchange of data between the networks. Through the protocol, users can communicate without the involvement of intermediaries. The cross-chain protocols can be classified as acting as applications or being part of blockchain platforms. When they act as an application, they do not need changes to the blockchain platform software to operate. These protocols are appropriate for situations in which users are unwilling or unable to modify their blockchain platform software. In contrast, other protocols require changes to the underlying blockchain software to work. They are part of the blockchain platform. They do not need contracts or servers external to the blockchain to operate [109]. There are two primary cross-chain protocols, respectively: 1) Inter-ledger [106], which is a set of actions for asserting transfer across heterogeneous blockchain systems, and 2) Hashed Timelock Contract (HTLC) [110],

which is a mechanism for trustless cross-chain atomic swaps. A high-level architecture of the cross-chain bridge model is shown in figure 33.

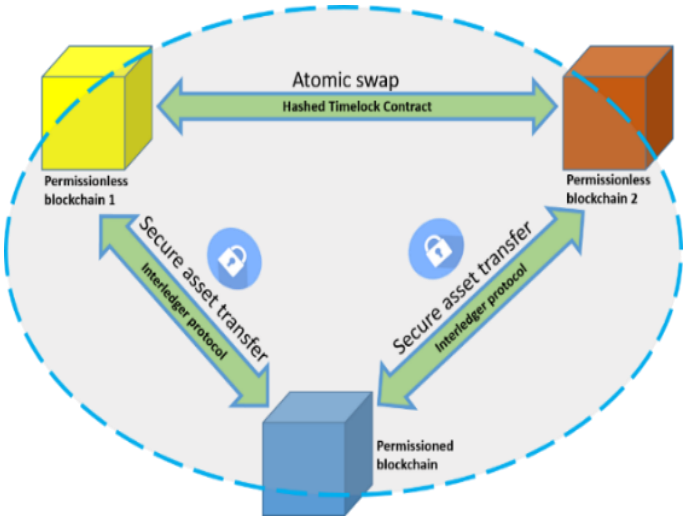

Figure 33. Architecture of cross chain communication

Cross-chain interoperability is a very complex and intensive computation process for transferring transactions over blockchains in which involves cross-chain interoperability solutions to facilitate the transfer, followed by the usual mechanisms of transaction initiation, verification, block addition, confirmation, and fee payment. There are at least two major cross-chain communication protocols involved in the process to control the communication between the different blockchains based on the specific use case and the blockchains involved. It is positive and necessary to seek or generate a quantitative model to study the impacts of the performance of the value transfer and ultimately provide the optimal solution.

A quantitative model and analysis of the performance of the cross-chain interoperability that is based on inter-ledger [106] and hashed timelock contract [110] is to be addressed and resolved in this research with respect to an extensive [42] and practical set of design and performance-related variables. There have been few adequate yet practical performance models targeting the specific behavior of cross-chain interoperability found, to the best of the authors' knowledge. H. Su et al.

[111] proposed a graph weight approach to calculate the cross-chain interaction probability. With this approach, the authors develop a quantitative condition-trigger model as the basic model of analysis and introduce the dimension in the edge weight, then use the weight of the graph (formed by different actions) to analyze the probability of cooperation, finally evaluating the condition-trigger model and its different trigger types, the all-trigger and dynamic-trigger. The evaluation shows that the method can analyze the action-cooperation relationship between smart contracts on different blockchains. Y. Jiang et al. [112] propose a cross-chain framework to integrate multiple blockchains for efficient and secure IoT data management to increase the security and scalability of the blockchains. By proposing a data access control model and designing a particular transaction type to provide fine-grained access control of data to blockchains to solve the privacy issues of cross-chain activities. The model merges transactions in the control station and confirms them based on the notary mechanism. The results of the evaluation demonstrated better protection of privacy. However, the model couldn't provide any information about the effectiveness and efficiency of transferring the transactions across the blockchains. B. Pillai et al. [113] proposed an application-level cross-chain communication model that presents a generalized form of cross-chain communication with a blockchain system, then analyzed the performance of the model and briefly compared it with other systems that employ a mother blockchain as an intermediary. The analysis shows that the model is relatively simple to implement in any blockchain system that supports smart contracts. The experimental results show that the model achieves relatively better performance than the mother blockchain-based models since the mother blockchain-based model must go through three major steps of operations to complete a cross-chain communication process, while the model only needs two steps. To deal with a lower performance in cross-chain data transferring, T. Lin et al. [114] proposed: 1) an improved cross-blockchain consensus algorithm, which is implemented based on the mortgage model instead of a probability model; 2) a cross-chain protocol with transverse expansion capacity, which would support the message transmission among chains; 3) a high-performance cross-chain blockchain network structure. The

authors developed a cross-chain network architecture that was imbued with an improved consensus algorithm. By the evaluations, it shows that the architecture can gain higher transaction efficiency, handle the security issues of cross-chain, and support large-scale business applications.

In this context, a cross-chain communication model is proposed to trace the operations and their performance of cross-chain transactions in blockchain networks in general. The model can be used to observe the realistic behaviors of cross-chain communications. The parametric simulations show that the model is accurate when it comes to the real-world scenarios. Moreover, the model can be used to perform extensive parametric studies to identify the influence factors of a system and can ultimately help to improve its performance, availability, and security. This research is organized as follows: The details of the proposed model and model analysis will be presented in the next section; they will be followed by numerical simulation results to validate the efficacy, and then the last section will conclude and discuss the work.

4.2 Proposed Model for Cross Chain Communication Protocols

4.2.1 Proposed Cross-chain Communication Protocol and the Assumptions

The different types of blockchain operate on different protocols and standards, which make it difficult to transfer assets between the blockchains. When a blockchain network is congested with too many transactions, it can slow down the processing of transfers and increase the fees associated with those transfers. Further, transferring assets between blockchains requires the use of cryptographic signatures and other security measures to prevent unauthorized access or theft. If these measures are not properly implemented or maintained, it can pose a security risk to the assets being transferred. Transferring assets from one blockchain to another requires careful consideration of the technical, practical, and regulatory challenges involved. To address these issues and make cross-chain transfers more seamless and secure, a cross-chain communication

protocol is proposed to provide the value or data transfer capabilities, such as asset swapping or asset transfer across two blockchains, by employing the Cross-Chain Communication (CCC) System Model [106] and Hashed TimeLock Contract (HTLC) [110]. The cross-chain communication protocols involve two decentralized networks (i.e., blockchains) X and Y, with underlying ledgers L_x and L_y, respectively. There are two wallets, P and Q, such that P and Q are executed on X and Y, respectively. Transaction of P denoted by TX_P is supposed to be committed on L_x, and likewise TX_Q on L_y. For Cross-Chain Communication (CCC) protocol, it can be viewed as a synchronization of transactions of P and Q such that Q writes TX_Q to L_y only after it is confirmed that P has written TX_P to L_x. To this end, P must convince Q that it has created a transaction TX_Q which was posted on L_x. Specifically, transactions of Q must verify that at a given instant of time t the ledger state $L_x(t)$ contains TX_P. The transactions of P and Q can trigger the state transition in its respective underlying Cross-Chain Communication by 1) committing a transaction to its respective underlying ledger L, or 2) stopping interactions within the communication. For Hashed TimeLock Contract (HTLC) protocol, it can be viewed as an asynchronization of transactions of P and Q such that Q writes TX_Q to L_y and P writes TX_P to L_x at the same time. To this end, P must promise Q that it will create a transaction TX_p to L_x as soon as Q writes TX_Q to L_y. The transactions of P and Q can trigger the state transition in its respective underlying Hashed TimeLock Contract by 1) committing a transaction to its respective underlying ledger L, or 2) stopping interactions within the communication. The mechanism of the proposed communication protocols is limited to performing two different communication processes across the blockchains, namely, atomic swaps and asset transfers, in a decentralized and trustworthy manner. The following assumptions are made for the communication protocol:

- The consensus participants in both X and Y are honest such that they comply with the designated protocol. The underlying data structures of X and Y are blockchains, i.e., append-only sequences of blocks, where each block contains a pointer to its predecessors.

- All the arrivals in cross chain communications are in a Poisson distribution at a rate λ, and the distributed ledger state progresses in discrete rounds indexed by natural numbers $r \in N$. At each round r, a new set of transactions (committed in a newly generated block) is written to the ledger L.

- The ordering of transactions in a block is crucial for their validity. However, for simplicity, the position of transactions in blocks are assumed to be correct in first-in-first-out (FIFO) manner.

- The two participators X and Y are no matter how synchronized and there is no clock drift between them.

- There are two transactions P (TX_P) and Q (TX_Q) in servers on two isomorphic or heterogeneous blockchains X and Y, respectively, such that X serves the transaction TX_P in exponential distribution at service rate μ_1 to write on L_x, and Y serves the transaction TX_Q in exponential distribution at service rate $\mu_2$to write on L_y.

4.2.2 Asset Transfer between Isomorphic and Heterogeneous Blockchains

Proposed cross-chain communication employs two different types of communication protocols, such as hashed time-lock contracts and the inter-ledger asset transfer protocol for transferring assets between isomorphic and heterogeneous blockchains, respectively. The cross-chain communication is parameterized by the involved blockchains X and Y and the associated ledgers L_x and L_y, the involved wallets P and Q, the transactions TX_P and TX_Q as well as their descriptions d_P and d_Q, respectively.

