ROBUST AND EFFECTIVE CONSENSUS APPROACHES FOR BLOCKCHAIN SYSTEMS

by

Lirui Tian, Master of Networks and Security

Thesis Submitted in Fulfilment of the Requirements
for the Degree of Master of Science (Computing) by Research

from

College of Sciences and Engineering, University of Tasmania

Supervisors:
Associate Professor. Quan Bai
Doctor. Zehong Cao

University of Tasmania
Augest, 2021

Declaration of Originality

I declare that this thesis contains no material which has been accepted for a degree or diploma by the University or any other institution, except by way of background information and duly acknowledged in the thesis, and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due acknowledgement is made in the text of the thesis.

Signed: _____
Lirui Tian

Date: 19 August 2021

Authority of Access

This thesis may be made available for loan and limited copying in accordance with the *Copyright Act 1968*

Signed: _____
Lirui Tian

Date: 19 August 2021

ABSTRACT

Reaching consensus on the global state of a ledger among distributed participants is an important challenge in blockchain systems. There are many consensus approaches studied to required that trust should be reduced or eliminated. The most commonly used consensus approaches in blockchain systems, for example, Proof of Work (PoW) approach consumes large amounts of power, but transaction throughput is limited. Consensus approaches such as Proof of Stake (PoS) is proposed to mitigate this outstanding issue. However, these consensus methods have concentration of wealth of participants, allowing "wealthy" stake holders to have high probability controlling block's generation. That conducts a small subset of nodes have power to control creation of blocks in blockchain network. Furthermore, most existing consensus methods failed to maintain a proper balance between security and performance in the ledgers.

This thesis analyses some of the most commonly used consensus approaches in blockchain systems. The thesis presents detail on consensus approaches for reaching the agreement. Based on this analysis, two novel blockchain consensus approaches are proposed. Proof of SecureStake (PoSS) is proposed to address imbalance of wealth and the security of the ledgers over block generation, and Proof of Assurance (PoAssu) is proposed to enhance performance in transaction throughput for per second.

ACKNOWLEDGEMENTS

The process of pursuing the Master of Science (Computing) by Research (M.Sc by Research) is really very challenging. The completion of my M.Sc by Research thesis turns out to be the outcome of a long journal of learning, exploring and researching new knowledge.

It is my great pleasure to acknowledge some of many people who have inspired, supported and encouraged me all along the way, especially during the COVID-19 pandemic period. First and foremost, I would like to express my sincere gratitude to my primary supervisor Associate Professor. Quan Bai who gave the opportunity to start my academic career. I greatly appreciate his patience, enthusiasm and immense knowledge, which make him an excellent supervisor and the best supervisor that I ever had before. I am also extremely grateful to my secondary supervisor, Dr. Zehong Cao, for his valuable advice, encouragement and sincere guidance during my study. Many thanks for your supervision which was vital to me in achieving my goals. I also greatly appreciate College of Sciences and Engineering, University of Tasmania for full-tuition scholarship and research opportunity to me.

Second, I also would like to thank all my friends, and colleagues in our Lab for their friendship and helps.

Finally, I greatly appreciate my family for their love, support and understanding.

This thesis is dedicated to
My parents & wife

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Blockchains have become popular technology, since 2008 Satoshi Nakamoto released Bitcoin in public [1]. Bitcoin is the first blockchain application which provides features of decentralized and immutability. The "trust-free" promotes great benefits to the security and privacy of many applications. The original Bitcoin system utilizes the block hash values and organizes data as a chain of the block [2, 3].

All nodes in a blockchain system are able to store and verify the transactions that are processed in the systems, and propose new blocks in blockchain network. The chain begains at a genesis block with index 0, and newly block appended to last block on the chain. However, today's blockchain is combined with timestamp together. The ideal of timestamp is introduced to digitally verify a document [4]. In blockchain, timestamp can be used with blocks and appended on the chain. The linked blocks can be presented as a hashchain in the system [5, 6] with timestamp. Nowadays, various blockchain techniques have been studied for improving and extending the new functions and features to the application layer of decentralised payment. On the one hand, most applications of blockchain benefits for different industries. For example, Ethereum system [7] can perform conditional payments and execute complicated code on the blockchain by using smart contract. There are some blockchains [8, 9, 10, 11] are proposed to fulfill new missions and engage the blockchain into critical network infrastructure. These extensions are proving the way to the existing system to apply to finance, transport, and healthcare industry. On the other hand, there are long-term issues which are related to balance of performance and security in consensus protocols.

It is very challenging to analysis the consensus approaches in different blockchain systems with complex networks. First, the most of proof-based consensuses have

shortcomings in selection of participants for participating consensus. For example, a primary concern is concentration of wealth where the "wealthy" nodes are more likely to get elected. An attacker that controls most of the "wealth" can exclude and modify the ordering of transactions. That effect leads to a small subset of nodes having a power to control the generation of the block, and it is also potential to increase degree of centralisation which against the nature of blockchain decentralisation. In the security aspect, it is vulnerable for attacks like guessing attacks. Second, transaction throughput is very low in most of mainstream blockchain system. Low transaction throughput healthy impacts efficiency of real world trading market.

By considering the aforementioned challenging issues and the research gaps, this thesis provides research into novel blockchain consensus approaches to mitigate concentration of wealth where the "wealthy" nodes are more likely to get elected for the block generation, and increase randomness of election for mitigating guessing attack. Also, the transaction can be confirmed immediately and transaction throughput can be enhanced.

1.1 Blockchain Transaction and Architecture

This section aims to introduce the characteristics of structures of blockchain transaction and architecture.

In blockchain, system is required to verify users's identity when there is a transaction between two or more parities. For example, there is a transaction between Tom and Jack, Tom sends Jack 5 dollars, when the request of this transaction is coming to a verification; this verification is required to ensure that message is indeed came from Tom. Figure 1.1 presents a verification's overview. To complete that work, cryptography is applied: the public key and private keys [12], which are digital signature key pairs. When a sender wants to send a transaction, sender can sender's private key to "sign" this transaction, and receivers are able to identify it by using sender's public key. In Bitcoin, the elliptic curve digital signature algorithm [13] is used for creating a signature for each transaction made by users.

The verifier needs to verify it that entity belongs to the right person or not, by checking the sender's public key, because public key is publicly accessible. After that, using the sender's public key with request transaction to verify function to get result: false or true. Then, it can be identified that the requester is the true sender

if the result returns true back. Also, if the adversary wants to crack this signature to make a fake transaction, then the adversary needs to guess 2^{256} cases, which is impossible due to the signature in each transaction contains 256 bits. In a blockchain system, the verifier also is required to check the validity of the transaction, and each block contains information like many validated transaction.

Figure 1.2 presents an example of the Blockchain. To be more specifically, An architecture of block inside shows in Figure 1.3. The block also contains core fields in block header:

- Prev_Hash: That can be a reference for pervious blocks, because it is link of a perviously block in the chain. All the information can be inputted to a hash function and get a hash value, after that hash value will be assigned to filed as $Prev\_Hash$ in newly increased block. In the Bitcoin, a Sha-256 is applied for blockchain system in order to gain this value [14].
- Timestamp: It is to mark the time for each transaction on the blockchain.
- Tx_Root: That can be known as Merkle root [15]. In this filed, it contains the hash value for all validated transactions of the block. Figure 2 shows an example of Blockchain, all the transactions are applied hash function to get each hash value; then each two hash values combine with pair for another round of hash function in order to get another hash value for this pair. That work will continue until there is only a single entity for a block. Thus, that process can be stand for the Merkle root.
- Version: When the node proposing the block appended to the chain, it will contain protocol version in this field.
- Nonce: It is to require miners proving the efforts for mining in order to get right to append block to the chain. Nonce filed is applied in PoW based blockchain.
- Bits: The mining difficulty level of the PoW has been indicated in this filed.

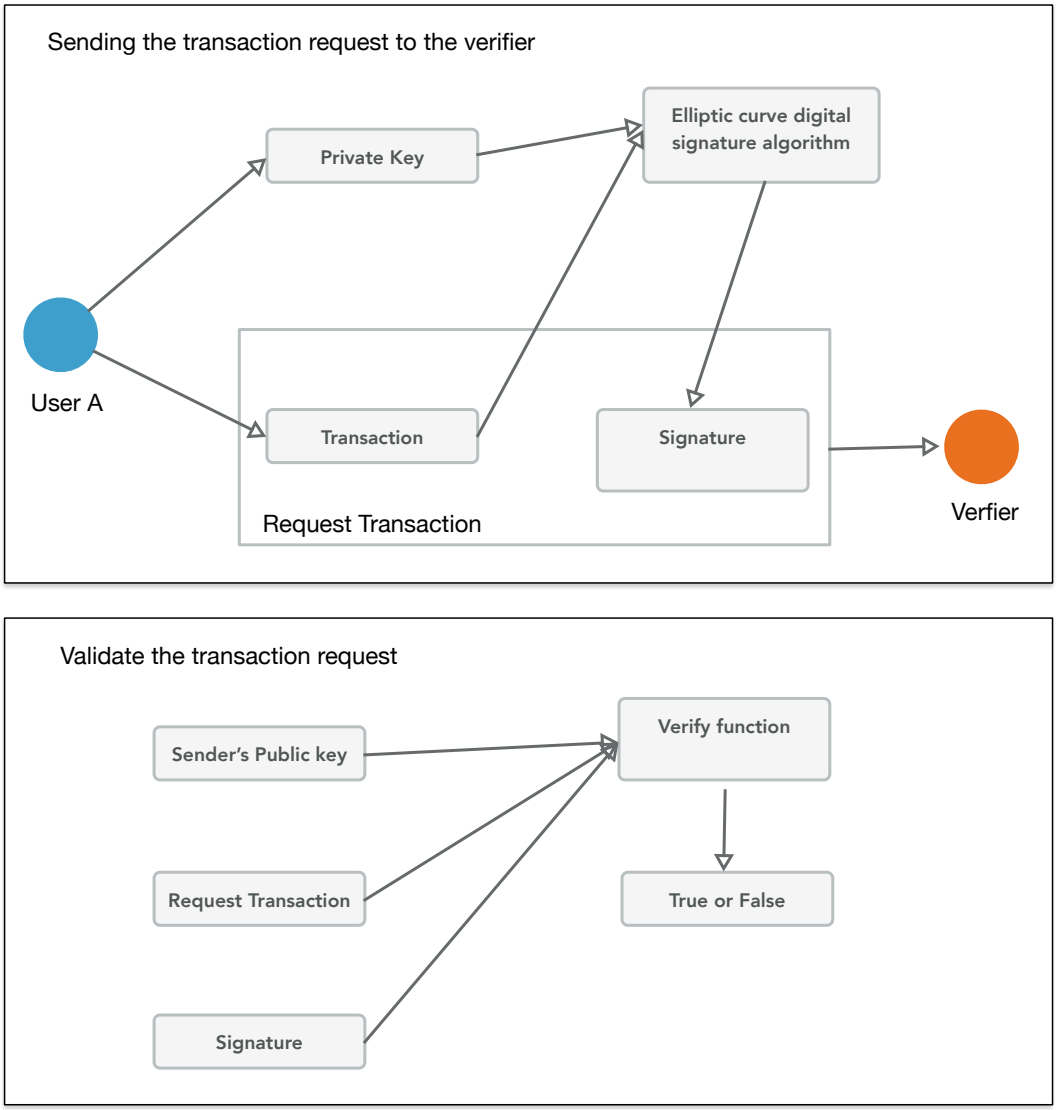


Figure 1.1: How a verifying system checks if the transaction is requested by the true sender

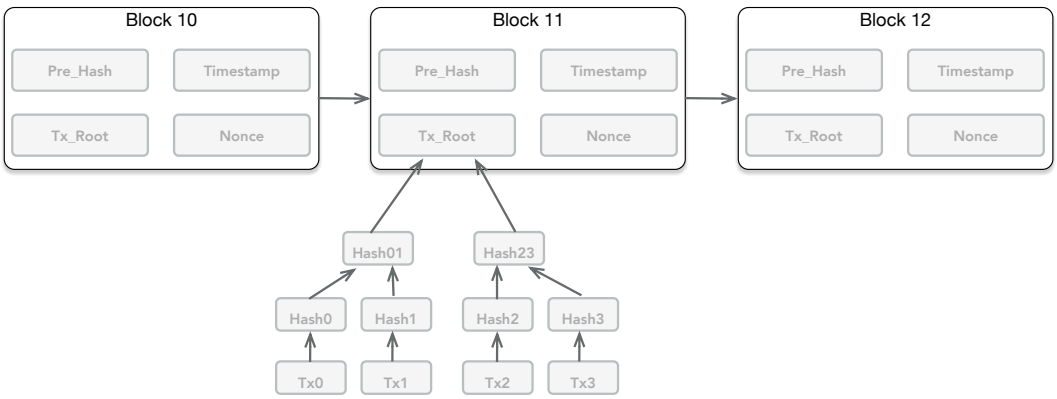


Figure 1.2: An example of Blockchain [70]

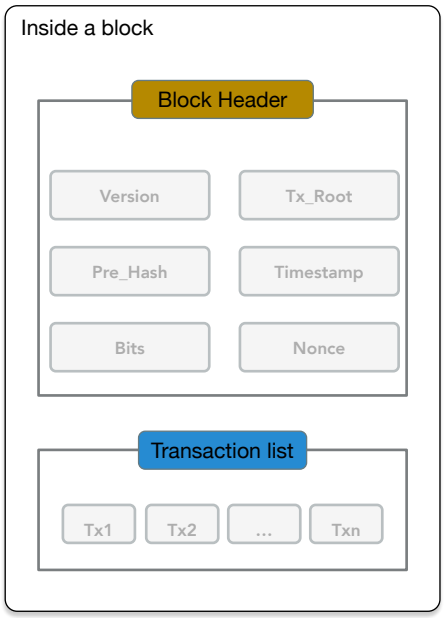


Figure 1.3: The fields inside a block

1.2 Consensuses in Blockchain

This section aims to introduce the outline of consensus approaches in blockchains.

Blockchain is a complex system which can be considered as permissionless and permissioned for current blockchain technologies [16]. A permissionless (Public) blockchains provide an open environment for nodes participating without restriction on number of participants to join. Nowadays, nodes are welcome to join and leave in the blockchain network anytime. Permissionless blockchain systems encourage the more miners to join the blockchain network to get involved with verifications. Also, the more miners are joining in the permissionless blockchains, the more decentralised degree is achieved in a blockchain system. While a permissioned (Private) blockchain is limited on nodes participating [17], and decentralised degree is low. Furthermore, consensus approaches is the core components in all types of blockchains [18]. However, consensus protocol is different between the permissionless and permissioned blockchains. In blockchain systems, most consensus approaches have a concept which is concentration on wealth, which allows "wealthy" stakeholders more like to have controlling on block generation, especially in PoS based blockchain systems.

The mining environment is important for users participating in blockchain, especially in a permissionless environment. Miners are willing to participate in the blockchain system as one of distributed ledgers in order to gain reward from the system. The

permissionless blockchains are to encourage more and more nodes or miners joining the blockchain network so that more miners can verify, append and maintain the transactions that are involved in the blocks, and also more decentralised degree is conducted in the blockchain system. In blockchain systems, miners who are elected as block producers can claim reward when they generate blocks successfully. Thus, an election scheme in consensus protocols is critical for blockchain systems. Different blockchain systems might have different criteria for electing the appropriate candidates or leader as block producer.

Consensus protocol is the core property in the blockchain system where a valid block is agreed upon to be appended to the chain. For the blockchain consensus instances present properties that define what guarantees are in place and how the decision is made. The most important thing in consensus protocol is providing candidateship election and determining a committee of nodes to participate in the blockchain system, whose primary responsibility is to propose a new block, or validate transactions in the blocks and provide a final decision for appending the new correct block at chain. Therefore, consensus approaches can be considered as candidateship election algorithms deployed in blockchain systems.

In blockchain systems, blocks are jointly determined by multiple participants (Miners or Nodes). For example, in Honey Badger [19], typically a block is proposed by a selected node. The node who has been selected and produced the block that can receive rewards from the system. Some nodes also can receive rewards by participating in a consensus group as committee members. The different blockchains have different consensus approaches for candidateship election schemes. Some of them are based computational power, such as Proof of Work (PoW) [20]. Some of other are based on stake, such as Proof of Stake (PoS) [21].

Therefore, the innovation and analysis of these complex blockchain systems requires a broad efforts on consensus approaches. The thesis is developed on blockchain based consensus approaches.

1.3 Research Motivations

There are many research motivations for exploring the blockchain technologies. However, the blockchain systems also have many limitations based on consensus approaches.

In proof-based consensus approaches, such as, Proof of Work (PoW) requires a large amount of computation to solve puzzles, and also miners invest hardware and spend electricity to participate in the mining process in order to claim reward from the system to cover the cost they paid. However, miners who do not have good hardware still spend time and electricity to mining and maintain current blocks. Thus, in this way it is not advantageous to some miners with poor equipment. Furthermore, the small subset of ming pools controls the majority of mining power. Also, the time to finalise a block that is taking too long and transaction throughput is low in Bitcoin. It also affects scalability and efficiency of blockchain systems, if blocks are generated more and more in the future for upcoming transactions. Proof of Stake (PoS) consensus is designed to address energy waste caused by mining process in PoW. However, PoS allows the miners who hold the "wealthy" asset to have full controlling on the block generation. Therefore, it causes that blockchain systems have more security vulnerabilities which affect data security and performance. For example, "Nothing at Stake" is a well-known attack on PoS systems [22]. Also, the election scheme is disadvantageous for others who do not have much asset (Tokens or Coins) as their stake, but they still need to spend electricity and time to maintain and store current blocks in the blockchain network.

In vote-based consensus approaches, such as, Byzantine fault tolerance (BFT) based consensus it is evident that executing the votes for reaching an agreement. Participants join BFT network with restriction due to complicated degree of exchanging messages, and it can be impossible to make the trust assumption if nodes join BFT network freely. Thus, it is hard to provide an open environment for miners joining and participating. Also, performance is dropping and getting lower and lower, if nodes are joined more and more in the blockchain network.

In the contemporary research field, there is much literature describing the proof-based consensus in permissionless blockchains and vote-based consensus in permissioned blockchains. The primary research gaps are identified as follows:

Very limited research works have been dedicated on how to provide a comprehensive election scheme in consensus protocol for nodes participating in blockchain networks, so that more and more nodes can freely join, and get more chances to be elected as candidates or leaders for block generation.

For proof-based consensus approaches, miners participate in the blockchain network, and showing enough eligibility can have the right to append a new block to the chain.

However, the most of them employ a token or coin as single criteria for electing block producers. The election scheme in consensus is discouraging the "poor" participants and allowing "rich" stakeholders to have full control over the generation of blocks. Thus, "poor" participants find it hard to get elected or compete with others. For long-term consideration, nodes will loss motivation to join or maintain the blocks as one of ledgers in the blockchain network.

For vote-based consensus approaches, especially BFT-based consensus is often conducted in private and consortium Blockchain, in which the decentralization degree is lower than in public Blockchain with proof-based consensus. Thus these blockchain systems have limited scalability due to message complexity and the types of communication patterns in the protocols. Practical Byzantine Fault Tolerance (PBFT) [23] is used in Hyperledger blockchain [24, 25] and required several rounds of communication with all-to-all message passing involving all consensus members. Thus, in practice, PBFT can only scale to a few dozen nodes. Also, there is less freedom for node joining and leaving any time.

In summary, current blockchain consensus approaches have some of the following limitations:

- The miming environment that provided in PoW based blockchains is advantageous to miners with advanced computing hardware, but it is disadvantages for nodes with poor condition hardware to do mining processes. Also, transaction per second is poor in the platform that causes lower efficiency for trade. The majority mining power is controlled by a small subset of ming pools.
- PoS based blockchains are increased potential of centralization and the concerns of governance, and vulnerabilities on data security. It makes "wealthy" nodes have full control over the block generation process. Also, single criteria for election candidate or leader in consensus approach. Election process is predictable, which brings vulnerabilities to guessing attacks [26].
- BFT based blockchains are less openness, flexibility and limited scalability. More centralised degree is conducted.

To overcome some limitations in existing blockchain consensus systems, the following research questions have been identified in this study.

1.4 Research Questions

Research Question 1: How to design a consensus approach to address single election criteria in the election process, and mitigate guessing attack.

- Sub-Research Question 1.1: How to combine trustworthiness and performance as comprehensive sore of consideration for candidateship election.
- Sub-Research Question 1.2: How to achieve higher randomness of leader election in candidateship election process.

Research Question 2: How to develop an approach to enhance blockchain performance in transaction throughput per second.

- Sub-Research Question 2.1: How to develop a method to achieve the mining efficiency for participants or nodes.
- Sub-Research Question 2.2: How to design an approach to avoid forks, and interact with BFT-style's approaches.

1.5 Research Methodology

In this study, the research presents an interactive process, including seven major steps. The research methodology leveraged in this thesis is demonstrated in Figure 1.4, adapted from the Scientific Method as an Ongoing Process [28]. The first step is reviewing literature on related research work. Next step is identifying research gaps. Subsequently, I raised the research questions based on research gaps. The next step, a novel protocol is proposed and developed to tackle the research issues. After that, it is to deploy and conduct experiments by using comprehensive consensus mechanisms. Then, collecting experimental results and making observations on results in order to explore the insights and evaluate the performance. If the outcome is tackling research gaps and satisfying, the effective and robust consensus approaches are finalised. Otherwise, comprehensive consensus mechanism is considered to be refined or parameters need to be adjusted, then deploy experiments again. The final step is writing research reports and articles.

Figure 1.4: The Research Methodology Leveraged in This Thesis [27]

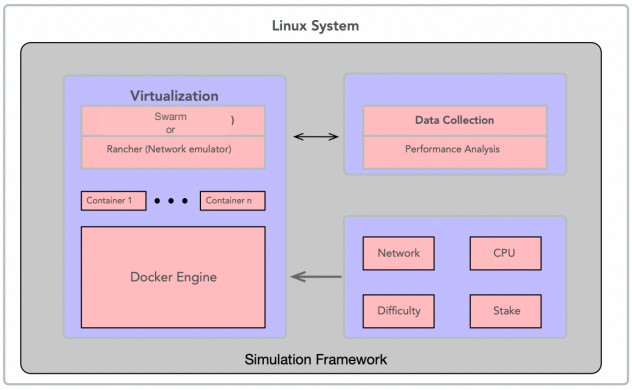


Figure 1.5: Simulation Framework

1.5.1 Evaluation and Validation

It is hard to evaluate a blockchain itself based on traditional metric, such as multiple candidates election and security model in blockchain network. This is because most simulators produce speed of transaction throughput rather than performing particular values. However, the validation of the model can be adapted when it is applied to specific problems, such as size of network. It has significant restrictions on other factors such as confirmation time for election of a candidate.

Local Blockchain Network Simulator [29] and Matlab R2021a [30] are able to simulate the blockchain operation environments and performance evaluation by using lightweight virtualisation. Caliper benchmark [31] is also available to apply and interact with our simulation environment. There are a few approaches to regulate the blockchain distributed nodes or participants, including the size of network and block structure. However, there is no unified standard to evaluate different types of blockchain consensus methods. Thus, we evaluate and validate our proposed approach by considering two major modules: configuration module and virtualisation module.

Virtualisation Module

Virtual participants: participants or nodes are operating in virtual machine by using Docker. Docker container offers logical isolation, lightweight virtualisation, and portability. Each container is able to run the core of the proposed algorithm. Docker swarm or Rancher are used to create and manage a connection for communication between the different network or physical machines. Thus, evaluating a large scale size

of blockchain network is possible by employing virtualisation modules in Rancher or Docker swarm. The architecture of virtualisation-based simulation framework presents in Figure 1.5.

Configuration Module

Configuration module is employed to allow tuning the evaluation environment to assist accurate observations. The major configuration parameters that focus on the network condition, size of committee members and amount of stake.

- Network configuration: The underlying topology in the blockchains and its network metrics is a critical role in deciding performance of blockchain. The framework enables to customise the network topology by the number of nodes or participants required in hosted single physical machines or multiple machines.
- Election candidates or leader size configuration: Size of committee members provides scale of reaching consensus from autonomous areas. Committee members can be specific about the number of areas in the entire blockchain network. This is an important parameter to allow study on how committee members affect the performance and security of the entire network.
- Amount of stake configuration: Amount of stake is the pivot of a PoS network, it controls its probability for election candidates. In a simulation environment with adjustable amounts of stake, the probability for election candidates must be well configured to fit the current environment and adjust it to compare with a pure PoS system.

1.5.2 Evaluation Metrics

The different metrics and constraints that should be considered and evaluated by the simulation environment. For proof-based networks (e.g. PoW, PoS, DPoS), there are four traditional metrics that can also be considered to evaluate performance in Proof of SecureStake. Traditional evaluation metrics are following [32, 33, 34]:

- The Transactions Throughput (TPS): Transaction throughput is the transaction rate per second, which is committed by the blockchain.