The asset transfer between isomorphic blockchains executes transactions in a time-bound manner between two users across two isomorphic blockchains. The communication can be realized either bidirectional or through routed channels to enable secure transfers of assets over X and Y simultaneously, without requiring trust on any of the intermediaries. There are two key features that distinguish the communication between X and Y from standard cryptocurrency transactions, such as: Hashlock. It is an asynchronous approach with a type of encumbrance that restricts the accessing to an output until a specified piece of data (pre-image) is publicly revealed. It has the useful property that once any Hashlock is opened publicly, any other Hashlock secured using the same key can also be opened; Timelock: is a synchronous approach with a function that restricts the spending of funds until a specific time (or block height) in the future. Figure 34 shows a Hash Time locked assert transferring process:

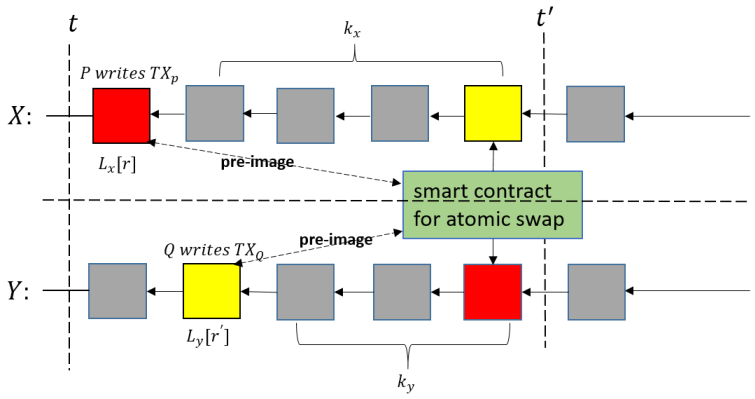

Figure 34. Hash Time locked assert transferring process.

- X opens a channel to Y, and Y opens a channel to the atomic swap.
- At round r, X generates a pre-image s and adds it to the payment along with an extra condition $Hash(s)$ for Y, then X lock the asset M with $Hash(s)$.

- Y provides the data that is used to produce a pre-image s, adds a copy under the same condition that X has put on the assert M, and then uses its channel to the smart contract to make the payment N at round r'.

- X has the original data that was used to produce the pre-image s, so X can use it to finalize the payment N and fully receive the payment from Y. By doing so, the assert transfer necessarily makes the pre-image s available to Y, then Y use it to claim the asset M from X before period $T = (t' - t)$ expires.

Heterogeneous blockchains are blockchains that have different technical specifications, such as consensus mechanisms, data structures, and smart contract languages. As the number of blockchains continues to grow, the ability to exchange information and value between heterogeneous blockchains becomes increasingly important for enabling new use cases and applications. The proposed cross-chain communication can also be employed for the transfer of goods, assets, or objects between two heterogeneous blockchains. An asset transfer procedure for value or data exchange between different types of blockchains be shown in Figure 35.

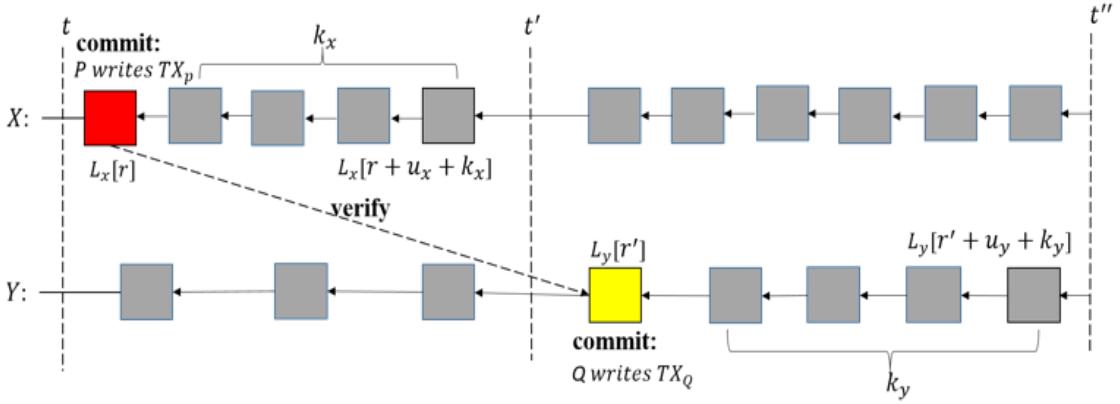

Figure 35. Cross chain asset transfer over heterogeneous blockchains

- Pre-Commit on X: Upon the arrival of the requests of Interledger asset transfer, a publicly verifiable commitment to execute the communication protocol is published

on X: P writes transaction TX_P on its local ledger L_x^P at time t in round r. Due to Persistence and Liveness of L_x, all honest parties of X will report TX_P as stable ($TX_P \in L_x$) in round $r + u_x + k_x$.

- Verify: The correctness of the commitment on X by P is verified by Q checking (or receiving a proof from P) that (i) $d_P = desc(TX_P)$ and (ii) $TX_P \in L_x$ hold. Due to Persistence and Liveness of X, it is assumed without loss of generality that (ii) will succeed at time t' which corresponds to round $r + u_x + k_x$ on X, if P was executed correctly.

- Commit on Y: Upon successful verification, a publicly verifiable commitment is published on Y: Q writes transaction TX_Q on its local ledger L_y^Q at time t' in round r' on Y. Due to Persistence and Liveness of L_Y, all honest parties of Y will report TX_Q as stable ($TX_Q \in L_Y$) in round $r' + u_y + k_y$, where u_y is the liveness delay and k_y is the "depth" parameter of Y.

- Abort: If the verification fails and/or Q fails to execute the commitment on Y, the CCC protocol can take an abort step on X such that any modifications TX_P made to the state of L_X are to be rollbacked. As blockchains are an append-only data structure, rollback requires broadcasting an additional transaction $TX_{P'}$ which resets X to the state before the commitment of TX_P.

4.2.3 Queueing Model of Cross-chain Communication

The proposed queueing model takes into consideration the proposed cross-chain communication protocol that employs the technologies of inter-ledger communication [106] and hashed time-lock contracts [110]. From a setup transaction for cross-chain asset transfer on the participant to the transaction that finally committed the transfer on the participant, there are two phases in which

multiple transactions are to be generated. An m/Cox/1 queueing model is assumed in the proposed model for asset transfer across isomorphic and heterogeneous blockchains. The model is shown in Figure 36.

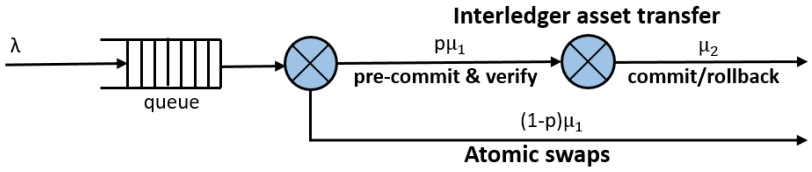

Figure 36. A m/Cox/1 cross-chain communication queueing model

Based on the mechanisms of inter-ledger and HTLC communication protocols, the Poisson process is assumed for a cross-chain asset transfer arrival at a rate of λ. The proposed model limited the types of transmission requests for cross-chain communication that can be selected to two: atomic swaps and inter-ledger asset transfers. The probability of an inter-ledger asset transfer is assumed to be p , so the probability of an atomic swap is $(1-p)$. If the atomic swap is selected, the whole process of asset exchange across X and Y will be completed within $t'-t$, so the atomic swap service can be assumed to be an exponential process at the rate $\mu_1 = \frac{1}{E[t'-t]}$. If the inter-ledger asset transfer is selected, a two-phase process will be triggered. The process is assumed as following:

- A pre-commit transaction ($TX_P \in L_x$) will be served in an exponential process at the service rate $\mu_1 = \frac{1}{E[t'-t]}$ at the beginning, and the results of the service will be verified on X as well.

- After the transaction is verified successfully, a new transaction ($TX_Q \in L_Y$) will be served on Y as an exponential process at the rate of $\mu_2 = \frac{1}{E[t''-t']}$, and the results of the service will be verified on Y. If Y fails to verify the transaction (TX_Q), a rollback transaction

$TX_{P'}$ will be served an exponential process at the rate of μ_1 on X chain to reset X to the state before the commitment of TX_P.

4.3 Cross Chain Model Analysis

In Figure 36, the cross-chain transfer requisition arrives according to a Poisson process at the rate of λ and is served according to a Coxian distribution [114]. This Coxian distribution Cox (x; μ_1, μ_2) is defined as a continuous probability distribution on the interval $[0, \infty)$, with parameters $\mu_1, \mu_2 \in R, p \in (0,1]$. The service time requires maximally 2 phases, where each phase $j = 1, 2$ is exponentially distributed with parameter μ_j. After each phase, the transaction needs a subsequent phase of service with probability p and finishes service with probability $1 - p$. Here, p is the probability that the cross-chain communication using the inter-ledger protocol, while $1 - p$ is probability that the cross-chain transaction takes HTLC protocol. The time spent in the phases are exponentially distributed with parameter $\mu_1$or μ_2, therefore the average time spent in that phase is $\frac{1}{\mu_j}, j = 1, 2$. The process of cross chain asset transfer follows that the probability that a cross chain transfer requires at least a phase of service (μ_1). Thus, the average service time $E(S)$ of cross-chain transfer is:

$$E(S) = \frac{1}{\mu_1} + \frac{p}{\mu_2} \tag{1}$$

and the traffic rate ρ:

$$\rho = \lambda E(S) = \lambda \frac{(\mu_2 + p\mu_1)}{\mu_1 \mu_2} \tag{2}$$

Let $\{U_i, V_j\}$, where $i = 0, 1, 2, \ldots n$ and $j = 0, 1, 2$ be a Markovian process that is defined for number of transactions (i) in the queue and the transaction in service is currently receiving

service at the (j) phase. Let π_{ij} be the steady-state probability that cross-chain asset transferring is in the (i, j) state if the traffic rate $\rho < 1$. Thus, the steady state-diagram as Figure 37:

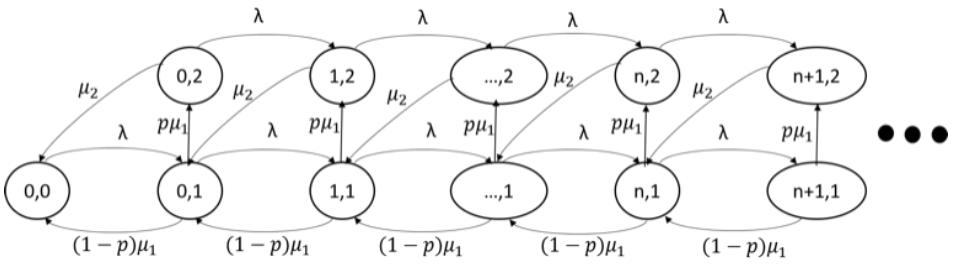

Figure 37. Steady state transition diagram

From above transition diagram, the balance equations can be wrote as following:

$$\lambda\pi_{0,0} = (1-p)\mu_1\pi_{0,1} + \mu_2\pi_{0,2} \tag{3}$$

$$(\lambda + \mu_1)\pi_{0,1} = \lambda\pi_{0,0} + (1-p)\mu_1\pi_{1,1} + \mu_2\pi_{1,2} \tag{4}$$

$$(\lambda + \mu_1)\pi_{i,1} = \lambda\pi_{i-1,1} + (1-p)\mu_1\pi_{i+1,1} + \mu_2\pi_{i+1,2}, \qquad i \geq 1 \tag{5}$$

$$(\lambda + \mu_2)\pi_{0,2} = p\mu_1\pi_{0,1} \tag{6}$$

$$(\lambda + \mu_2)\pi_{i,2} = \lambda\pi_{i-1,2} + p\mu_1\pi_{i,1}, \qquad i \geq 1 \tag{7}$$

$$\pi_{0,0} + \sum_{i=0}^{\infty}(\pi_{i,1} + \pi_{i,2}) = 1 \tag{8}$$

Let's define two generating functions:

$$g_1(z) = \sum_{i=0}^{\infty}\pi_{i,1}z^i \tag{9}$$

and

$$g_2(z) = \sum_{i=0}^{\infty} \pi_{i,2} z^i \tag{10}$$

then, the balance equations and the generating equations can be rewritten as following:

$$[-\lambda z^2 + (\lambda + \mu_1)z - (1-p)\mu_1]g_1(z) = \mu_2 g_2(z) - \lambda(1-z)\pi_{0,0,} \tag{11}$$

$$[\lambda(1-z) + \mu_2]g_2(z) = p\mu_1 g_1(z) \tag{12}$$

With $\pi_{0,0} + g_1(1) + g_2(1) = 1$, according to H.G. Perros [115], equations (11) and (12) can rewritten as following:

$$g_1(z) = \frac{\pi_{0,0}\rho_1\big(1 + \rho_2(1-z)\big)}{P_0 - P_1 z + P_2 z^2}, \qquad where\ \rho_i = \frac{\lambda}{\mu_i},\ \ P_0 = 1 + \rho_2 - p\rho_2,$$

$$P_1 = \rho_1 + \rho_2 + \rho_1\rho_2\ \ and\ P_2 = \rho_1\rho_2 \tag{13}$$

$$g_2(z) = \frac{\pi_{0,0}\rho_2 p}{P_0 - P_1 z + P_2 z^2} \tag{14}$$

Since $g_1(1) = \rho_1$, $\sum_{i=0}^{2}(-1)^i P_i z^i = \pi_{0,0}$, $g_2(1) = \rho_2 p$ and $\pi_{0,0} = 1 - \rho_1 - \rho_2 p$. Let $q_n, n = 0,1,2 \ldots$, be the steady state probability that there are transactions in cross-chain asset transferring, then get the generating function:

$$q_0 = \pi_{0,0},\ \ q_n = \pi_{n-1,1} + \pi_{n-1,2}$$

$$P(z) = \sum_{n=0}^{\infty} q_n z^n \tag{15}$$

and

$$P(z) = q_0 + z\big(g_1(z) + g_2(z)\big) = \frac{q_0(R_0 - R_1 z)}{P_0 - P_1 z + P_2 z^2}, \qquad where\ R_0 = P_0,\ R_1 = (1-p)\rho_1 \tag{16}$$

By dividing the numerator and denominator by P_0, get:

$$P(z) = \frac{q_0(1 - T_1 z)}{1 - S_1 z + S_2 z^2}, \qquad where\ T_1 = \frac{R_1}{P_0}, S_1 = \frac{P_1}{P_0}, S_2 = \frac{P_2}{P_0} \tag{17}$$

By differentiating equation (17), get the mean number of transactions in the queue L:

$$L = P'(1) = \frac{q_0[(-R_1)(P_0 - P_1 + P_2) - (R_0 - R_1)(-P_1 + 2P_2)]}{(P_0 - P_1 + P_2)^2} \tag{18}$$

Since $q_0 = \pi_{0,0,} = 1 - \rho_1 - \rho_2 p$, $P_0 - P_1 + P_2 = q_0$ and $R_0 - R_1 = 1$. Therefore, obtain:

$$L = \frac{\rho_1 + p\rho_2 + p(1-p)\rho_2^2 - p\rho_1\rho_2}{1 - \rho_1 - \rho_2 p} \tag{19}$$

Using Little's law, get the waiting time in cross-chain asset transferring W:

$$W = \frac{L}{\lambda} = \frac{\rho_1 + p\rho_2 + p(1-p)\rho_2^2 - p\rho_1\rho_2}{(1 - \rho_1 - \rho_2 p)\lambda} \tag{20}$$

4.4 Numerical Simulation

Numerical simulation is a faster and cheaper method of using mathematical models to simulate systems or processes without conducting physical experiments. In a numerical simulation, a model of the system is created using mathematical equations, and these equations are solved numerically to predict the behavior of the system over time. By defining performance metrics, the results of the simulation can be used to analyze the performance of the system under different conditions. Numerical simulation allows us to explore the behavior of a system under a wide range of conditions. Its' results can be useful for studying the sensitivity of the system to different parameters. Numerical simulation is a powerful tool for designing and optimizing systems and has been widely used in computer science. The primary objective of this numerical simulation is to reveal various preliminary performances of cross-chain communication of interest, such as the

average queue length and the transaction waiting time. In this section, the numerical simulation is separated into four subsections. The first subsection studies how the tasks of cross-chain communication impact the system. The second subsection studies how to improve cross-chain communication service performance by increasing the service rates and the effectiveness of reducing the traffic rates on the performance of cross-chain communication. The third subsection studies how the performance is impacted by various traffic rates. The last subsection studies how the proportion of the different kinds of cross-chain communication influences the asset transfer performance.

4.4.1 Impact of Arrival Rate on Performance

In this subsection, equations (19) and (20) be used to simulate the cross-chain communication model under the conditions of the service rates $\mu_1 = 30/s$, $\mu_2 = 40/s$ and probability $p = 0.2$ (the proportion rate of inter-ledger asset transfer = 20%), given the different arrival rates λ. Figure 38 shows the results of the simulations, which predict the performance impact of various arrival rates.

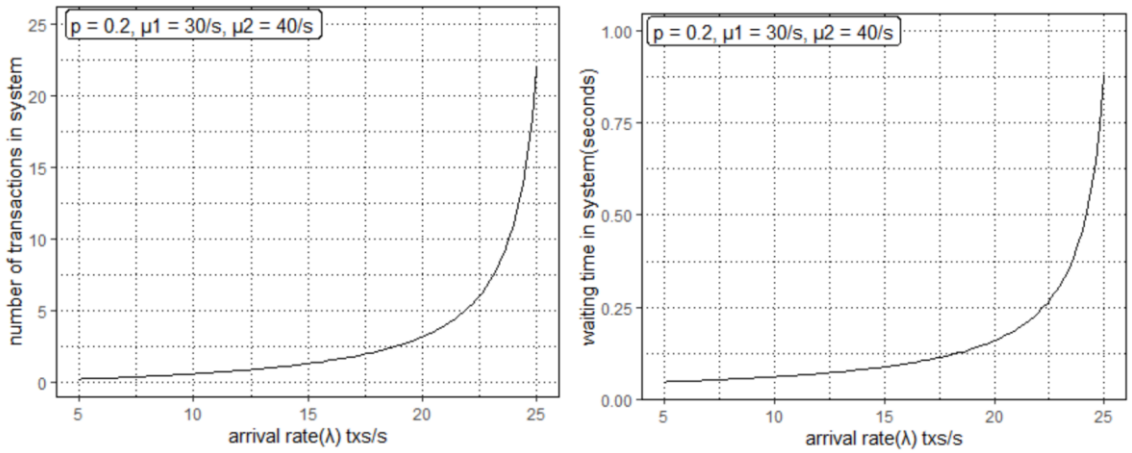

Figure 38. Impact of arrival rates

In figure 38, the number of transactions in the system and the waiting time for the completion of the transaction grow very slowly before the arrival rate reaches 20 transactions per second. After the rate passes over 20 transactions per second, the speed of the increasing number of transactions and the waiting time become very fast. When the arrival rate reaches 25 transactions per second, the system is at a bottleneck. To address the bottleneck, increase the service rates. (μ_1, μ_2) is needed. The next subsection will study how various service rates impact cross-chain asset transfer performance.

4.4.2 Impact of Service Rate on Performance

In this subsection, equations (19) and (20) will be used to simulate the cross-chain communication model under the conditions of the arrival rates $\lambda = 30/s$ and probability $p = 20\%$, given the different service rates $\mu_1,\ \mu_2$. Figure 39 shows the results of the simulations, which predict the performance impact of various service rates.

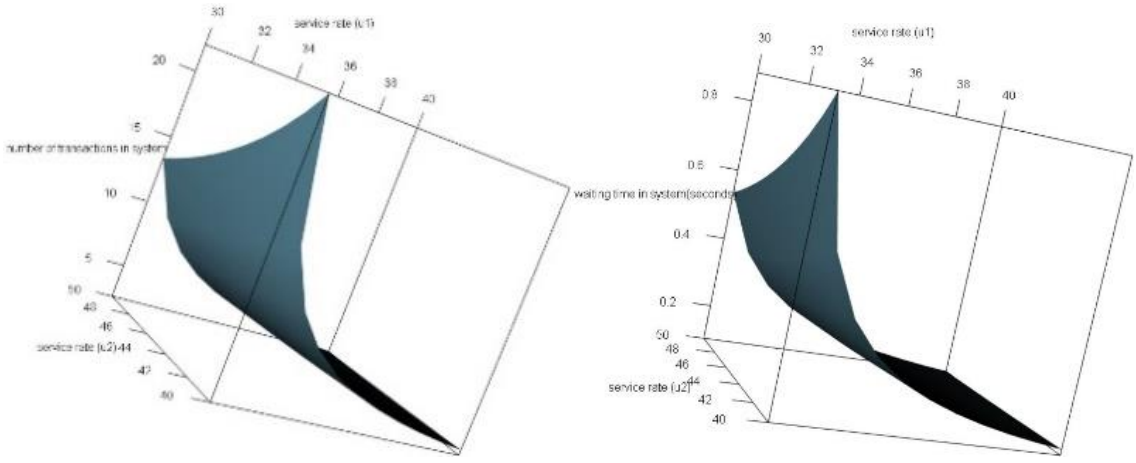

Figure 39. Impact of service rates

In figure 39, the limitation of arrival rate ($\lambda = 25/s$) can be broken by increasing the service rate μ_1 from 30 to 40 and the service rate μ_2 from 40 to 50. With the increase in service rates, the number of transactions in the queue drops from 25 to 0, and the waiting time reduces from 1 second to 0. The simulation shows the fact that both the arrival rate and service rate affect the

performance of cross-chain communication. The results show that the higher arrival rate can slow down atomic swaps and asset transfers, but the higher service rate can speed up cross-chain communication. The next subsection will study how the traffic rates ($\rho_1 = \frac{\lambda}{\mu_1}$, $\rho_2 = \frac{\lambda}{\mu_2}$) impact the communication.