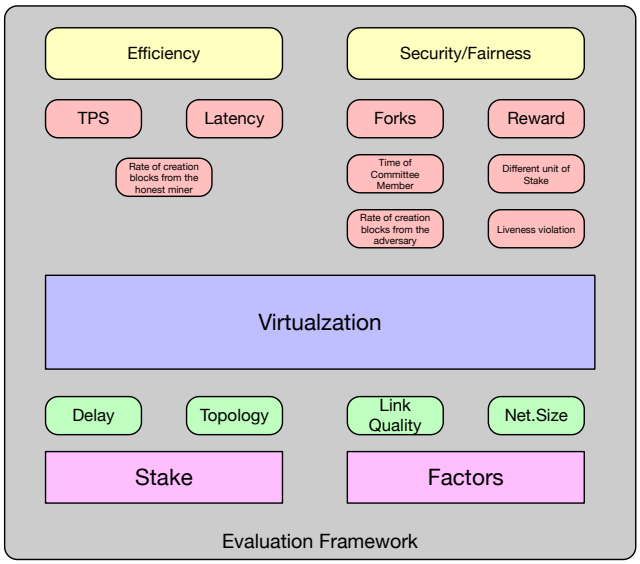


Figure 1.6: Evaluation Framework

- Network Latency: Network latency is the time required to produce the next block. In other words, it is the total time a user has to wait after processing a transaction.
- Number of Forks: Forks are essentially a split in the blockchain network. In other words, a fork can be defined as when there is a blockchain diverges into two potential paths forward.

In our experiments, To compare Transaction Throughput for Per Second (TPS) under the PoW, PoS, PoAssu, and Raft consensus approaches, we define the same transaction size and block size for them. For example, TPS value of PoW approach is tested by counting block data in Ethereum and Bitcoin. Detailed of transaction data are counted by using the RPC interface to get blockchain information. TPS average is calculated by The number of transactions in each block and newly generated blocks.

The evaluation metrics in this thesis are not restricted to these four traditional evaluation metrics. Other evaluation metrics are defined and considered according to the needs from extended approaches and problems. These following additional evaluation metrics can be defined and considered:

- Times of Candidate Election: It is elected frequency for miners elected as candidate for each round.

- Times of Leader Election: It is elected frequency for miners elected as leader for each round.
- Leader Election Entropy: It is measure of disorder for leader election process, the higher the entropy the greater the disorder, otherwise the selection will be deterministic.

These traditional and new defined metrics are evaluated the security and efficiency in blockchain systems. The experiment details are given in the experiment section of each chapter.

1.6 Contributions of the Thesis

In this thesis, I have proposed two novel blockchain consensus approaches, focus on permissionless blockchain systems. The contributions of this research work are summarised as follows:

- The multiple criteria (trustworthiness and performance factors) are considered into election scheme as the comprehensive score in electing process.
- The vulnerability on security such as guessing attack can be mitigated, and process of candidateship election is hard to predict by adversary.
- Performance can be enhanced in transaction throughput for per second.

1.7 Thesis Structures

The remainder of the thesis is constructed as follows:

- Chapter 2 reviews and analyses some of the most commonly used consensus approaches in blockchain systems.
- Chapter 3 introduces the idea of Proof of SecureStake (PoSS), a novel consensus method which can provide an election scheme taking trustworthiness and performance as multiple election criteria as in staking nodes. Furthermore, PoSS mitigates guessing attack in leader election process.

- Chapter 4 introduces the idea of Proof of Assurance (PoAssu), which can provide an Assurance scheme enhancing performance in transaction throughput per second.
- The conclusion and future work of the thesis os presented in Chapter 5.

Chapter 2

Literature Review

In this chapter, I review and provide unique insights and landscape for the most commonly used consensus approaches in blockchain systems. Then, I deconstruct the blockchain into two core components: candidateship election and consensus mechanism, so that the complexity of blockchain systems can be explained in that sample way. Furthermore, I further explore the contemporary research works associated with the process of candidateship election. Subsequently, I expand the scope to the extended security concerns and problems.

2.1 Candidateship Election

The primary purpose of candidateship election in blockchain systems is to determine a leader or committee as participants in consensus. The leaders or committee are to repressible validate, propose, and provide a decision for appending block on the chain [35]. Figure 2.1 shows an overview of the blockchain landscape. The current candidateship election can be considered into two types: vote-based and proof-based approaches.

2.1.1 Proof-based Approaches

In the proof-based approaches, nodes need to demonstrate that they have performed sufficient proof to obtain the right for block proposal work. The process of the grant this right can be considered as a candidateship election.

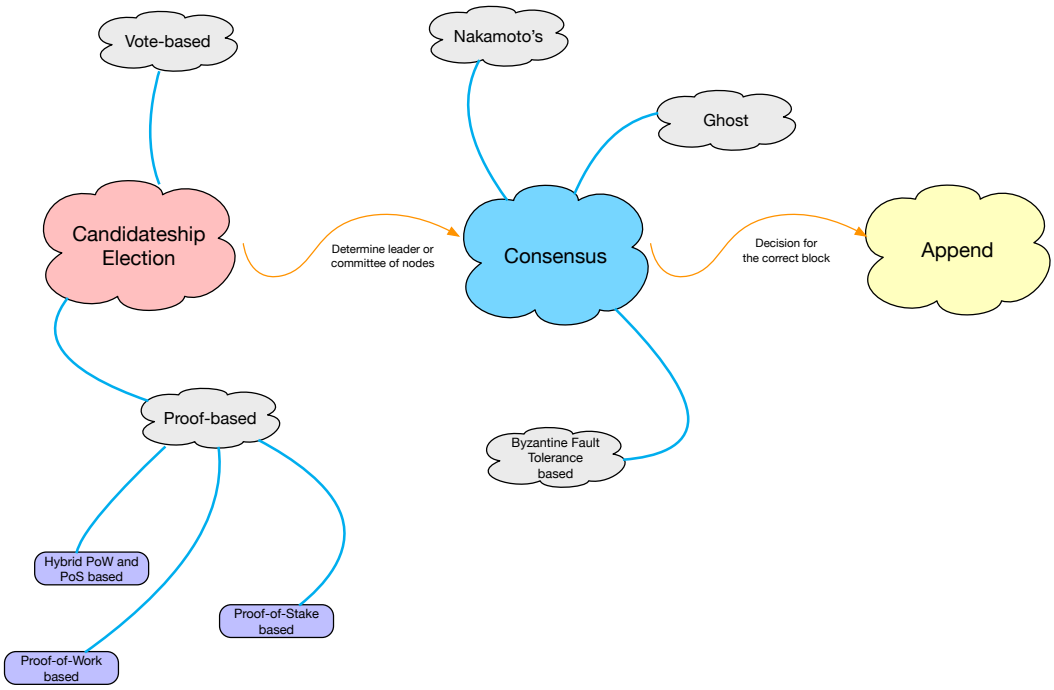


Figure 2.1: Overview of Blockchain Landscape

Proof of Work

Original proof of work in Bitcoin system, incentive mechanism promotes miners to participate blockchain network as one of the distributed ledgers to maintain blocks that appended on the chain. Only one miner can receive a reward for each newly created block when the miner has been selected as block producer according to the candidateship election scheme.

The design in the Bitcoin system was heavily influenced by Adam Back's Hash-cash [36]. Nodes or Miners need to solve the crypto puzzle by finding specific nonce in order to propose blocks. The pseudorandom function is applied to the Bitcoin proof of work for an Sha-256 hash, and input is the combination of nonce and new block hash. The speed to calculate specify nonce due to the speed of the hardware that miners provided by themselves [37].

Moreover, the key of finding specify nonce is computing power. Therefore, CPU-bound PoW based blockchains are heavily relied on CPU power to calculate a puzzle by searching specific nonce. Invariants based on PoW, Dwork et al. [38] researched the concept of memory-bound PoW in order to prevent email spam. The memory-bound PoW consensuses [39, 40] are suitable in blockchain systems. In particular,

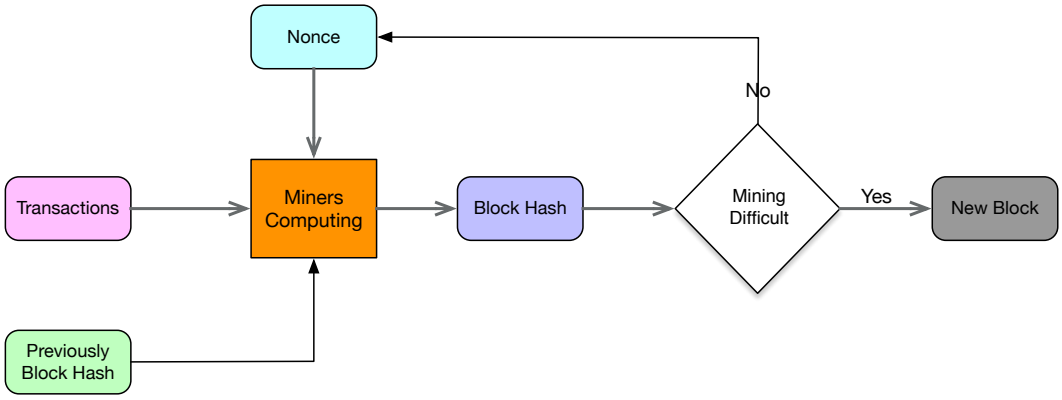


Figure 2.2: Overview of Work Flow for Proof of Work

the memory-bound PoW depends on random access in memory rather than CPU power. The CryptoNote [41] and ZeroCash [42] both apply the memory-bound PoW consensuses in the system platform.

Solving the puzzle is the priority for miners whether CPU-bound or memory-bound PoW in blockchain systems. The miner who solves the puzzle will be selected as "winner", the "winner" proposes a new block in the bitcoin network. To confirm the new block on the chain, at least six validations are required to append on the chain. To be specific, all the miners are required to process verified transactions before solving this puzzle, including other information such as $Timestamp$ and Pre_Hash into a block. After that, miners keep trying this puzzle, putting the result into the block. All the information are combined inputted to an SHA-256 hash function and inside the block header [43].

In PoW based consensus approaches, the longest chain rule is another core component of consensus mechanism. The longest chain rule is that miners must choose a single chain with highest block number. This longest chain is canonical chain in the blockchain network to solve forked state, if there are same event that has been proposed more than once at same height.

As mentioned, mining difficulty depends on a given threshold T, the secret value can be accepted when the output is lower than threshold T. Otherwise, the nodes keep making another guess of the secret value until any one of them gets the answer. Figure 2.2 shows an overview of the workflow for Proof of Work.

In a blockchain network, the PoW requires nodes to solve a crypto puzzle which is adjusted by difficulty. In order to gain the right for appending a block on the chain,

blockchain system has an agreement between all nodes for proposing the block. While there is confusion caused if every miner is broadcasting their blocks containing the verified transactions. For instance, who will then put the transaction into their block, and broadcast it to network if we only consider verified transactions. Therefore, the first miner that found solution for the crypto puzzle will get the right to proposal the block. Apparently, the candidateship is meaningless if this transaction could be duplicated in different blocks. Therefore, PoW utilises a process of solving the crypto puzzle to express candidateship election among all the miners.

Process of candidateship election can be presented as mining process in our understanding. Because of the usage of SHA-256, finding that value is extremely difficult, which makes all miners can have a chance to try many times to find the answer unless one of them is lucky enough. The mining environment provides conditions of openness and public competition. Everyone can join to participant solving the puzzle in this environment, and the efforts paid for guessing or computing the right value. However, miners with advanced hash computing power have the advantage to solve the puzzle. The first miner that found solution for the crypto puzzle will get the right [44]. The mining difficulty level is becoming harder and harder according to increased block for every day, so miners with poor hardware it extremely hard to solve the puzzle. In that scenario, miners may not have the motivation to maintain and verify blocks as one of the distributed ledgers in the blockchain network.

The number of blocks increase for each day, which increase the difficulty level of solving puzzle. Therefore, miners who want to get right to propose a block that they have to invest more in the hardware and time to do mining. While miners who do not own modern and powerful equipment, being disadvantages on mining. Some miners invest or own modern and powerful equipments, the suitable nonce cloud be found easier. However, miners who do not have good hardware still spend time and electricity to do mining and maintain current blocks. Thus, in that way mining environment or process of candidateship election is compromised motivation by discouraging these participants.

PoW achieves the goal for candidateship election among all miners in blockchain, there is still an existing some limitations. The process of mining needs to find the solution for a crypto puzzle which requires a large amount of computation. Thus, that conducts high resource and energy consumption. Moreover, costs are increasing for a miner when the mining difficulty level rising, which affects efficiency and future

operation in the blockchain platform.

Miners find a solution for puzzles that is a long process. The long waiting period spends on the process of mining, finalising transactions and appending block, which relates directly to a shortcoming of PoW in mainstream blockchains. Therefore, transaction throughput is heavily influenced by the mining process, which heavily affects long waiting time on confirmation of a trade-in system platform.

Also, PoW based blockchain assumes that honest nodes control the majority. The adversary wants to dominate the blockchain that must control over 51% nodes among all nodes in the blockchain network. Thus, it is time and financial competition for the adversary. The more honest nodes joined the blockchain network, the domination blockchain is more difficult. That is the reason why public blockchain like Bitcoin is an incentive for more miners to join the blockchain network. In Bitcoin, the Winner-Take-All scheme [45] is a disadvantage to others who get nothing in return, these miners who do not own advanced hardware that may lose motivation to maintain and verify blocks in blockchain according to the current ming environment. As the results, degree of decentralisation will be reduced. In Kwon et al [46] study, the incentive mechanism is highlighted for the importance of decentralisation. A incentive mechanism should be designed to gradually narrow the rich-poor gap between miners with great wealth and miners with less wealth so that blockchain system can achieve a good decentralisation. In other words, goals of incentive mechanism are not only incentive more miners joining systems, but also it is to reduce the gap between the effective power of the rich and poor, not the gap between their resource power. Even if the "rich" possess significantly large resource power, the decentralisation level can still be high if the "rich" participate in the consensus protocol with only part of their resource power and so not large effective power.

A study [47] presents that incentive mechanism can attract more miners to do mining work and maintain blockchain system, however, it also can bring challenge on 51% attack in PoW system, the current incentive mechanism may encourage rational miners to launch 51% attacks based on two cases. The first case is that a miner of a stronger blockchain launches 51% attacks on a weaker blockchain, where the two blockchains share the same mining. The second case is that a miner rents mining power from cloud server provide to launch 51% attacks, As result of 51% attack can conduct double-spend attack, so miner are able to get profit from these two attacks. If double-spend attack is more profitable than mining, miners are more intended to

launch 51% attacks rather than mine honestly.

Proof of Stake

In 1998, Wei Dai introduced the first stake based voting mechanism [48], it mentions that users can vote on an included transaction based on money that staked in system for resolution. Proof of Stake (PoS) first proposal was discussed in Bitcoin community to address issues of energy-wasting caused by mining process in PoW [49]. The key concept of PoS is to require miners proving the valid assets as the stakes such as coins, instead of using hashing power to solve the crypto puzzle. In PoS, stakes can be taken from miner's deposits or current balance. Thus, the block producer will be chosen randomly from the deposited node, and the chance became block producer can be mapped to the stake's amount they have.

The basic idea of PoS is to decide who will most likely get elected for producing or proposal block generation by using their stakes. Thus, using the stake as proof has one significant benefit which is anyone who has more stakes that would be more trustworthy. She or He would not want to perform some behaviour like fraudulence or double-spend to attack system due to there is much of her or his profits on the chain. Moreover, adversaries must hold at least 51% of all stakes in the current PoS network, if they want to comprise PoS approach.

The pure PoS consensus approach can be found in the Nxt blockchain [50]. In this blockchain system, a miner with more stake holds, the higher chance she or he may have it to propose a block. To be more specific, if t is sum of the all coins from all the miners in network, and there is a miner P who holds p coins (p $<$t), so the chance of miner P to get elected for proposing a block, that is p/t. If a miner has been elected for proposing a block, the miner need to go though verifying the transactions, collecting transactions, proposing the block, and broadcasting the block to network. A similar method was introduced by Bentov et al. [51], the method is to select the miner based on the stake miners owned: miner with higher chance must have more stake a miner own so that more chance she or he would become the block appender.

Kiayias et al. [52] proposed a new approach of candidateship election. That approach based on the following the Satoshi procedure in order to execute the PoS. The candidateship election should be randomly by computing the entropy value. Also, process of computing should be secure enough to achive that no one can predict the results or manipulate the process of candidateship election.

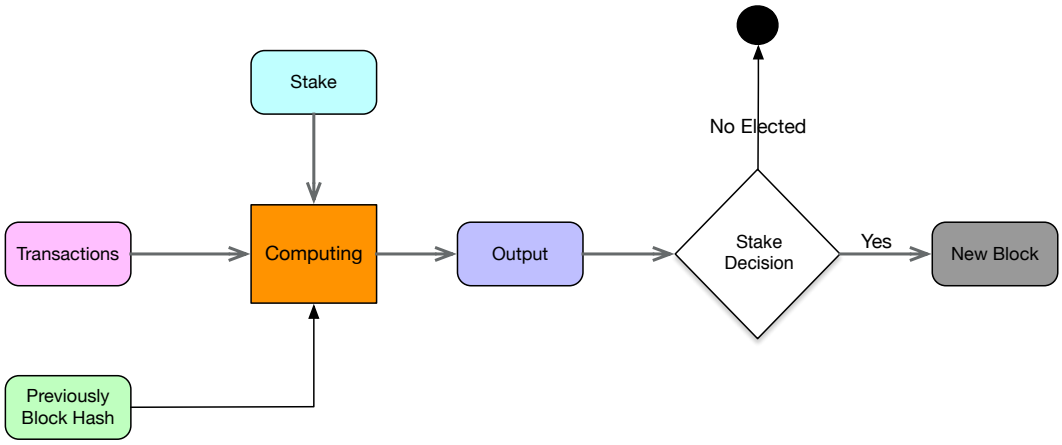


Figure 2.3: Overview of Work Flow for Proof of Stake

The new approach was proposed by Bentov et al. [51] for candidateship election which elects a miner as a leader for block appender. But the difference is Kiayias et al provide a snapshot of the stake of each stake shareholder, then a collection of stake for shareholders will be selected to approach protocol. Based on the outputs of these results, the stake shareholders and leaders executing the approach in the next round will be elected, which ensures that the process of calculation is almost impossible to simulate. Also, in that approach, the elected miner are able to create a block only, so the works like putting transactions are processed by a group of stake shareholders. This group of stake shareholder is named endorsers who are responsible for voting like committee. The rewards will be delivered to endorsers and leader equally.

In Larimer's proposal [53], it is to employ the stake as proof for voting instead of using it as probability of election for a block producer. These kind consensus approaches are also known as delegated proof of stake (DPoS). In DPoS based platform, the number of people who own stakes, and these stakeholders can be considered as two types of group. A group will have to vote for a delegation, and another group is named "witnesses" who are responsible to verify the transactions and maintain the block in the network. Moreover, the more stakes a person holds in the system, that means the more voting power will be assigned to the witness.

The witnesses are to verify the transactions, and produce the block after verifying congress is finished. Furthermore, the group of witnesses is always shuffled. The speed of creating a new block is 2 seconds per block, and produced blocks are sequential. Also, a witness will be removed from the delegation, if the witness is failed to produce block on time. The rewards always can be grant whenever any witness

produces a new block to append at a chain.

The core concept of PoS relies on nodes providing validity through depositing assets, replacing the nonce to solve the puzzle with a stake-based candidateship election. The asset can be considered as the stake which can be taken from the deposits or current balance. Also, the concept of PoS is adapted into the number of different types of candidateship election scheme. The purpose of this candidateship election is to filter the nodes/miners as committees in order to participate in the consensus and gain weight of voting power. Figure 2.3 shows an overview of the workflow for proof of stake.

PoW's candidateship election is a disadvantage for miners who do not own modern hardware [54]. Because some miners could solve the crypto puzzle easily by employing or owning modern and powerful equipment. While others with a poor condition on hardware are very difficult to be the first one to find a solution for the puzzle.

Most of the variants PoS are using a stake as a huge weight for candidateship election, which makes rich stake holders more powerful. While stakeholders with less asset would be disadvantages in the candidateship election process. In candidateship election, the rich stake holders are still an advantage in the entire process.

The PoS has a research contribution to it. PoS mitigates huge energy-consuming for PoW, however, there are several weaknesses in PoS. One of the outstanding problems is that PoS increases the potential of centralization and the concerns of governance. In the concept of PoS, the distributed assets are people willing to deposit. The stake greatly impacts the candidateship election and it could conduct to a small subset of nodes being in a position of power for the entire system operation. In order to mitigate this issue, some PoS algorithms use frequent candidateship changes and apply randomness function so that committee member can be changed, and also a larger mining pool of nodes can be used. But this way still heavily influenced by the use of incentive mechanisms [55]. PoS is satisfying the requirements of candidateship election but it is vulnerable to the number of attacks.

In addition, the Winner-Take-All scheme [45] is also applied in PoS blockchain system. As result, it is disadvantage for some stakeholders who do not hold a large number of stake to maintain PoS system. These stakeholder may lose motivations to participants blockchain system. Brunjes et al. [56] introduced Reward Sharing Schemes (RSS) to promote the fair formation of stake pools in collaborative projects that involve a large number of stakeholders such as the maintenance of PoS

blockchain. RSS are able to avoid that single stakeholder creates a large number of pools in the hopes to dominate the collaborative project.

Proof of Lock

The concept of Proof of Lock is that nodes who have deposited in themselves as validators in order to participate in censuses group [57]. Validators' voting power is equal to the amount of the deposits coins they own. Validators are posting a bond transaction in the entire blockchain network to ensure all the nodes know that validators have voting power. Also, validators are broadcasting cryptographic signature, votes, to agrees upon the next block.

The bondcoin can be unlocked by validators themselves after they post an unbending transaction. However, coins still keep locked after sent unlock message to the network, so there is a predetermined period of time called the unbinding period. Then, the validator is free to transfer or spend those coins.

A group of validators who own at least 2/3 of total voting power, is named a 2/3 majority of validators. Similarly, a group of validators who own at least 1/3 of total voting power, is named a 1/3 majority of validators.

In Tendermint platform [57], the implementation of PoS is required nodes to lock or freeze stakes in order to destroy or remove assets as a punishment, if nodes perform malicious actions. The candidateship elects the nodes who have locked coins in network. The voting weight of elected nodes are mapping to the number of stakes they locked as a ratio to the sum of total locked stakes in the blockchain network.

Proof of Deposit

The idea is similar to Proof of Lock, it requires nodes to deposit assets. A small subset of nodes who have deposited coins in network that will be randomly selected as participants in consensus group. The election scheme in Proof of deposit is different due different platforms may have different security or performance considerations. For example, only a single node will be elected when the chain-based blockchain implementation is used. Another example, a subset of nodes will be elected when BFT-style blockchain implementation is chosen.

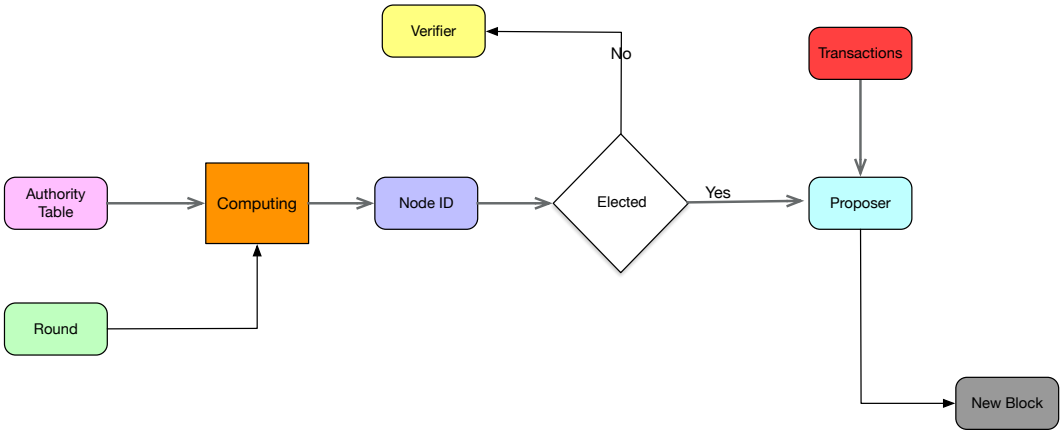


Figure 2.4: Overview of Work Flow for Proof of Authority

Proof of Authority

The concept of Proof of Authority (PoA) was introduced to add extension features to the Ethereum platform [58] on the testing network. The core of proof of authority is to elect trusted node from the blockchain network to produce new blocks.

The proposal of PoA was primary focused on maintaining blockchain with minimal waste of energy on the testing network. In proof of authority, it requires authorities who are elected to verify and produce blocks. A leader or group of the authorities is elected in order to produce the blocks which can be appended on the chain for each round. Figure 2.4 shows an overview of the workflow for Proof of Authority.

PoA provides a candidateship election in blockchains. Moreover, in PoA's candidateship election needs to pre-elected nodes as the authority committee in consensus group. In committee members, a leader is elected according to the calculation of blocks and time. Thus, the entire blockchain network is based on trusted authorities that elected to create correct blocks and append the chain.