4.4.3 Impact of Traffic Rate on Performance

In this subsection, equations (19) and (20) will be used to simulate the cross-chain communication under the condition that given the probability $p = 0.2$ and various pairs of different traffic rates ρ_1, ρ_2. Figure 40 shows the results of the simulations, which predict the performance impact of various pairs of traffic rates.

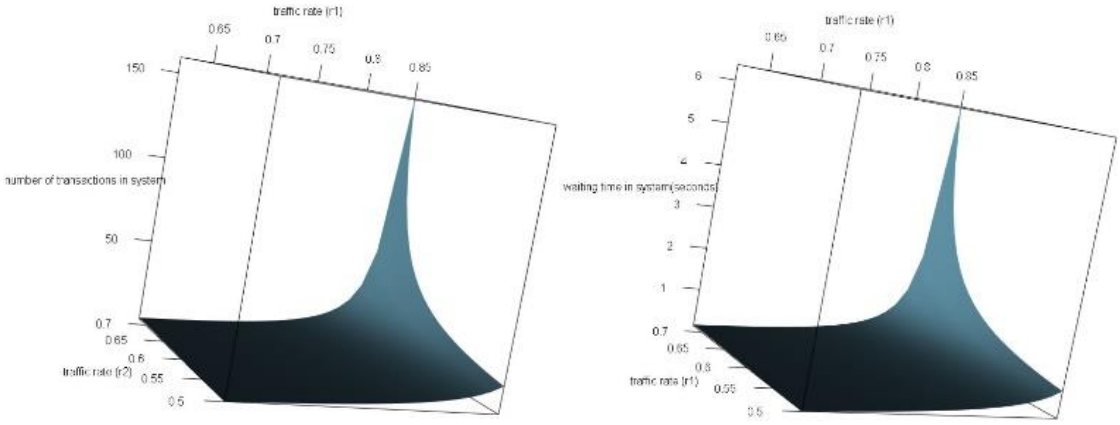

Figure 40. Impact of traffic rates

In figure 40, the performance getting worse gradually when the traffic rates ρ_1 and ρ_2 increase from 0.5 to 0.7 and 0.6 to 0.85, respectively. It also can be seen that the traffic rate ρ_1 seems to have more impact on the performance than the traffic rate ρ_2. The reason seems to be that the proportion of inter-ledger asset transfers ($p = 0.2$) is smaller than the proportion of atomic swaps ($1 - p = 0.8$) in this numerical simulation. The next subsection will study the effect of the proportions of the inter-ledger asset transfers and atomic swaps.

4.4.4 Impact of Proportion of the Inter-ledger Asset Transfer on Performance

In this subsection, equations (19) and (20) will be used to simulate the cross-chain communication under the conditions that given the service rates $\mu_1 = 30/s$, $\mu_2 = 40/s$ and arrival rates $\lambda = 25/s$, and various proportions of inter-ledger asset transfer p as well. Figure 41 shows the results of the simulations, which predict the performance impact of various proportions of inter-ledger asset transfer.

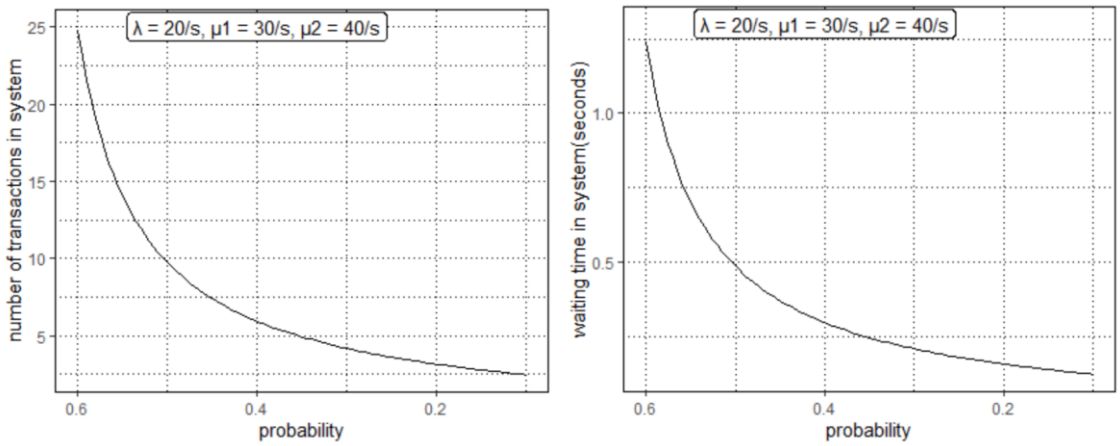

Figure 41. Impact of probability

In figure 41, the throughput of cross-chain communication is improved by reducing the proportion of inter-ledger asset transfers. This result implies that inter-ledger asset transfer is more expensive than the atomic swap. This conclusion suggests that increasing the number of atomic swaps or reducing transactions of inter-ledger asset transfer can improve cross-chain communication performance. Therefore, encouraging atomic swaps but limiting inter-ledger asset transfer seems like a feasible way to maximize the utilization of cross-chain communication performance.

In summary, the simulation results tell us to lower the traffic rates (ρ_1, ρ_2) by reducing the service times ($E[t' - t], E[t'' - t']$) through the asset transfer algorithms improvement to

mitigate the increasing verification time. It is also suggested to improve the efficiency of cross-chain communication by increasing the proportion of atomic swaps. However, these two suggestions are likely to encounter a technical challenge regarding how to maintain cross-chain utilization and security under the pressure of stringent performance requirements. Besides the simulations that have been conducted in this research, which have shown the impacts of the traffic intensity (ρ) and the proportion of inter-ledger asset transfer (p), the performance model also shows the impacts of changing the arrival rate (λ) and the service times ($\mu_1,\ \mu_2$). In addition to the model involving two participants (atomic swap and inter-ledger asset transfer), the performance model is readily extensible to the relay chain. [116].

4.5 Conclusions

This research has presented a cross-chain communication protocol that can transfer the asset across the isomorphic or heterogeneous blockchains and a queueing model based on the protocol to evaluate the performance of cross-chain communication with respect to the number of cross-chain transactions in the queue and overall waiting time, along with a practical validation work. A set of variables taken into account in this model and simulation include the number of cross-chain transactions in queue (L), the waiting time of the transaction (W) along with a few constants such as the transaction arrival rate (λ), cross chain service rates ($\mu_1,\ \mu_2$) and proportion of the two types of communication ($p, 1-p$). Parametric simulations have been conducted, and it has been demonstrated that the results are in great agreement with expectations and intuition. By giving different arrival rate λ in an ascending order, the queue length and waiting time increase very quickly after the rate reaches 25 per second. However, by increasing the cross-chain service rates or reducing the proportion of cross-chain communication between heterogeneous blockchains, the queue length and waiting time decreased very drastically. These trends imply that the traffic intensity ($\rho_1 = \frac{\lambda}{\mu_1},\ \rho_2 = \frac{\lambda}{\mu_2}$) and rate of inter-ledge asset transfer (p) are the major factors that

can affect the performance of cross-chain communication. This result suggests increasing the number of atomic swaps or reducing transactions for inter-ledger asset transfers to improve cross-chain communication performance. In conclusion, the proposed model provides a comprehensive yet theoretical basis to assure and ultimately optimize the design of cross-chain-based applications with respect to cost and performance.

CHAPTER V

CONCLUSIONS, DISSCUSION AND FUTURE WORK

A comprehensive study has been conducted on the performance modeling across the full spectrum of blockchain protocols in order to establish a sound theoretical foundation to optimize performance. Around the performance analysis based on the proposed quantitative models, a new idea that can potentially solve the drawbacks of blockchain is proposed and discussed. It is suggested that future work focus on developing a self-governed AI-based cloud computing architecture to serve the cross-chain community.

5.1 Conclusions

In Chapter II, a quantitative model based on queueing theory was proposed that aims to evaluate the performance of permissionless blockchain with respect to the number of transactions in the queue and overall waiting time, along with a practical validation work. A set of variables was considered in the proposed model, including the number of transactions in queue and the waiting time of the transaction, along with a few constants such as the transaction arrival rate and the block size. Numerical simulations were conducted, and it was demonstrated that the results are in great agreement and show evidence that traffic intensity and block size are the two major factors that can affect the performance of permissionless blockchain. This phenomenon encourages us to lower the system loading rate by reducing the block time during the consensus process (i.e., proof-of-work) through algorithm improvements to mitigate the increasing difficulty otherwise. It is also suggested to reduce the wait time by increasing the block size. However, the suggestions

are likely to encounter a technical challenge about how to improve the performance of a permissionless blockchain without losing its scalability and security. Through a series of analyses and simulations, it was proven that the proposed model can provide a comprehensive yet theoretical foundation to assure and optimize the design of a higher performance permissionless blockchain.