Proof of Capacity

Proof of Capacity (PoC) also called as Proof of Space [59], using hardware to run the blockchain. The process of block production is to use a hardware storage that can be used as proof for the valid works, instead of wasting huge resources to do mining work. The process of candidateship election requires a node proving validity by retrieving and storing shards of file. The node is not only providing validity but also aiding in the blockchain operation, once the node accessed storage and deliver

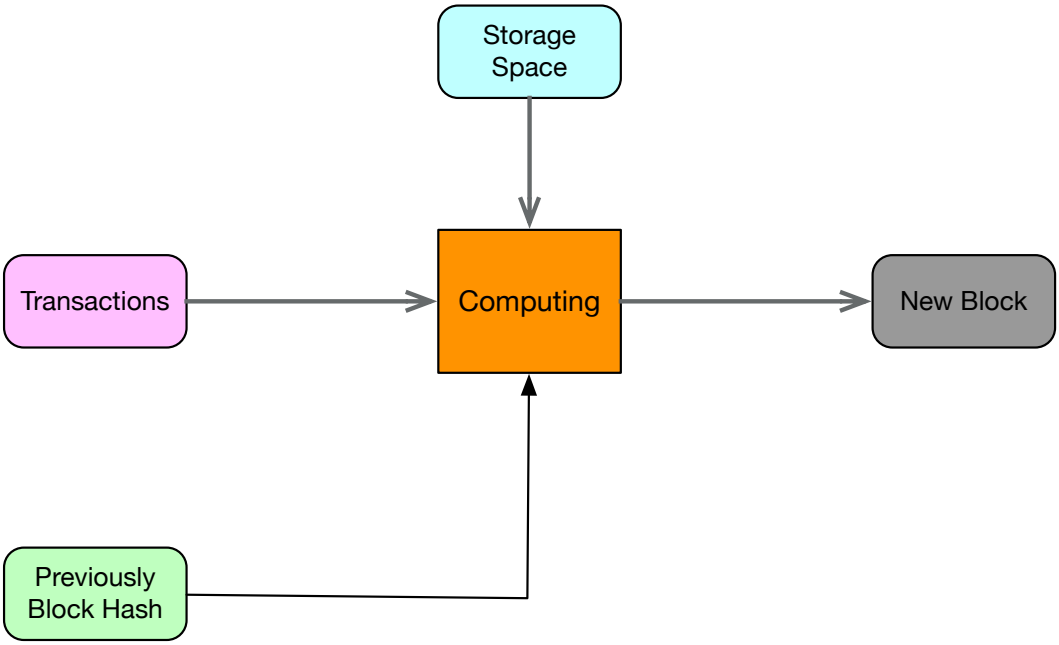


Figure 2.5: Overview of Work Flow for Proof of Capacity

the file successfully. This process allows nodes to store and share amongst the entire network.

The concept of PoC requires a node to prove validity by storing important information in order to avoid large waste of electrical resources. The main purpose of this candidateship election is that a node provides validity by storing important information. The idea of PoC is also based on PoS, goal of PoC is to mitigate the number of wasted resources on the system. Figure 2.5 shows an overview of the workflow for proof of capacity. PoC is taking stroage space as a kind of the stake for election, the node that provides storage that cloud have the power to create blocks.

Proof of Location

The mainstream blockchains using Proof of Work based candidateship election requires computationally intensive tasks in order to get right for block production. However, Proof of Location uses IoT device's dynamic geographic location to verify and create proofs of the device's location, so that the geographic location can be used for signing transactions, interacting with a smart contract for verification of location [60].

The concept of Proof of Location is to use IoT device's geographic location as proof

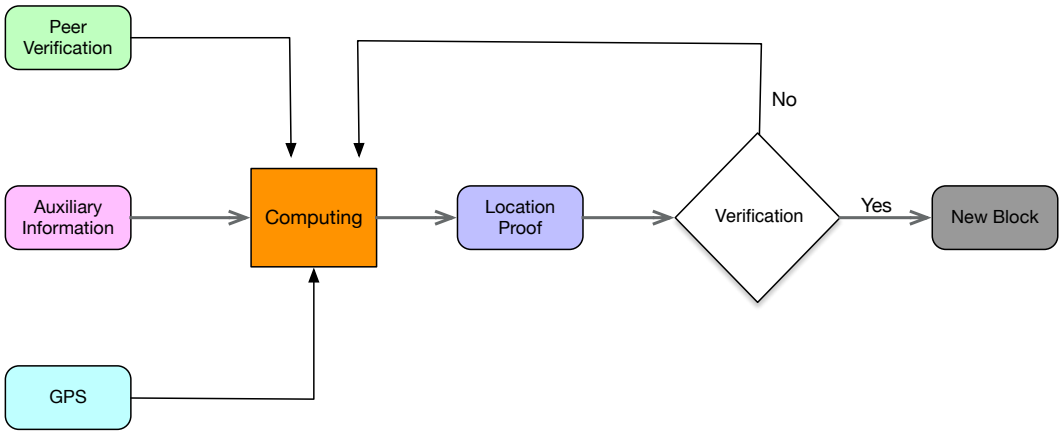


Figure 2.6: Overview of Work Flow for Proof of Location

to integrate into interaction with the blockchain. Proof of Location relies on a peer node that is closed to others so that the peer node can be signed by others to verify the dynamic location. Furthermore, there are low-range communication channels among all the nodes in order to avoid geographic location spoofing. Figure 2.6 shows an overview of the workflow for proof of location.

Proof of Location provides a candidateship election approach based on IoT device's location. To achieve that the location must be verified by a peer node which is nearby devices. Although there are countermeasures for location spoofing, the location spoofing could have probability happens by collusion amongst players as peer nodes.

Trusted Execution Environment

The concept of Trusted Execution Environment (TEE) is to provide the strong assumption which is secure space of trusted hardware so that code can be secure running in this environment. TEE is heavily impacted by mainstream hardware manufacturers such as Intel [61], ARM [62] and AMD [63]. TEE put a trusted on these manufacturers. In addition, TEE based platforms require users to trust the manufacturers of the trusted components.

A node that gets right to produce a block that can be elected by using a random function to select nodes based on trusted execution environments on hardware. TEE trusted hardware devices have a unique identification that can be used as proof, and also this identification can be used once. This identification can be used to avoid spoofing the number of hardware devices.

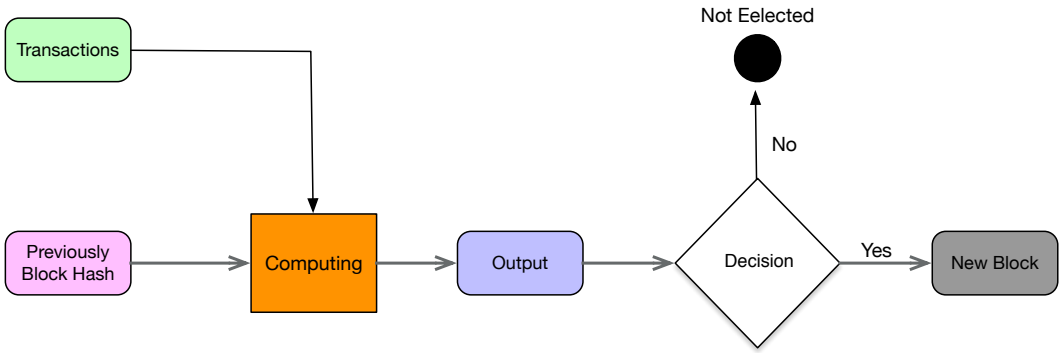


Figure 2.7: Overview of Work Flow for TEE based

Candidateship election in TEE is to select nodes with a probability proportional to the number of TEEs. It means nodes can increase the chance to be elected in order to produce block by purchasing or employing more hardware devices. Figure 2.7 shows an overview of the workflow for TEE based.

TEE based candidateship election is heavily relying on specialized hardware with unique hardware identification, and trust is based on manufactures to guarantees a secure environment. However, trusting hardware means trusting vendors that supply the hardware, which will conduct centralization by vendors.

Proof of Reputation

Proof of Reputation was introduced to address the long-term security risks brought from computational power, even when more than 50 per cent of computing power is temporarily controlled by an attacker [64]. Proof of Reputation employs reputation scores to elect members into a consensus group. Thus, every miner in Proof of Reputation has decision power, which means miners have voting power based on reputation. The reputation builds on total valid work that contributed to the blockchain system and regularity of that work performed over an entire period of time during system activity. In that way, miners can gain a reputation score from blockchain, also miner's reputation will be reduced if they deviate from system specification. Figure 2.8 shows the workflow for Proof of Reputation. Furthermore, miners who are in the consensus group may have different reputation percentage, for example, m is one of the members from the consensus group, so a percentage of m's reputation is equally $\frac{m'sreputation}{M'sreputation}$ (M is the total reputation of this consensus group).

Proof of Reputation can be summarised as two core parts:

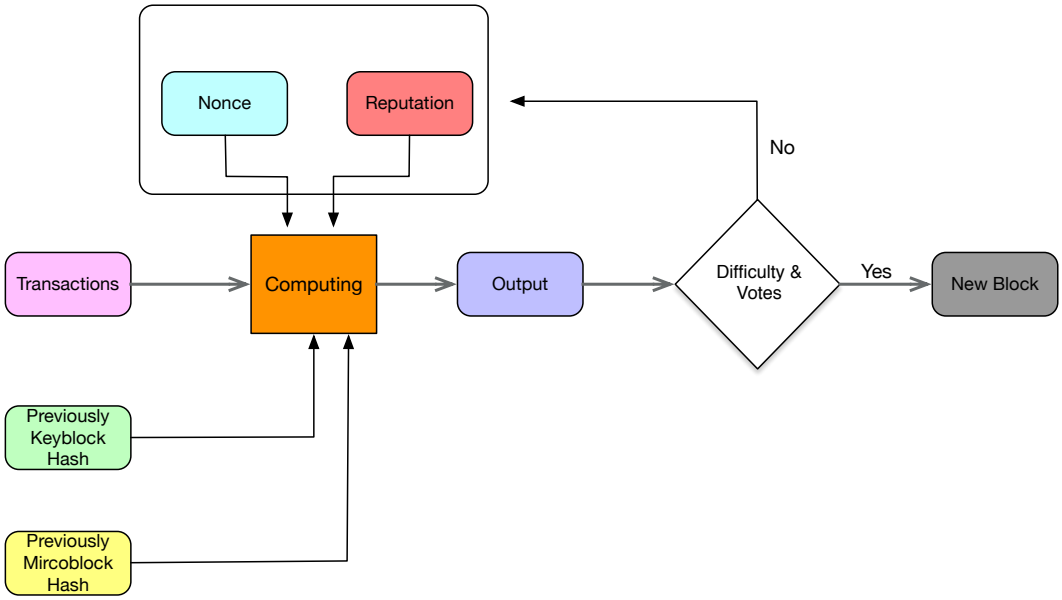


Figure 2.8: Overview of Work Flow for Proof of Reputation

- Firstly, miners who have top of reputation score will be elected as members into to consensus group.
- Secondly, miners who are in the position of consensus group will run BFT style's protocol to reach an agreement with certain votes from members.

As mentioned, reputation is employed to quantify performance and behaviour that existing in blockchain system based on historical data. Miner's reputation score will be dropped to zero if the attack is detected, which means that the adversary must rebuild his or her reputation to lunch the valid attacks again once the adversary launched his or her first attack. Thus, to conduct attacks in Proof of Reputation is costly time investments.

Although, Proof of Reputation can mitigate security risk on 51% attack, however, the adversary may be penitently honest behaviour for a long time in order to gain more reputation score, then single or collusion adversary can launch valid attacks once which make system lose safely and liveness when single or collusion adversary has enough miners who gained high reputation score in system.

Proof of QoS

Proof of QoS (PoQ) [65] was proposed to provide fairer and energy-saving environment for selection participants based on score of QoS. The score of QoS is combined

four different factors:

- Activity rate: The times of node i has been nominated that divides by the total number of block generation attempts.
- Error rate: The number of times that node i failed to generate a block that divides by the total number of attempts that made by this node to generate blocks.
- Deposit ratio: The amount of a node i that deposited in system that divides by the sum of total amount of all nodes deposited in system.
- Reference factor: This factor reflects the node that refers to another node. If the suggested nodes's error rate is lower than pre-defined threshold, this factor will be increased by a given value in the block.

Based on above factors, QoS selection scheme are able to combine all the factors together as single stake score of a miner as a miner's performance. The probability to get elected as block producer is heavily impacted by individual's QoS score.

2.1.2 Vote-based Approaches

In vote-based consensus approaches, nodes will exchange messages with others to agree on the blocks or transactions to be appended to the ledger.

HotStuff

HotStuff [66] is a leader based Byzantine Fault-Tolerant protocol and innovated from PBFT and Tendermint. HotStuff has made the change from mesh network to star network. In HotStuff, communication relies on a leader node for each time with a star network instead of using a P2P network. The leader node sends a message to other nodes after the leader handled the message. Thus the cost of communication is reduced by the star network. HotStuff has three core phases which are showing in Figure 2.9. These three phases can be summarised as follows:

- Phase 1 (Pre-Commit): Other nodes vote for prepare message, leader node will broadcast pre-commit message to all nodes when leader received several votes that are more than N - f votes (N is the number of nodes and f is the number

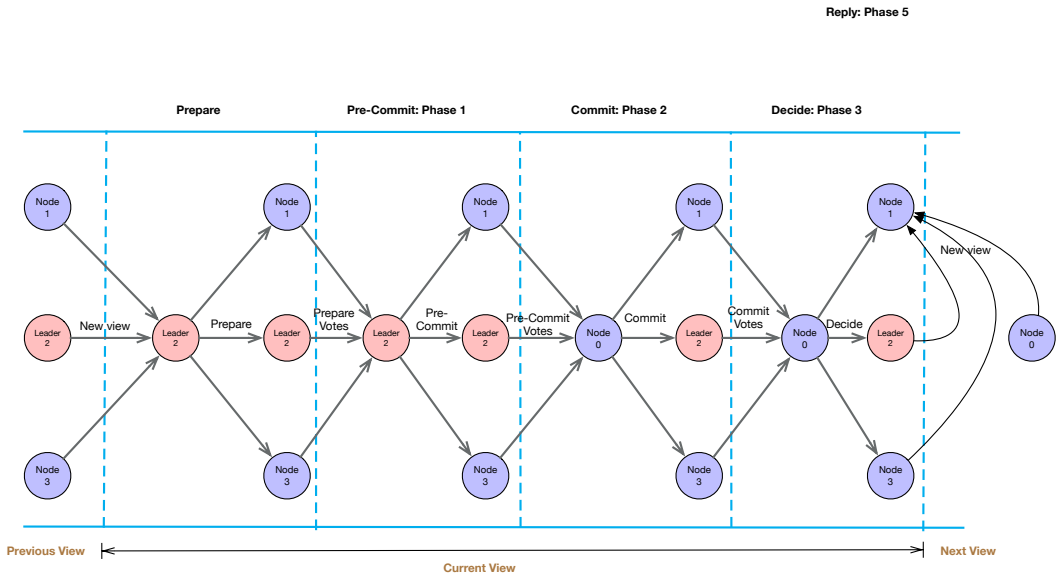


Figure 2.9: Work Flow for HotStuff Three Phases [66]

Byzantine faulty nodes in consensus group). Then other nodes will know that process of status can be moving into the next stage now.

- Phase 2 (Commit): Other nodes vote for the pre-commit message, the leader will broadcast the commit message to all when the leader collected enough votes. Then, nodes who received the commit message will be locked current status to achieve the agreement.

- Phase 3 (Decide): Other nodes vote for a commit message, the leader will broadcast decide message to all when the leader collected enough votes. Nodes who received decide message that will execute it and move into the next stage with a new view.

In HotStuff, there is no requirement for any randomisation. The concept of HotStuff presents three core phases to achieve consensus with partial synchrony.

Algorand

Algorand [67] is a new cryptocurrency system which can enhance TPS and avoid forks. Algorand based on pure PoS, and adopted with Byzantine Fault-Tolerant (BFT) to commit a block. In Algorand system, there are two majority processes to commit a block. The fist process is to randomly select small subset of verifiers from

all miners. The second process is to reach an agreement by running BFT among all verifiers. Once verifiers receive 1/3 votes from other verifiers, then more than half of verifiers use own private key to sign signature in the block, finally, the signed block will be broadcasted to Algorand network. For each round, verifiers are able to randomly change, and all the current verifiers do not know who are participating in consensus group for next round. The process of election verifiers is similar to Single Secret Leader Election (SSLE) [68] scheme. The difference of SSLE is to aims randomly choose exactly one leader from the group with the restriction that the identity of the leader will be known to the chosen leader and nobody else. However, the result of election can be release to public at a later time for revealing his/her identity and proving that she has won the election.

Hyperledger Sawtooth

Castro and Liskov proposed Practical Byzantine Fault Tolerance (PBFT) based on Byzantine fault tolerance [69]. Then, the Hyperledger Sawtooth platform (version 1.2) adapted PBFT into operation. Candidateship election can be found from Hyperledger Sawtooth platform, it named as Membership Service. However, this membership service is to pre-define roles for a set of nodes in-network, and also provide a set of authentication services to the third party who has permission to join.

In the Hyperledger Sawtooth platform (version 1.2) [70], all the peers are the same based on network configuration. Peer node can have multiple roles in the network. There are four types of peer nodes defined in Hyperledger Fabric 2.0 platform:

- Committing peer: every peer in the communication channel can be a committed peer so that they can receive a new produced block and add copy into the ledger as an append operation.
- Endorsing peer: The peers with smart contract can be endorsing peers so that all the transactions can be signed and recorded by endorsing peer.
- Leader peer: There is a leader peer defined when the platform has multiple peers in the channel. A leader peer has the repressibility to distribute transactions from the orderer to committing peers.
- Anchor peer: Anchor peer is to define a communication channel for peer's communication between another parity who involved in this permissioned blockchain.

2.2 Consensus Mechanism

The consensus mechanisms in the blockchain have a repressibility for determination next proposed block in order to append on the chain. Determination next chosen block is a critical step to the operation of the blockchain, which contains the transactions and timestamp. In the distributed environment, Byzantine Fault-Tolerant (BFT) protocols are proposed to address agreement among all nodes due to there may have potentially malicious in a distributed network. This problem is also well known as the Byzantine Generals Problem [71].

Byzantine Generals Problem

In 1982, Lamport proposed Byzantine Fault Tolerance in order to address the problem of a reliable computer system caused by the failure of one or more of its components. Lamport thinks a failure component may exhibit a type of behaviour that is sending conflict information to different parts of the system. Thus, to ensure consistency and correctness information passed by each component is key to achieve a reliable computer system. The problem of coping with this type of failure can be described as the Byzantine Generals Problem.

There are serval Byzantine arms camped outside of the enemy city, each division commanded by its own general. When all Byzantine arms attack this enemy city, then the enemy city will be destroyed. Otherwise, the enemy city will destroy each of them, if they start to attack at different time.

So, Let's assume that there is an amount of 2N+1 generals in total, and one of them is a traitor. There is a mission of attack or retreat which needs to communicate with each other, then the mission can be operated by all of them. Let's assume that there are half of the loyal generals want to attack, and half of them want to retreat. The traitor now can misguide the message to a set of generals to retreat or attack. After a traitor misguided set of generals, the situation will be conducted that amount of N generals operated mission on the attack, amount of N generals operated mission on retreat, the final result would be fatal for the Byzantine army. Thus, this problem presents a worse result when communication has a conflicting message and participates cannot achieve an agreement.

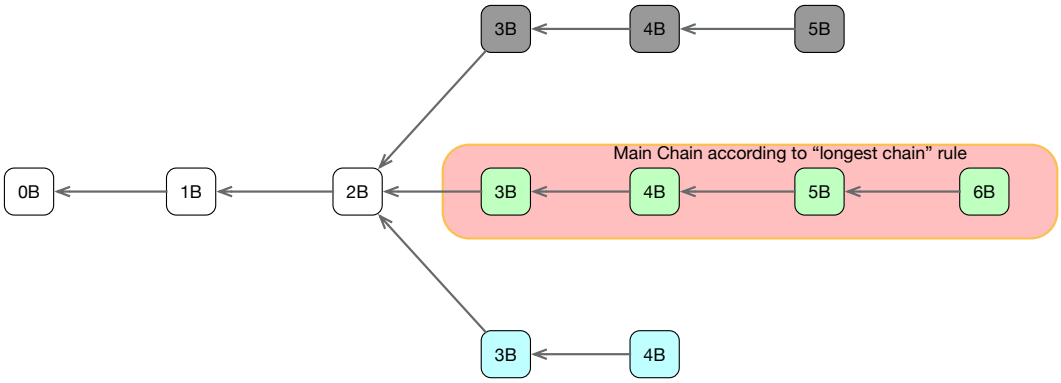


Figure 2.10: Longest Chain on Nakamoto's consensus

2.2.1 Nakamoto's Consensus

As mentioned, Bitcoin is the first application for permissionless blockchain, and Nakamoto's consensus is introduced by Bitcoin. Then, Nakamoto's consensus is adopted into many other blockchain platforms [72, 7]. The longest chain rule is core of concept of Nakamoto's consensus to achieve an agreement among all nodes. The longest chain means that a single chain with highest block number (canonical chain) to solve forked state, when there are more than one chain in the network. For example, there are 2 chains at same height, the miners will keep adding new blocks on the first valid block, however, miners will reject conflicting blocks when current branch chain is not the longest chain. Figure 2.10 shows how Nakamoto's consensus handles the forking problem. The core aspect of the longest chain rule is to trust most works that performed before and use this principle in the blockchain network. Blockchain is an operation in Nakamoto's consensus with majority honest nodes.

In the permissionless blockchains, Nakamoto's consensus provides a view of consistency for blocks [73]. It means that all miners will eventually choose the latest block with the highest height of the chain. Nakamoto's consensus solves the forked state problem and eventually reach the same view among all miners.

Moreover, Nakamoto's consensus guarantees security based on assumption that the majority of nodes is honest. Also, the speed of block generation is high with the number of propagated blocks in-network, blockchain must wait that there is no branched chain happened, then confirmation of appending new block is successful. However, the process of choosing the longest chain could conduct a number of low

performance and potential attacks.

Although the longest chain rule provides a solution for the forked state, it causes that the number of block's proposals are wasted. PoW is to select the first node that solves the puzzle, which is a waste of mining power if we consider candidateship election in its approach. Nakamoto's consensus is to elect the blocks on the longest chain, which is also a waste of mining power. Because the number of propagated blocks are rejected if the blocks are not selected on the main chain.

The outstanding issues for Nakamoto's consensus are that more conflicting blocks will be conducted and waste of mining power if the speed of the creation block exceeds the speed of propagation time in the network. Another issue is that Nakamoto's consensus is required majority voting power which are honest nodes in the network. It means that the number of honest nodes must be exceeded 51% of the total number of nodes in order to against attack like double-spending.

2.2.2 Ghost

Nakamoto's consensus uses the longest chain rule to guarantee security, and the speed of creation block and the size of the block are fixed. This result leads to a number of block's proposals are wasted, poor transaction throughput, a long waiting period for block confirmation. Then, most researchers tried to increase the size of the block in order to reduce the time of the creation block and improve transaction throughput. However, that result leads to the following problems:

- Keep conducting forked branches: Forked state means the degree of security will be reduced and vulnerable for attacks.
- Mining reward is a delay due to network latency: Reward from the entire blockchain network is not only related to mining power, node with lower network latency is more like to get the reward.
- It is vulnerable to Selfish Mining: Malicious nodes produced a block but they do not announce to the network until the main chain has been found.

For example, in Figure 2.11 forked branch is started from block 0, 1A and 1B are forked and speed of the creation block exceeds the speed of propagated block in-network, the blockchain could be heavily presented forked branches. In that period

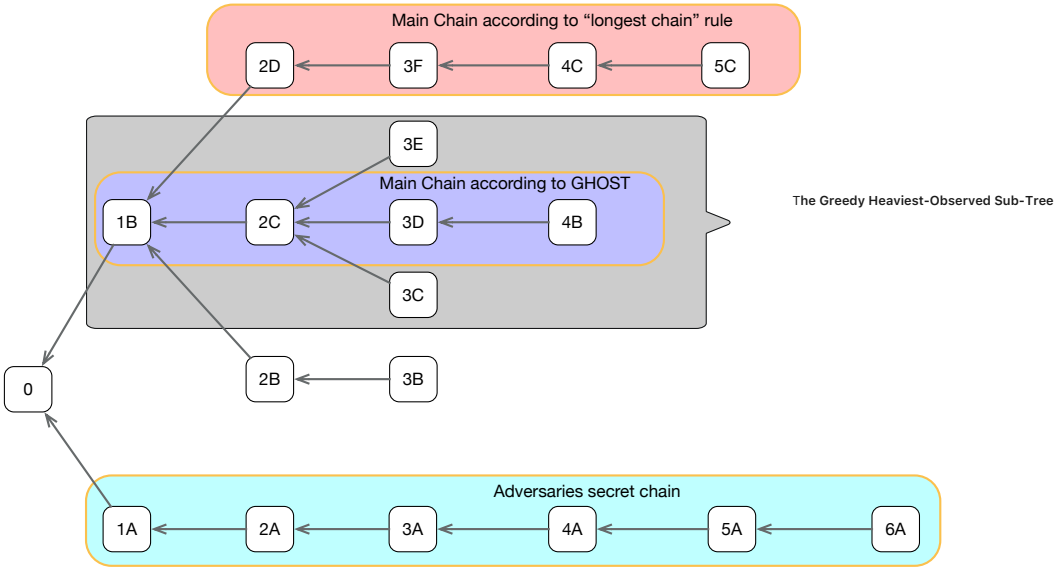


Figure 2.11: Main Chain on GHOST [74]

of time, adversaries created a secret chain with the longest length (1A-2A-3A-4A-5A-6A).

Blockchain was heavily impacted by these outstanding issues until The Greedy Heaviest-Observed Sub-Tree (GHOST) is proposed [74]. In GHOST, the idea is improved from the longest chain rule, so choosing the greedy heaviest-observed sub-tree for each forked branch. For example, in Figure 2.9 forked branch is started from block 0, 1B and 1C are forked. Sub-tree of 1A is doing selfish mining, then it mined 1A to 6A in total 6 blocks. The Sub-tree of 1B has 12 blocks, apparently, 12 blocks are more than 6 blocks. Thus, the Sub-tree of 1B will be elected as the main chain, so that it mitigates problems from forked branches, and also it makes the main chain keeps extension. GHOST mechanism will keep executing Greedy Heaviest-Observed Sub-Tree for each forked branch until it ensures there is main found. The main chain will be 0-1B-2C-3D-4B in the GHOST.