Chapter III presented multiple quantitative models to evaluate the performance of various types of nodes in a permissioned blockchain with specific reference to Hyperledger Fabric with respect to the number of transactions and blocks in the queue and overall waiting time, along with a practical parametric simulation. A set of variables was considered in the simulation, including the number of transactions and blocks in the queue and the waiting time of the transaction, along with a few constants that were referenced from published data for practical purposes, such as the mean service time and the block size. Extensive numerical simulations based on the versatility of the proposed quantitative models were conducted and showed that the results were as expected without loss of generality and that a performance bottleneck is located around the committing node. Based on the result of simulations, an analysis was conducted to demonstrate how the arrival rates and the service rates co-influence the performance to facilitate a more dynamic approach for the optimization. Also, the impact of the number of channels has been analyzed to reveal that the queue length and the waiting time grow quite slowly by increasing the number of channels, if VSCC or ledger updating is increased as expected. In conclusion, the research demonstrated that the proposed models for the various types of nodes, either individually or as a whole, provide a quantitative basis to assure and optimize the performance of each type of node in Hyperledger Fabric and ultimately the overall performance of permissioned blockchains.

In Chapter IV, a communication protocol across two heterogeneous blockchains and a new m/Cox/1 queueing model-based performance model are proposed. In the protocol, there are two distinct types of asset transfer that have been taken into consideration, such as atomic swaps and

inter-ledger asset transfers, and two types of communication controls that have been considered, such as the hashed timelock contract and the inter-ledger asset transfer control. In the performance model, the asset transfer request arrival is assumed to be a Poisson process, and the two services that provide the service for pre-committing, verification, and committing of asset transfer are assumed to be exponentially distributed. The ratios of the communication protocols are assumed to be constants. Extensive numerical simulations have shown the impact of asset transfer request arrival rates, service rates, and traffic rates, as well as the protocol ratio, on the number of transfers in the queue and the average waiting time for an asset transfer across blockchains. Various simulation results demonstrated that the performance model is valid as expected and showed the evidence that the proposed model can provide a valuable tool to assure and ultimately optimize the design of cross-chain-based applications.

5.2 Discussion

Blockchain is an incredibly disruptive technology that provides security and transparency levels never seen before and is desired not only by the finance industry but by any type of industry. However, blockchain technology is far from being a perfect disruptive technology. Some of its limitations prevent it from being widely used in the industry, such as its increased complexity, higher energy consumption, lower scalability, etc. In this research, a series of quantitative models based on queueing theory have been provided, and many numerical simulations have been conducted to try to find solutions for optimizing the performance and scalability by reducing the complexity and energy consumption and improving the efficiency of blockchain. Beside the few reasonable and necessary assumptions and the parameters that are referenced from published data, the proposed models have proven their strong ability to analyze, evaluate, and eventually optimize the dynamic blockchain-based application. It should be noted that the overall solution of the blockchain application always faces a dilemma: how to maintain blockchains that are both decentralized and secure under the pressure of stringent performance and scalability

requirements? This can be one of the reasons why there are so many blockchain networks with various types of consensus algorithms and different ledger storage systems existing in the world. To address the difficulty, a natural idea is to build another peer-to-peer network that is constructed by edge nodes of various blockchains by running specific smart contracts to provide the communication bridge between the blockchain networks [122]. There have been some solutions existing in the industry to provide a bridge for communication between the blockchains, such as Polkadot [117], Cosmos [118], and various blockchain exchange networks. However, those products couldn't provide a better performance and higher scalability solution to the industry due to a redundant consensus process and a lack of automatic regulation of workload for the asset transfers among the different blockchains. Inspired by the results that came from the research in Chapter IV, one way to improve the efficiency of cross-chain communication is by constructing a loosely coupled P2P network among all the cross-chain agents (the edge nodes that installed a cross-chain smart contract) and then dynamically routing the different cross-chain requests to the different destinations based on the different situations by dynamically pairing the edge nodes in order to guide the agents to efficiently fulfill cross-chain asset or data transfers. With this method, the asset or data can be quickly and efficiently routed to the destination blockchain, thereby improving the overall performance without losing scalability and security. Since the agents are not able to self-provide the guide to choose the right destination according to the circumstances due to a lack of understanding of the overall resource distribution, they need a blockchain oracle [123] to advise them on how to dynamically choose the path to reach the destination. Therefore, providing a blockchain oracle that can quickly and promptly find a valid guideline for adjusting the communication path for cross-chain agents is the key, which not only solves the bottleneck of asset or data transfer among many blockchain networks but also maintains scalability and security. Chainlink [124] seems to be able to provide similar functions through building a decentralized blockchain oracle network that connects smart contracts. One drawback of blockchain oracle is that it may not be able to provide the guideline on time since the

service is based on the results of off-line data analysis. Another is that using a blockchain oracle will tend to centralize cross-chain asset transfers around an oracle service provider such as Chainlink, thereby making the blockchain lose its purpose of decentralization eventually.

5.3 Future Work

Noting the drawbacks of the blockchain oracle, the future work will focus on developing a self-governed AI-based cloud computing architecture that provides a framework, which is the foundation of the cross-chain community, to support asset or data transfer among many of the various blockchain networks in the world. Based on the benefit of the highest level of transparency, blockchain provides a huge amount of trusted data to learn and analyze. With the assistance of machine learning technology, the cross-chain agents can find or be fed the pattern behind the information exchange in or out of the blockchain and the distribution of blockchain resources. In machine learning, data labeling is very expensive and lacks timeliness. Using reinforcement learning [119] can help the agent find the asset or data transfer patterns (namely, policies in reinforcement learning) without having to do the data labeling in blockchain. Also, the feasible incentive mechanism of blockchain can help reinforcement learning find the most cost-effective path for transferring assets or data. The path will advise cross-chain agents to find the most effective way to transfer assets or data based on individual requirements and situations, thereby improving blockchain overall performance. Since blockchain technology can organize a parallel computational environment by utilizing virtual machine technology, many groups can be created in the environment by autonomously or manually selecting the agents to form a group that includes three or four blockchains. Each agent in the group learns asset or data transfer through reinforcement learning and then competes to get the most accurate local model. Heterogeneous federated learning [120] iteratively averages the parameters in all local models into a global model, which is then evaluated and retrained into a new global model by the agents with the local data that includes the asset or data transfer across three or four blockchains. By iterating the

above collaborative training process tens of thousands of times, eventually a validated and reliable global model can be distributed to all the agents for advising and regularizing the asset or data transfer or information exchange in the entire blockchain world.

REFERENCES

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] Mastering Bitcoin. O'Reilly | Safari. [Online], Available: https://www.oreilly.com/library/view/masteringbitcoin/9781491902639/ch02.html

[3] D. Yaga, P. Mell, N. Roby, and K. Scarfone, “Blockchain Technology Overview (Draft NISTIR 82022)," https://csrc.nist.gov/CSRC/media/Publications/nistir/8202/draft/documents/nistir8202-draft.pdf.

[4] Wikipedia: Merkle tree. https://en.wikipedia.org/wiki/Merkle_tree

[5] Diego Geroni "Top 5 Benefits Of Blockchain Technology", [Online], Avaliable: https://101blockchains.com/benefits-of-blockchain-technology/

[6] "Can Bitcoin Be Hacked?" [Online]. Available: https://river.com/learn/can-bitcoin-be-hacked/#:~:text=Bitcoin%20is%20a%20relatively%20new,been%20uttered%20on%20the%20network.

[7] TJ Jung, "How transparency through blockchain helps the cybersecurity community", [Online], Available: https://www.ibm.com/blogs/blockchain/2019/04/how-transparency-through-blockchain-helps-the-cybersecurity-community/

[8] Wikipedia; Smart contract. https://en.wikipedia.org/wiki/Smart_contract

[9] IBM, “Blockchain overview”. [Online]. Available: https://www.ibm.com/topics/what-is-blockchain

[10] Chen, Huashan; Pendleton, Marcus; Njilla, Laurent; Xu, Shouhuai (12 June 2020). "A Survey on Ethereum Systems Security: Vulnerabilities, Attacks, and Defenses". ACM Computing Surveys. 53 (3): 3–4. arXiv:1908.04507

[11] coinbase, “What is blockchain infrastructure?” [Online]. Available: https://www.coinbase.com/cloud/discover/dev-foundations/blockchain-infrastructure

[12] crypto.com, "Consensus Mechanisms in Blockchain: A Beginner’s Guide" [Online], Available: https://crypto.com/university/consensus-mechanisms-in-blockchain#:~:text=Consensus%20for%20blockchain%20is%20a,trust%20in%20the%20Blockchain%20network.

[13] Investopedia, “Genesis Block” [Online]. Available: https://www.investopedia.com/terms/g/genesis-block.asp#:~:text=A%20Genesis%20Block%20is%20the,occur%20on%20a%20blockchain%20network.

[14] IBM, “Smart contracts defined”, [Online] Available: https://www.ibm.com/topics/smart-contracts

[15] Investopedia, “Decentralized Applications (dApps)” [Online]. Available: https://www.investopedia.com/terms/d/decentralized-applications-dapps.asp

[16] Noah Fields, “4 Types of Blockchain Technology Explained” [Online]. Available: https://komodoplatform.com/en/academy/blockchain-technology-types/

[17] Robinson, P. (2021). Survey of crosschain communications protocols. *Computer Networks*.

[18] "Hashed Timelock Contracts," 2017. [Online]. Available: https://en.bitcoin.it/wiki/Hashed Timelock Contracts

[19] Interledger Foundation, "Interledger: Interledger Protocol V4 (ILPv4)," 2021. [Online]. Available: https://interledger.org/rfcs/0027-interledger-protocol-4/

[20] J. Chow, "BTC Relay," 2016. [Online]. Available: https://media.readthedocs.org/pdf/btc-relay/latest/btc-relay.pdf

[21] Deloitte, "Blockchain and the five vectors of progress". [Online]. Available: https://www2.deloitte.com/us/en/insights/focus/signals-for-strategists/value-of-blockchain-applications-interoperability.html

[22] wipro, "Improving performance & scalability of blockchain networks". [Online] Available: https://www.wipro.com/blogs/hitarshi-buch/improving-performance-and-scalability-of-blockchain-networks/

[23] Wikipedia: Empirical research. https://en.wikipedia.org/wiki/Empirical_research

[24] Wikipedia: Analytical Performance Modeling. https://en.wikipedia.org/wiki/Analytical_Performance_Modeling

[25] "ANALYTICAL MODELING" [Online] Available: https://www.isixsigma.com/dictionary/analytical-modeling/#:~:text=Analytical%20modeling%20is%20a%20mathematical,with%20proper%20technique%20and%20care.