2.2.3 Byzantine Fault Tolerant (BFT) Consensus

Byzantine Problem has been provided with a solution by using signatures so that the absence of messages can be able to detect. BFT consensus has defined the following requirements to ensure an agreement [75]:

- Termination: every non-faculty node determines an output

- Agreement: every non-faculty node eventually determines the same output
- Validity: It is valid if every node has the same input
- Integrity: every non-faculty node's decision and consensus value must be proposed by some non-faculty nodes.

In the blockchain, there may have a number of nodes acting malicious, BFT consensus can be adapted to achieve consensus with a certain condition. In blockchain network should be satisfied: $N \geq 3f + 1$ where f is a number of malicious or faculty node, N is a total number of nodes. The reason is mentioned above as Byzantine Generals Problem.

The traditional BFT protocols should not be directly used in blockchains due to set of participants in the consensus is not fixed or predefined in permissionless blockchain, and also participants in permissionless blockchain can leave and join anytime they want. The size of the set of participants in the consensus is bigger, the complexity of messages is higher.

2.2.4 Practical Byzantine Fault Tolerance (PBFT)

PBFT [76] is proposed, and improved from traditional BFT protocol, so that complexity of messages is reduced from index class level to polynomial level. Furthermore, PBFT reaches consensus by leader-based communication with five phases. Figure 2.12 shows the PBFT communication phase.

Leader-based communication with five phases in PBFT can be summarised as follows:

- The first phase (Request): It is to broadcast the client's request message to the leader node.
- The second phase (Pre-Prepare): The leader node sends a pre-prepared message with signature and unique serial number N to other nodes.
- The third phase (Prepare): All the other replica nodes send a message which contains the current view, message serial number and digest message to each other when all the nodes received a pre-prepared message from the leader. If nodes received a number of received pre-prepare messages which is more than

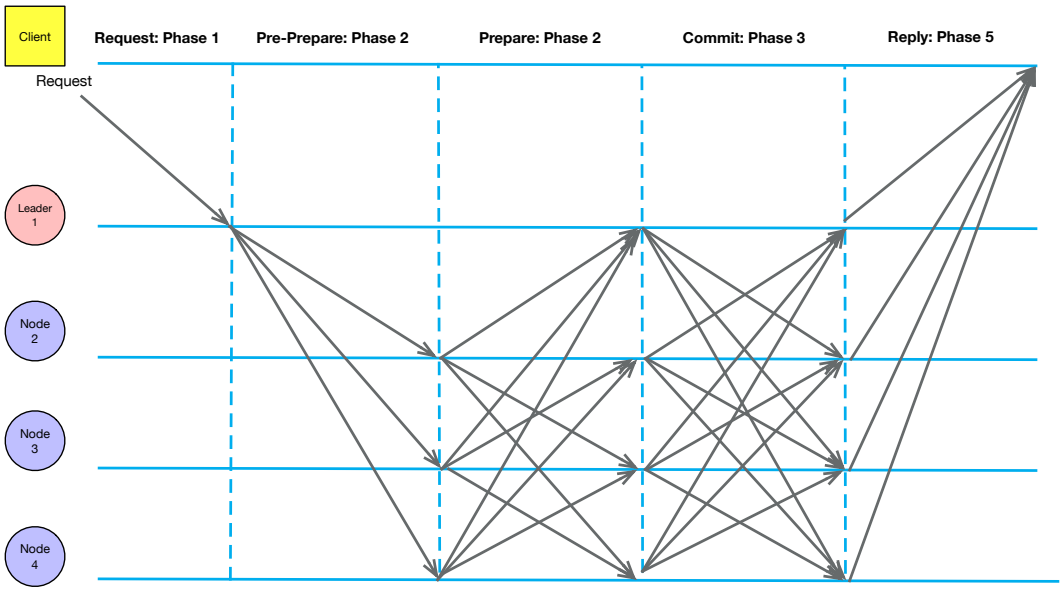


Figure 2.12: PBFT communication phase [23]

2f (f can be considered as number of potentially faulty nodes), the process can be operated in the next phase.

- The fourth phase (Commit): Each node will broadcast a commit message which contains the current view and serial number as N. When nodes received the same commit message with a number of 2f, then-current messages can be executed and marked as committed status.
- The fifth phase (Reply): All the nodes will reply result to the client.

2.3 Summary

In this chapter, I have given a detailed review of the related works in most recent blockchains systems, and provide unique insights and the landscape of blockchains.

By analysing the pros and cons of these studies, the gaps from the literature review are summarised as follows.

- Most applications of blockchain concentrate on the benefits for different industries. However, few research works are conducted to investigate long-term issues which are related to the mining environment in consensus protocols for candidateship election and security. By considering the aforementioned challenging issues and the research gaps, this thesis provides research into new consensus approaches.

- Several proof-based consensus protocols have been proposed. However, most of them have shortcomings in the candidateship election scheme in the consensus protocol. In PoS candidateship election schemes, a primary concern is wealth nodes are more likely to get elected. An attacker that controls most of the wealth can exclude and modify the ordering of transactions. That effect leads to a small subset of nodes having the power to control the generation of the block, and it is also potential to increase the degree of centralisation which against the nature of blockchain decentralisation. In the security aspect, it is vulnerable to attacks like guessing attacks. Because adversaries can predict who will generate the next block due to the implementation of PoS is pseudo-random and a small subset of nodes controlling the generation of the block. Thus, providing a unique insight and landscape into the most recent blockchains and deconstruct blockchains into two critical components, identifying their issues and challenges that are the first aim of this thesis.

- The tokenized is used by cryptocurrency domain for staking in most PoS based blockchains as a single factor. The more tokens or coins that miners have, the more trustworthiness that miners own in the financial world. Some miners do not have many tokens or coins that are deposited in the system, but they have modern computing hardware to provide better system performance for stable operation. That is a disadvantage for them. Thus, to provide a comprehensive selection scheme that considers trustworthiness and performance as multiple

criteria for candidate or leader election. Also, to mitigate guessing attacks [26] for election block producers. These are the second aim of this thesis.

- The third aim of this thesis is to propose another scheme that can enhance transaction throughput per second, and transactions can be confirmed immediately, achieving high transaction throughput.

The literature review has pointed out the above research gaps of candidateship election and consensus mechanism, including the process of election nodes and mining environment for participants. This also reveals the in-depth studies demanded in this field. The part of the solution presented in this thesis cover the research gaps by proposing Proof of SecureStake and Proof of Assurance based consensus approaches to address these problems.

Specifically, in Chapter 3, the Poof of SecureStake is proposed, serving as a basis for further investigations. A novel Proof of Assurance, introduced in Chapter 4 is capable to achieve high transaction throughput per second.

Chapter 3

SecureStake based blockchain consensus approach

3.1 Introduction

In Bitcoin system, nodes election scheme can be considered into a race of solving the crypto puzzle, and the first miner who solved the crypto puzzle can have right to mine a block [77]. Therefore, computing power are key to win this race. The miners with advanced computer hardwares are advantage to solve puzzle. While miners without modern hardwares are hard to find a solution for puzzle. Moreover, the process of mining consumes vast amount of energy [78]. In order to address huge energy wasted issue, Proof of Stake (PoS) was proposed in cryptocurrency community forum [49] to replace Proof of Work (PoW) in Bitcoin system. However, some of these approaches have shortcomings on security and performance issues. The outstanding issue of some of other protocols are small subset of nodes being the position of power to control block generation. In other words, these protocols increase potential of centralization and the concerns of governance such as PoS. This also can lead that an attacker that controls most of the "wealth" can exclude and modify the ordering of transactions.

To solve the limitations some of these, we propose a novel consensus approach, called Proof of SecureStake (PoSS). In PoSS, two important factors, node trustworthiness and node performance are considered in consensus approach. The more tokens that a node has owned, the more trustable the node is in the system. CPU, Memory, Storage, Network and Bandwidth are considered as key factors for system performance. Thus, we combine the trustworthiness and performance as stake resources in blockchain system. In PoSS, we only categorise the resources in the form of (1)

the amount of token that deposited in system, (2) CPU power or the number of CPU slices provided in system, and (3) the amount of memory allocated in system as demonstration. Moreover, we partition the entire blockchain environment into different levels of resource pools, and map nodes to different resource pools based on individual's resource. For each round, each resource pool is required to elect a candidate as a representative based on election scheme for reaching a consensus, and a leader is randomly elected from candidates for block generation. After that, the elected nodes from different resources pools are running a BFT-based consensus to commit a block. Eventually, all the participants will update the new blocks from candidates and append new blocks on their chain.

PoSS consensus approach is that we are not only taking token as single "stake" as the measurement for nodes participating over the block generation, but also take node's multiple criteria as the comprehensive performance (e.g., CPU core and amount of memory) into consideration. In addition, our election scheme has high randomness of leader election comparing with pure PoS, which means that an adversary is hard to predict the block producer.

3.2 SecureStake Protocol

In this section, we firstly present terminologies for our approach, then we present an overview of proposed Proof-of-SecureStake approach, after that we discuss a design of Proof-of-SecureStake approach.

3.2.1 Terminologies in PoSecureStake

- *SecureStake*: SecureStake is employed to evaluate comprehensive resources for single node in blockchain network in order to elect a candidate from different resource pools. We employ the three components as main factors, which are the amount of token, number of CPU core and size of Memory.

- *Resource Pool*: Resource pools are defined by dynamic resources for current environment, and dynamic to change periodically. There are four types resource pools in blockchain network according to different levels of SecureSake's score. Nodes with low level of SecureStake are mapped to low resource pool, nodes with middle low level of SecureStake are mapped to middle-low resource pool,

nodes with middle middle-high level of SecureStake are mapped to middle-high resource pool, nodes with high level of SecureStake are mapped to high resource pool. Resource pool in our PoSecureStake can be summarised as following aspects: (1) It maps node with different level of SecureStake's score. (2) Candidate election happens from each resource pool.

- *Candidate*: Candidates are representative from each resource pool based on SecureStake election scheme. Node with high SecureStake score has higher chance became a candidate in each resource pool. Thus, higher SecureStake is always better, however, node with highest SecureStake does not mean it can be guaranteed as candidate due to random function apply to our election scheme. Candidates play the roles which are responsible for upcoming blocks to decide whether the current block should be appended on the chain. Candidates are periodic changed in our approach due to cost of real-time change is too high for distributed nodes.

- *Leader*: After candidates are elected from each resource pool based on election scheme, a leader is elected from candidates by *Random Seed* function. Candidates run *Random Seed* function to generate random number which presents each candidate, then a candidate with highest number is chosen as block producer in current round.

3.2.2 Proof of SecureStake Overview

In PoSS, making the deposit and providing their address to blockchain should be completed for new miners before they join blockchain network. The miners providing public keys are necessary for further communication as their identification. The initial miners are already existing and working in this system before the bootstrapping PoSS approach in order to kick-off the system for our experiments. The method to simulate miners (participants) in blockchain network. The first method is that employing docker instances to simulate the election process. The K-Medoids function is operated by dockers. Figure 3.1 shows an example, there are four types resource pools in the blockchain when k is equal to 4 (k represents the number of resource pools or clusters), and each resource pool has four miners. In our PoSecureStake, there are serval stages to finalise a block: (1) the first stage is to partition resource