[26] Wikipedia: Stochastic modelling. https://en.wikipedia.org/wiki/Stochastic_modelling_(insurance)

[27] Wikipedia: Artificial intelligence. https://en.wikipedia.org/wiki/Artificial_intelligence

[28] Wikipedia: Machine learning. https://en.wikipedia.org/wiki/Machine_learning

[29] Wikipedia: Data analysis. https://en.wikipedia.org/wiki/Data_analysis

[30] Wikipedia: Operations research. https://en.wikipedia.org/wiki/Operations_research

[31] Wallstreetmojo Editorial Team, "Stochastic Modeling". [Online] Available: https://www.wallstreetmojo.com/stochastic-modeling/

[32] Wikipedia: Computer simulation. https://en.wikipedia.org/wiki/Computer_simulation

[33] Will Kenton, "Quantitative analysis (QA)," [Online] Available: https://www.investopedia.com/terms/q/quantitativeanalysis.asp

[34] Imed Bouchrika, "How to Write Research Methodology: Overview, Tips, and Techniques" [Online], Available: https://research.com/research/how-to-write-research-methodology

[35] Wikipedia: Markov chain. https://en.wikipedia.org/wiki/Markov_chain

[36] Wikipedia: Hidden Markov model. https://en.wikipedia.org/wiki/Hidden_Markov_model

[37] Wikipedia: Balance equation. https://en.wikipedia.org/wiki/Balance_equation

[38] Wikipedia: Quasi-birth–death process. https://en.wikipedia.org/wiki/Quasi-birth%E2%80%93death_process

[39] Wikipedia: Laplace–Stieltjes transform. https://en.wikipedia.org/wiki/Laplace%E2%80%93Stieltjes_transform

[40] Ke, Zuqiang and Nohpill Park. “A Queueing Model for Industrial Public Blockchains and Validation” *2021 22nd IEEE International Conference on Industrial Technology (ICIT)* 1 (2021): 712-717.

[41] Ke, Zuqiang and Nohpill Park. “Hyperledger Fabric Node Types and Performance Study.” *2021 Third International Conference on Blockchain Computing and Applications (BCCA)* (2021): 119-126.

[42] Ke, Zuqiang, Jongho Seol, Abhilash Kancharla and Nohpill Park. “Performance Modeling and Assurance for Cross Chain”. *2022 Fourth International Conference on Blockchain Computing and Applications (BCCA)* (2022): 305-311.

[43] "Cross-Chain Communication" [Online] Available: https://coinmarketcap.com/alexandria/glossary/cross-chain-communication

[44] Kathleen E. Wegrzyn, Eugenia Wang. “Types of Blockchain: Public, Private, or Something in Between” [Online] Available: https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between

[45] G. Wood, “Ethereum: A secure decentralized generalized transaction ledger.” Available online: https://ethereum.github.io/yellowpaper/paper.pdf. (accessed on 6 February 2020)

[46] D. Tapscott and A.Tapscott, “Blockchain revolution: How the technology behind Bitcoin is changing money, business, and the world”. Penguin, 2016.

[47] IBM, “The difference between public and private blockchain.” Available online: https://www.ibm.com/blogs/blockchain/2017/05/the-difference-between-public-and-private-blockchain/. (access on 6 Feb. 2020)

[48] Norman T.J. Bailey, “On queueing processes with bulk service”, J. Rpy. Statist. Soc. Ser. B16 (1954), 80-87.

[49] M.L. Chaudhry and J.G.C. Templeton, A First Course on Bulk Queue. Wiley, New York, 1983.

[50] “Why is it taking 20 minutes to mine this Bitcoin block?”, Available online: http://r6.ca/blog/20180225T160548Z.html. (accessed on 6 Feb. 2020)

[51] Fairley P., “The ridiculous amount of energy it takes to run Bitcoin” Available online: https://spectrum.ieee.org/energy/policy/the-ridiculous-amount-of-energy-it-takes-to-run-bitcoin. (accessed on 6 Feb. 2020)

[52] C. Decker and R. Wattenhofer, "Information propagation in the Bitcoin network," IEEE P2P 2013 Proceedings, Trento, 2013, pp. 1-10.

[53] N. Papadis, S. Borst, A. Walid, M. Grissa and L. Tassiulas, "Stochastic models and wide-area network measurements for blockchain design and analysis," IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, Honolulu, HI, 2018, pp. 2546-2554.

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

[55] Li, Q., Ma, J., Chang, Y. Ma, F., and Yu, H., “Markov processes in blockchain systems.” Comput Soc Netw 6, 5 (2019).

[56] “Current dynamics of transaction inclusion on ethereum”, Available online: https://medium.com/@ethgasstation/current-dynamics-of-transaction-inclusion-on-ethereum-ae8912edc960. (access on 6 Feb. 2020)

[57] “A gentle introduction to Ethereum”, Available online: https://bitsonblocks.net/2016/10/02/gentle-introduction-ethereum/#:~:text=Currently%20the%20maximum%20block%20size,block%20(1%2C500%2C000%20%2F%2021%2C000). (access on 6 Feb. 2020)

[58] “Bitcoin block time historical chart”, Available online: https://bitinfocharts.com/comparison/bitcoin-confirmationtime.html. (access on 6 Feb. 2020)

[59] “Ethereum expected block time chart”, Available online: https://etherscan.io/chart/blocktime. (access on 6 Feb. 2020)

[60] Breuer, L., and Baum, D., An introduction to queueing theory and matrix-analytic methods. Springer, 2005.

[61] “Mempool transaction count”, Available online: https://www.blockchain.com/charts/mempool-count. (access on 6 Feb. 2020)

[62] “Bitcoin block limits: sizing up the debate”, Available online: https://cryptobriefing.com/bitcoin-block-limits-sizing-up-the-debate. (access on 6 Feb. 2020)

[63] D. Kraft, “Difficulty control for blockchain-based consensus systems.” Peer Peer Netw. Appl. 9(2), 397–413 (2016)

[64] “Proof of stake FAQ” Available online: https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ. (access on 6 Feb. 2020)

[65] “The mystery behind block time” Available online: https://medium.facilelogin.com/the-mystery-behind-block-time-63351e35603a. (access on 6 Feb. 2020)

[66] Wikipedia: L’Hospital’s rule. [Online] Available: https://en.wikipedia.org/wiki/L%27H%C3%B4pital%27s_rule

[67] Wikipedia: Renewal theory. https://en.wikipedia.org/wiki/Renewal_theory

[68] Wikipedia: Little's law. https://en.wikipedia.org/wiki/Little%27s_law

[69] Wikipedia: Peer-to-peer. https://en.wikipedia.org/wiki/Peer-topeer

[70] Makani, S., Pittala, R., Alsayed, E., et al.: A survey of blockchain applications in sustainable and smart cities. Clust. Comput. (2022). https://doi.org/10.1007/s10586-022-03625-z

[71] Wikipedia: Consensus decision-making. https://en.wikipedia.org/wiki/Consensus_decision-making

[72] Alketbi, A., Nasir, Q., Talib, M.A.: Blockchain for government services-use cases, security benefits and challenges. In: 2018 15th learning and technology conference (L &T), pp. 112–119. (2018)

[73] Wikipedia: Proof of work. https://en.wikipedia.org/wiki/Proof_of_work

[74] Wikipedia: Proof of stake. https://en.wikipedia.org/wiki/Proof_of_stake

[75] Wikipedia: Byzantine fault. https://en.wikipedia.org/wiki/Byzantine_fault

[76] Chen, Q., Srivastava, G., Parizi, R.M., Aloqaily, M., Al Ridhawi, I.: An incentive-aware blockchain-based solution for internet of fake media things. Inf. Process. Manage. 57(6), 102370 (2020). https://doi.org/10.1016/j.ipm.2020.102370

[77] MIT Technology Review: Blockchain's real promise: automating trust. https://www.technologyreview.com/2019/06/13/102979/blockchains-real-promise-automating-trust/

[78] Wikipedia: Internet of things. https://en.wikipedia.org/wiki/Internet_of_things

[79] Wikipedia: Hyperledger. https://en.wikipedia.org/wiki/Hyperledger

[80] Thakkar, P., Nathan, S., Vishwanathan, B.: Performance benchmarking and optimizing hyperledger fabric blockchain platform. (2018). arXiv preprint arXiv:1805.11390

[81] Sukhwani, H., Wang, N., Trivedi, K.S., Rindos, A.: Performance modeling of hyperledger fabric (permissioned blockchain network). In: 2018 IEEE 17th international symposium on network computing and applications (NCA), pp. 1–8. (2018). https://doi.org/10.1109/NCA.2018.8548070.

[82] Vukolic´, M.: The quest for scalable blockchain fabric: Proof-ofwork vs. BFT replication. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics): preface, vol. 9591, pp. 112–125 (2016)

[83] Nakaike, T., Zhang, Q., Ueda, Y., Inagaki, T., Ohara, M.: Hyperledger fabric performance characterization and optimization using GoLevelDB benchmark. IEEE Int. Conf. Blockchain Cryptocurr. (ICBC) 2020, 1–9 (2020). https://doi.org/10.1109/ICBC48266.2020.9169454

[84] Dinh, T.T.A., Wang, J., Chen, G., Liu, R., Ooi, B.C., Tan, K.-L.: BLOCKBENCH: a framework for analyzing private blockchains. In: Proceedings of the 2017 ACM international conference on management of data, SIGMOD conference 2017, Chicago, IL, USA, May 14–19, 2017, pp. 1085–1100. (2017)

[85] Wikipedia: Ethereum. https://en.wikipedia.org/wiki/Ethereum

[86] Wikipedia: Cardano (blockchain platform). https://en.wikipedia.org/wiki/Cardano_(blockchain_platform)

[87] Wikipedia: Corda. https://en.wikipedia.org/wiki/Corda

[88] Wiki | Golden: MultiChain. https://golden.com/wiki/MultiChain-ZXE8RD9

[89] Wikipedia: Parity. https://en.wikipedia.org/wiki/Parity

[90] Kocsis, I., Klenik, A.: Towards performance modeling of hyperledger fabric. http://webspn.hit.bme.hu/*telek/cikkek/kocs17a.pdf

[91] Yuan, P., Zheng, K., Xiong, X., Zhang, K., Lei, L.: Performance modeling and analysis of a Hyperledger-based system using GSPN. Comput. Commun. (2020). https://doi.org/10.1016/j.com.2020.01.073

[92] Wang, C., Chu, X.: Performance characterization and bottleneck analysis of hyperledger fabric, pp. 1281–1286. (2020). https://doi.org/10.1109/ICDCS47774.2020.00165

[93] Hyperledger.org: Hyperledger-fabricdocs documentation. https://hyperledger-fabric.readthedocs.io/_/downloads/en/release-2.0/pdf/

[94] Androulaki, E. et al.: Hyperledger fabric: a distributed operating system for permissioned blockchains. In: EuroSys, pp. 30:1–30:15 (2018)

[95] Wikipedia: Stochastic process. https://en.wikipedia.org/wiki/Stochastic_process

[96] Wikipedia: M/M/1 queue. https://en.wikipedia.org/wiki/M/M/1_queue

[97] Harrison, Peter, Patel, Naresh M.: Performance modelling of communication networks and computer architectures. Addison-Wesley, Boston (1992)

[98] Wikipedia: Binomial distribution. https://en.wikipedia.org/wiki/Binomial_distribution

[99] Wikipedia: Geometric distribution. https://en.wikipedia.org/wiki/Geometric_distribution

[100] Luk, V.W.H., Wong, A.K.S., Lea, C.T., et al.: RRG: redundancy reduced gossip protocol for real-time N-to-N dynamic group communication. J. Internet Serv. Appl. 4, 14 (2013). https://doi.org/10.1186/1869-0238-4-14

[101] Wikipedia: Phase-type distribution. https://en.wikipedia.org/wiki/Phase-type_distribution

[102] Wikipedia: M/G/1 queue. https://en.wikipedia.org/wiki/M/G/1_queue

[103] Pollaczek, F.: U¨ ber eine Aufgabe der Wahrscheinlichkeitstheorie. I. Math. Z. 32, 64–100 (1930)

[104] Wikipedia: Burke's theorem. https://en.wikipedia.org/wiki/Burke%27s_theorem

[105] Kleinrock, L.: Queueing systems, vol. 1. Wiley, New York (1975)

[106] Zamyatin, A. et al. (2021). SoK: Communication Across Distributed Ledgers. In: Borisov, N., Diaz, C. (eds) Financial Cryptography and Data Security. FC 2021. Lecture Notes in Computer Science(), vol 12675. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-64331-0_1

[107] "BLOCKCHAIN INTEROPERABILITY – UNDERSTANDING CROSS-CHAIN TECHNOLOGY". Available online: https://www.leewayhertz.com/blockchain-interoperability-crosschain-technology/. (accessed on 14 May. 2022).

[108] "Blockchain Interoperability : Why Is Cross Chain Technology Important?". Available online: https://101blockchains.com/blockchain-interoperability/ (accessed on 14 May. 2022).

[109] Robinson, P. (2021). Survey of crosschain communications protocols. Computer Networks.

[110] "Hashed Timelock Contracts," 2017. [Online]. Available online: https://en.bitcoin.it/wiki/Hashed Timelock Contracts

[111] Su, H., Guo, B., Lu, J. et al. Quantitative cooperation analysis among cross-chain smart contracts. Neural Comput & Applic 34, 9847–9862 (2022). https://doi.org/10.1007/s00521-022-06970-7

[112] Jiang Y, Wang C, Wang Y, Gao L. A Cross-Chain Solution to Integrating Multiple Blockchains for IoT Data Management. Sensors (Basel). 2019 May 1;19(9):2042. doi: 10.3390/s19092042. PMID: 31052380; PMCID: PMC6539637.

[113] Pillai, B., Biswas, K., & Muthukkumarasamy, V. (2020). Cross-chain interoperability among blockchain-based systems using transactions. The Knowledge Engineering Review, 35, E23. doi:10.1017/S0269888920000314

[114] Cox, D. (1955). A use of complex probabilities in the theory of stochastic processes. Mathematical Proceedings of the Cambridge Philosophical Society, 51(2), 313-319. doi:10.1017/S0305004100030231.

[115] Perros, H.G. (1983). On the M/Ck/1 queue. Perform. Evaluation, 3, 83-93.

[116] J. Chow, "BTC Relay," 2016. [Online]. Available: https://media.readthedocs.org/pdf/btc-relay/latest/btc-relay.pdf

[117] Wikipedia: Polkadot. https://en.wikipedia.org/wiki/Polkadot_(cryptocurrency)

[118] Wiki-Golden: Cosmos. https://golden.com/wiki/Cosmos-5Z8MWMM

[119] Wikipedia: Reinforcement learning. https://en.wikipedia.org/wiki/Reinforcement_learning

[120] Wikipedia: Federated learning. https://en.wikipedia.org/wiki/Federated_learning

[121] Wikipedia: First principle. https://en.wikipedia.org/wiki/First_principle

[122] Diego Geroni: Blockchain Interoperability: Why Is Cross Chain Technology Important? https://101blockchains.com/blockchain-interoperability/?gclid=Cj0KCQjwz6ShBhCMARIsAH9A0qWyTqnBl7GMxbJYKAka1a3XQ0qRWTxKoY_v_awtX8_pzkgaFT_y6j4aAspsEALw_wcB

[123] Wikipedia: Blockchain oracle. https://en.wikipedia.org/wiki/Blockchain_oracle#:~:text=A%20blockchain%20oracle%20is%20a,that%20decentralised%20knowledge%20is%20obtained.

[124] Wikipedia: Chainlink (blockchain). https://en.wikipedia.org/wiki/Chainlink_(blockchain)

APPENDICES

A.1 Blockchain Technology

A.1.1 Blockchain Architecture

A blockchain network is designed as a decentralized network of nodes. It's a distributed ledgers architecture in which each node plays the role of a network administrator who voluntarily joins the network. Logically, a blockchain network can be seen as consisting of following five layers:

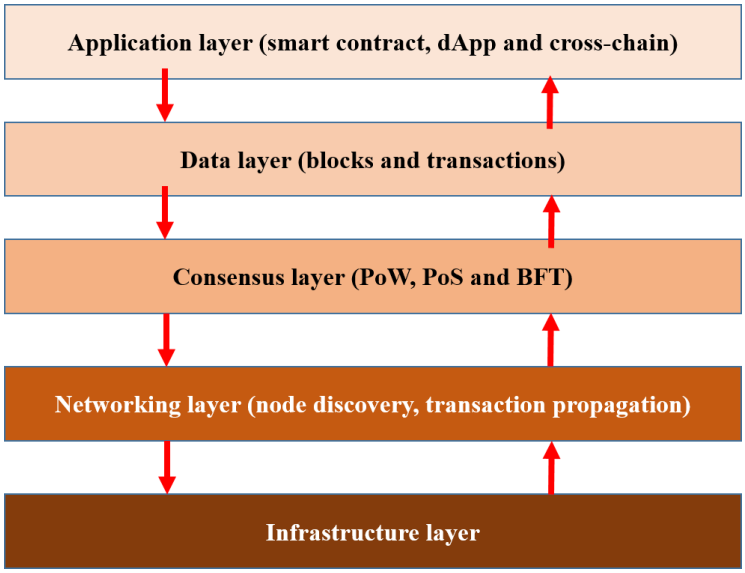

Figure 42. Blockchain architecture consists of five layers [10]

- Infrastructure layer: provides coordinate and maintain access to the frameworks that power the blockchains, and the underlying frameworks that need to operate their systems, such as cloud storage and security. [11]

- Networking layer: provides node discovery and transaction propagation. Node discovery in a peer-to-peer network is crucial. No network is defined when a new node boots up. The new node must detect at least one blockchain node to be a part of the network. There are several ways in which a node can identify peers and thus discover a network. Different blockchain frameworks use their own protocols to perform peer discovery and efficient routing. As a critical step in the process of ledger generation, transaction propagation makes it possible for each participant to have the same information about the state of the ledger. In a decentralized network, nodes connect to each other either directly or indirectly through other nodes. Once received by a node, valid transactions are immediately forwarded by that node to all other nodes that it is connected to, which then do the same with the nodes to which they are connected, until the transaction reaches a large percentage of the nodes in the network. This process typically takes a few seconds. [2].

- Consensus layer: consensus is a procedure in which the peers of a blockchain network reach agreement about the present state of the data in the network. Through this, consensus algorithms establish reliability and trust in blockchain networks. Consensus algorithms are used to verify transactions and maintain the security of the underlying blockchains. There are many different types of consensus algorithms, each with various benefits and drawbacks. Proof-of-Work, Proof-of-Stake, and Byzantine fault tolerance are three of the most widely used consensus algorithms.

- Data layer: provides the ledger storage in which the blocks and transactions are stored. The blocks hold batches of valid transactions that are hashed and encoded into a Merkle tree [4]. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data, with each additional block linking to the ones before it. These linked blocks form a chain. The iterative process confirms the integrity of the previous block, all

the way back to the initial block, which is known as the "genesis block" (Block 0) [13]. Hence, the integrity of the blocks and the transactions is assured.

- Application layer: provides a framework for running applications that include smart contracts, decentralized applications (dApps), and cross-chain communication protocols. A smart contract is simply a program stored on a blockchain that runs when predetermined conditions are met. It is typically used to automate the execution of an agreement so that all participants can be immediately certain of the outcome without any intermediary's involvement or time loss. It can also automate a workflow by triggering the next action when conditions are met [14]. A dApp is a digital application or program that exists and runs on a blockchain network of nodes instead of a single computer. It is outside the purview and control of a single authority and can be developed for a variety of purposes, including gaming, finance, and social media [15]. Cross-chain communication between blockchains allows different protocols to verify data and transactions without the intervention of a centralized third-party service. [43].

A.1.2 Type of Blockchains

Blockchain technology can not only be used in cryptocurrency and financial applications but can also be used in supply chain management, smart cities and IoT, etc. To meet the specific needs of different industries and use cases and optimize performance, security, and privacy based on those needs, it is necessary to have different types of blockchain networks to serve the different industries' applications. Currently, there are four main types of blockchain (public, private, hybrid, and consortium) with different use cases. While all blockchains are effectively peer-to-peer networks connected via nodes that execute transactions and add new blocks, the pathways to those nodes can either be permissionless or permissioned. Within that range of restriction lies the

difference between four types of blockchains. On one end of the spectrum is a permissionless blockchain, and on the other end is a permissioned blockchain [16].

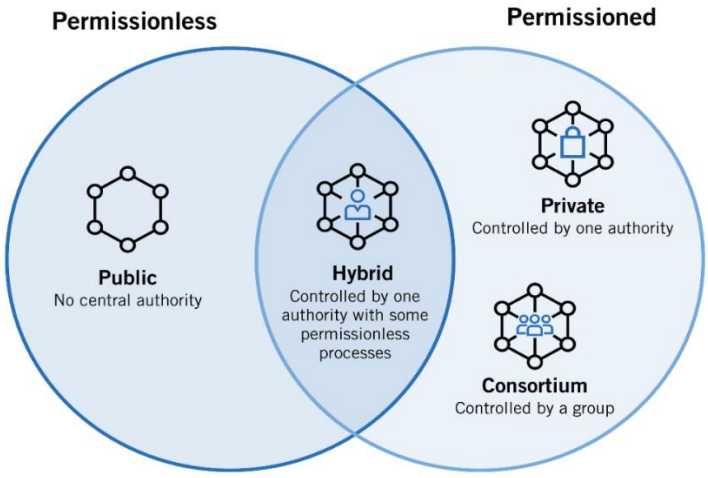

Figure 43. Relationship among four types of blockchains [44]

- Permissionless blockchain has to be open and accessible to all, therefore anyone can use their computer to become a network's node. Once the software is downloaded and installed on the computer, such a blockchain node can then perform mining, verify transactions, or access the entire ledgers. Permissionless blockchains can be used to displace traditional financial systems. There are many of Permissionless blockchain network. Bitcoin, Ethereum and Cardano [86] are the most famous.

- Permissioned blockchain is restricted as to who can join the network to become a node or access the network. These node lists are vetted by leading organizations, which can, at will, decide whether to constrict or expand the network. Correspondingly, permissioned blockchains are private distributed ledgers, commonly referred to as enterprise blockchains. Permissioned blockchains are a great asset when organizations that want to secure information flow without exposing it to the public eye. For this reason, companies use them for internal auditing, voting, asset management, logistics management, and

more. Corda [87], Hyperledger, and Multichain [88] are examples of a few permissioned blockchains.

A.1.3 Cross-chain Communication

Since Bitcoin was invented in 2009, many blockchain networks have been created with their own protocols and ecosystems. Today, there are more than 1,000 active blockchain networks. Each blockchain stores different kinds of data and transactions. If the transactions are carried out on the same type of blockchain network, it should be relatively easy to control. However, when there is a need to carry out transactions that will touch more than one type of blockchain network, things get complicated. Therefore, cross-chain communication protocols are needed to manage the transactions over different types of blockchain networks. Cross-chain communication protocols play a vital role in enabling blockchains to share and access their data and interoperate with one another. There are various inter-blockchain communication protocols, such as Atomic Swap, Polkadot, Inter-Blockchain Communication, Ren, Chainlink, and Lightning Network, etc. The protocols can be classified as Hashed Timelock Contracts (HTLC), Inter-ledger, and Bitcoin Relay [17]:

- Timelock Contracts (HTLC) are a mechanism for trustless cross-chain atomic swaps. That is, the technique allows two parties to swap value on one blockchain for value on another blockchain. Agreement occurs off-chain, with on-chain consensus used to ratify the earlier off-chain agreement.

- Inter-ledger is a set of payment protocols for sending value across heterogeneous blockchain networks. Senders communicate with receivers via connector nodes. Consensus is achieved by having the prepared messages contain a hash and the fulfill messages contain the corresponding preimage.

- Bitcoin Relay [20] allows users of Ethereum to do actions based on Bitcoin transactions using the Simplified Payment Verification approach described in the Bitcoin white paper. Simplified Payment Verification relies on block headers being transferred between blockchains.

A.2 Empirical Modeling

Empirical modeling is the process of developing models based on observed data or experimental results. The goal of empirical modeling is to understand the behavior of a system or process by analyzing the data and to use the insights gained to make predictions or optimize performance. A typical empirical modeling process involves the following steps:

- Collect data: It collects data on the system or process being modeled. This process includes conducting experiments, running simulations, or gathering observational data.

- Explore the data: The data is analyzed to identify patterns and trends and to gain insights into the behavior of the system. Explore data may use statistical methods, data visualization tools, and other data analysis techniques.

- Formulate a model: Based on the conclusion of the data analysis, a model can be formulated to describe the behavior of the system. The model can be a set of linear equations, or a complex nonlinear model, or a machine learning model.

- Evaluate the model: The model is evaluated to determine its accuracy and predictive power. The evaluation process may involve using cross-validation techniques or other statistical measures to assess the model performance.

- Use the model: Once the model has been validated, it can be used to make predictions, optimize performance, or guide decision-making. The model may also be refined or updated based on new data or observations.

Empirical modeling is particularly useful when the underlying physical mechanisms are complex or poorly understood, or when theoretical modeling approaches are not practical or feasible. Empirical modeling is widely used in engineering, science, and business and is often combined with analytical modeling to gain a more complete understanding of complex systems.

A.3 Analytical Modeling

Analytical modeling is the process of developing mathematical equations and models to describe and analyze a system or process. The goal of analytical modeling is to understand the behavior of the system or process under different scenarios or conditions and to make predictions or optimize performance based on the model. An analytical modeling process usually involves the following steps:

- Define the system: Clearly define the system that will be modeled. This process involves identifying the key components, inputs, outputs, and performance merits evaluation.
- Formulate assumptions: Assumptions are made about the behavior of the system, based on the available information, and understanding of the problem. The assumptions help to simplify the problem and make it more tractable.
- Develop mathematical equations: Mathematical equations are developed to describe the behavior of the system. The equations may be derived from first principles [121], or they may be based on empirical data or observations.
- Solve the equations: The mathematical equations are solved to obtain solutions that describe the behavior of the system. Analytical solutions need to be in a closed form, sometimes they may require numerical methods to obtain solutions.

- Interpret the results: The results of the analytical model are interpreted to gain insights into the behavior of the system. The results may be used to make predictions, to optimize performance, or to guide decision-making.

- Validate the model: The model is validated by comparing its predictions with empirical data or observations. If the model accurately predicts the behavior of the system, it can be used with confidence to make decisions and guide further analysis.

Analytical modeling can be applied to a wide range of systems and processes, including physical systems, financial systems, and biological systems. It is a powerful tool for understanding and optimizing complex systems and is widely used in engineering, science, and business. Analytical modeling and empirical modeling are all deterministic approaches, which means that they assume that the variables and parameters of the system are known with certainty, although sometimes an analytical model can be used to predict the behavior of a system under different conditions. In other words, if you want to use the model to estimate the probability of certain events occurring and to analyze the impact of random variability on the system's behavior, stochastic modeling may be a viable choice.

A.4 Stochastic Modeling

Stochastic modeling is a mathematical framework used to describe systems that exhibit random behavior or uncertainty. It is widely used in engineering to assess and design various technical and information systems, such as computer and communication networks, software systems, and distributed systems. Stochastic models can be used to provide insights into the behavior of complex systems that cannot be easily modeled using analytical methods alone. A stochastic modeling process is like an analytical modeling approach; additionally, it should also include the following two components:

- Probability distribution: It is used to describe the likelihood of a particular outcome or event occurring. Poisson and exponential distributions are the most common probability distributions that can be used in stochastic modeling.

- Markov chain: It is used to describe the probabilistic transitions between different states of a system. Markov chains are commonly used to model random processes.

Stochastic modeling is a powerful tool for understanding and predicting the behavior of complex systems that exhibit random behavior or uncertainty. A queueing model is a specific type of stochastic model that is based on queueing theory to study and evaluate the performance of systems.

A.5 Queueing Theory

Queueing theory is the study of waiting in lines, or queues. It involves the mathematical modeling and analysis of systems where customers arrive at a service facility and wait to be served by one or more servers. The theory aims to understand the behavior of these systems, such as the expected waiting time, the utilization of the servers, and the probability of a customer having to wait in line. It includes the following key concepts:

- Arrival process: The pattern of customer arrivals, which can be modeled as a Poisson process or a more general arrival process.

- Service process: the time it takes for a server to serve a customer, which can be modeled as an exponential distribution or a more general service process.

- Queue discipline: The rules that govern the order in which customers are served, such as first-come, first-served or priority-based.

- Queue length: the number of customers waiting in line at any given time.

- Waiting time: The average time that a customer needs to wait until the service is completed.
- Utilization (traffic rate): The proportion of time that a server is busy serving customers, as opposed to idle.

Queueing theory can be used to analyze and optimize various types of systems. It can be used to make predictions about system behavior and guide decision-making, such as determining the bottleneck, optimizing the algorithms, and configuring the number of servers to minimize waiting times.

VITA

Zuqiang Ke

Candidate for the Degree of

Doctor of Philosophy

Thesis: A STUDY ON PERFORMANCE MODELING AND ASSURANCE OF CROSS/PERMISSIONLESS/PERMISSIONED CHAINS

Major Field: Computer Science

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in Computer Science at Oklahoma State University, Stillwater, Oklahoma in May 2023.

Completed the requirements for the Master of Science in Computer Science at Oklahoma State University, Stillwater, Oklahoma in 2017.

Completed the requirements for the Bachelor of Science in Computer Science at Shanghai Jiao Tong University, Shanghai, China in 1987.

Experience:

Oklahoma State University, Graduate Teaching Assistant
Aug 2019 – May 2023.

Journal reviewer, *Security and Communication Network* 2022 – present.

Vanda Group, Chief Architect Jun 2004 – Jan 2013.

Computer and Technologies Holdings Limited, Technical Manager
Jul 1992 – May 2004.

Shanghai Jiao Tong University, Programmer Analyst Jul 1987 – Jun 1992.