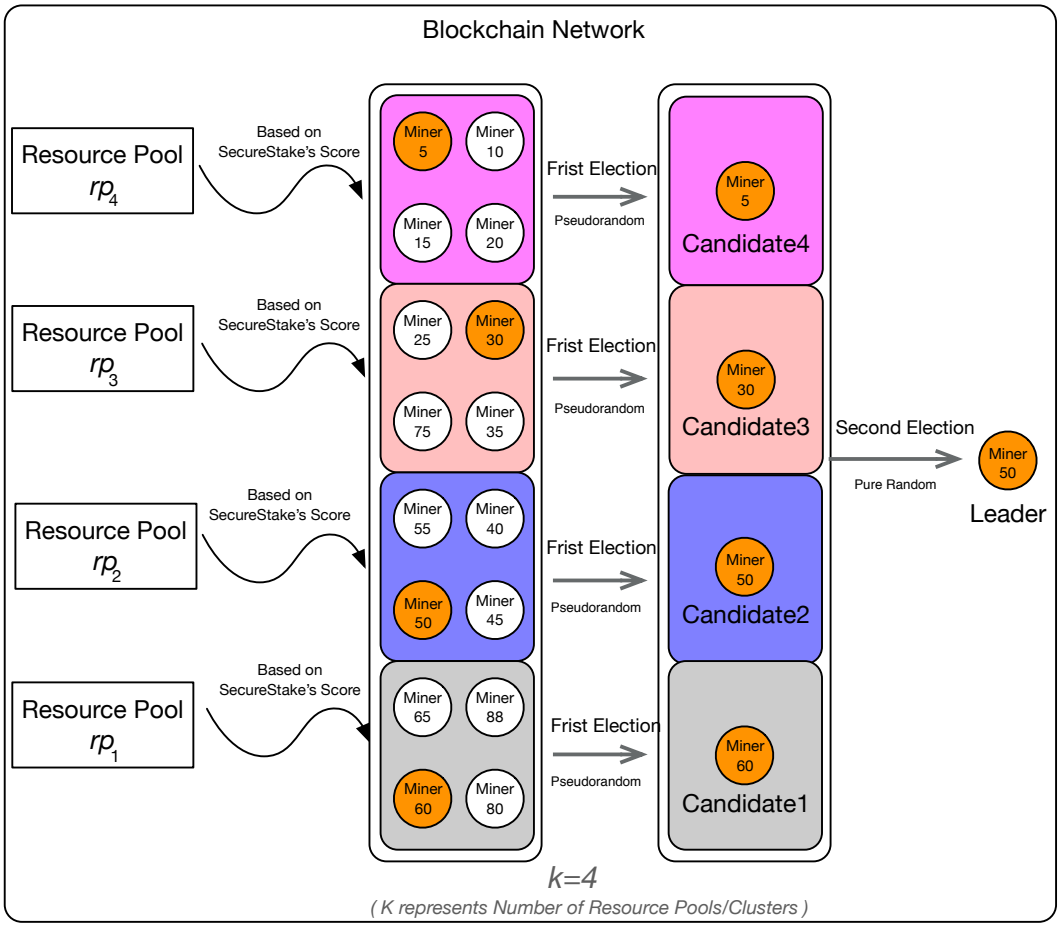


Figure 3.1: PoSecureStake Overview.

pools according to level of SecureStake's score in entire network. (2) the second stage is to label nodes with different level of SecureStake score in order to map them to different resource pools. (3) the third stage is to elect a candidate (miner 5 in RP_4 Resource Pool, miner30 in RP_3 Resource Pool, miner50 in RP_2 Resource Pool and miner60 in RP_1 Resource Pool) from each resource pool based on election scheme. (4) the fourth stage is to elect a leader from candidates in order to produce the block. (5) the fifth stage is to valid the block by candidates. (6) in the last stage, it is block broadcasting process to entire network, and nodes will append this new block on their chain. Eventually, the entire blockchain system is getting updated with new blocks.

3.2.3 Proof of SecureStake Design

In this section, we present design of Proof of SecureStake approach. Our majority focus is on design of election scheme.

In PoS [79], the adversary needs to own 51% of the cryptocurrency in order to launch an effective 51% attack. However, unlike Bitcoin system, adversary in a PoS system is highly discouraged from launching 51% attack because she or he would take risk of depreciation of his entire stake amount to do so. In comparison, anyone acts like a malicious node in the Bitcoin system will not lose their equipment if she or he launched a 51% attack. Moreover, even if a 51% attack succeeds, the value of PoS-based cryptocurrency will drop, and the adversary with the most stake will eventually lose the most [80]. For these reasons, those who attempt to launch 51% of the PoS blockchain will not be easily motivated according to "A Proof of Stake Design Philosophy [81]" Similarly, if someone owns such amount of tokens in our PoSS, it does not have motivations acting like a malicious node. That is reason we consider that the more tokens a miner owns the more trustworthiness it has, and less chance that it acts maliciously. In addition, if the miner acts like a malicious node, the punishment mechanism can be employed to destroy or freeze tokens it owns in future design. That is consideration for trustworthiness of a node based on above assumption.

Apart from trustworthiness, many other factors also have impacts on the performance of blockchain system such as transaction throughput for per second (TPS), but everything starts from "root" which is fundamental infrastructure. Therefore, underlying infrastructure is one of the fundamental factor [82]. In reality, a more advanced computer can respond to requests more quickly, and also provide the stable operations as well. The performance of a computer can be mainly measured by CPU, Memory (RAM) and Storage. Thus, we adopt CPU and Memory as selection criteria to represent a capacity of a node as a demonstration. It can incentive miners to investing on hardware to provide the better performance. Now, miners can invest on either tokens or hardware, or invest on both of them to increases their trustworthy or performance. Moreover, combining concepts of trustworthiness and performance is a good way to restrict each other. Therefore, in this section, we present the design of Proof of SecureStake approach. Our majority focus is on design of election scheme.

In addition, The reason that nodes with similar SecureStake scores are mapped to a

same pool is to mitigate wealthy miners are more likely to get elected. If nodes with similar SecureStake score are mapped to a same pool, these nodes almost have similar chance to get elected. While if similar nodes are classified into different pools, then they may have highest probability to get elected in different pools and have power to dominate generation of block. For example, there are 4 pools, and there are 4 miners (Miner99-1 with 0.991 SecureStake score, Miner99-2 with 0.992 SecureStake score, Miner99-3 with 0.993 SecureStake score, Miner99-4 with 0.994 SecureStake score) in the high resource pools. If Miner99-2, Miner99-3 and Miner99-4 are mapped into middle-high resource pool, middle-low resource pool and low resource pool, then Miner99-2 has highest probability to get elected in middle-high resource pool for each round, Miner99-3 has highest probability to get elected in middle-low resource pool for each round, Miner99-4 has highest probability to get elected in low resource pool for each round.

SecureStake

The factors we define for SecureStake as follows:

- *SecureStake Components*: We break the SecureStake into three different components, and each component can represent individual of resources, i.e., Token t, CPU c and Memory m. For a node $v_i \in V$, we use v_i^t, v_i^c and v_i^m to represent the amount of tokens deposited, the number of CPU slices owned and size of memory allocated by a node. These components are weighted by corresponding scaling parameters, i.e., ω_t, ω_c, ω_m, satisfying $\omega_t + \omega_c + \omega_m = 1$. Equation 3.1 normalises amount of components $z \in \{t, c, m\}$ related to v_i, where $\max(v_i^z)$ and $\min(v_i^z)$ represent maximum and minimum values among all nodes. For example, for a node v_i, we use $\theta_{v_i^t}$ to represent the token's score after normalisation.

$$\theta_{v_i^z} = \omega_z \cdot \frac{v_i^z - \min(v_i^z)}{\max(v_i^z) - \min(v_i^z)}, z \in \{t, c, m\} \tag{3.1}$$

- *SecureStake Vector*: We use $\vec{\theta}_{v_i} = < \theta_{v_i^t}, \theta_{v_i^c}, \theta_{v_i^m} >$ to represent v_i's SecureStake vector, and regard norm of $\vec{\theta}_{v_i}$ as SecureStake of v_i. Then the candidates get elected stochastically based its SecureStake.

- *Probability of Candidate & Leader* : For a node v_i, the candidate for current round is elected, and probability of a node v_i becoming a candidate depends

on its staked resource amount $\vec{\theta}_{v_i}$. Candidate election will be based on a node v_i with the following probability $p_{v_i} = \frac{||\vec{\theta}_{v_i}||}{\sum_{v_k \in rp_n} ||\vec{\theta}_{v_k}||}$ in each resource pool RP_n. After candidates are elected from each resource pool, becoming a leader's probability $p^l_{v_i}$ depends on the number of candidates. We regard a candidate of the resource pools RP_n as $s_{RP_n} \in S$.

- *Leader Election Entropy*: Let $H(p^l_{v_i})$ be the leader election entropy in blockchain. The higher the entropy is the higher the disorder. We employ leader election entropy to quantify or measure of disorder of leader election in blockchain in order to observe of "uncertainty" inherent in the variable's possible outcomes [83] shown in Equation (3.2).

$$H(p^l_{v_i}) = -\sum p^l_{v_i} \log_2 p^l_{v_i} \tag{3.2}$$

Algorithm 1 Computing SecureStake Components

Input: A set of nodes V, scaling parameters ω_t, ω_c, ω_m
Output: SecureStake $||\vec{\theta}_{v_i}||, \forall v_i \in V$

1: **while** $v_i \in V$ **do**
2: $\quad \theta_{v^t_i} \leftarrow \omega_t \cdot \frac{v^t_i - \min(v^t_i)}{\max(v^t_i) - \min(v^t_i)}$
3: $\quad \theta_{v^c_i} \leftarrow \omega_c \cdot \frac{v^c_i - \min(v^c_i)}{\max(v^c_i) - \min(v^c_i)}$
4: $\quad \theta_{v^m_i} \leftarrow \omega_m \cdot \frac{v^m_i - \min(v^m_i)}{\max(v^m_i) - \min(v^m_i)}$
5: $\quad \vec{\theta}_{v_i} \leftarrow < \theta_{v^t_i}, \theta_{v^c_i}, \theta_{v^m_i} >$
6: $\quad ||\vec{\theta}_{v_i}|| \leftarrow \sqrt{{\theta_{v^t_i}}^2 + {\theta_{v^c_i}}^2 + {\theta_{v^m_i}}^2}$
7: **end while**

Computing SecureStake Components

Algorithm 1 presents the SecureStake computing process, which works on the all the nodes in blockchain network. We count the amount of tokens, the number of CPU slices and size of Memory for individual node. We also count total the amount of token, the number of CPU slices and the size of Memory in blockchain. Each component is normalised, then weighted with scaling parameter ω_t, ω_c, ω_m (Lines 2-4, Algorithm 1). Furthermore, we obtain SecureStake for v_i by calculating norm of vector $\vec{\theta}_{v_i}$ (Lines 5-6, Algorithm 1). In addition, we set $\omega_t = 0.2$, $\omega_c = 0.4$ and $\omega_m = 0.4$ in our simulation experiment. However, scaling parameters ω_t, ω_c, ω_m can be adjusted based on different requirements.

Algorithm 2 Resource Pool

Input: The number of resource pools k, SecureStake $||\vec{\theta}_{v_i}||$ for all $v_i \in V$
Output: All resource pools $RP_n, n = 1, ...k$, set of nodes $v_i \in RP_n$
1: Randomly initialise k medoids as $\{CR_1^*, ..., CR_k^*\}$
2: **repeat**
3: **for** $v_i \in V$ but not belong to $\{CR_1^*, ..., CR_k^*\}$ **do**
4: Assign v_i into the pool $\underset{RP_n, n\in[1,k]}{\arg\min} \left| ||\vec{\theta}_{v_i}|| - ||\vec{\theta}_{CR_n^*}|| \right|$
5: **end for**
6: **for** $n \in [1, k]$ **do**
7: **for** $v_i \in RP_n$ but not belong to $\{CR_n^*\}$ **do**
8: Swap v_i and CR_n^*, recompute the cost by using Equation 3.3
9: **if** Cost C decreases **then**
10: Replace CR_n^* with v_i and let v_i be the new medoid of RP_n
11: **end if**
12: **end for**
13: **end for**
14: **Until** cost C is no change
15: **Return** RP_n, and v_i

Resource Pool

In order to get the number of resource pools, k-Medoids clustering function is used to generate the number of resource pools. A medoid can be defined as the point in the cluster, whose dissimilarities with all the other points in the cluster is minimum [84]. In Algorithm 2, k-Medoids clustering consists of two phases called [85]; (1) Build Phase: k initial medoids are randomly selected for k resource pools (Line 1, Algorithm 2). (2) Swap Phase: It is carried out by the cost function that reduced by interchanging a selected medoid with a node v_i. This process is continual until cost function no longer decreases (Lines 2-14, Algorithm 2). The cost function in k-Medoids algorithm is given in Equation 3.3 [86], where CR_n^* represnts the medoid of resource pool RP_n, and $||\vec{\theta}_{CR_n^*}||$ is norm of $\vec{\theta}_{CR_n^*}$. For example, it will output 4 resource pools with set of nodes in different pools, if we set $k = 4$

$$C = \sum_{n\in[1,k]} \sum_{v_i \in V} \left| ||\vec{\theta}_{v_i}|| - ||\vec{\theta}_{CR_n^*}|| \right| \tag{3.3}$$

Election Leader

In leader election stage, the election process is explained in Algorithm 3. Firstly, candidates should be elected from each resource pool base on probability p_{v_i} (Lines 2-6, Algorithm 3). Then, a leader is required to elected from candidates by using random seed function [87]. We use random seed function to generate random number for each candidate (Lines 7-9, Algorithm 3). Then we pick the "luckiest" one as a leader, which has the highest number by random seed function (Lines 10, Algorithm 3). In our example, probability of a leader being elected from candidates is also based on number of candidates. In addition, random seed function can generate result which shall be quoted in sixteen decimal places. Therefore, it is almost impossible to have same number for each round.

Algorithm 3 Election Leader

Input: All resource pools $RP_n, n = 1, ...k$
Output: Elected leader l
1: **for** $n \in [1, k]$ **do**
2: **for** $v_i \in RP_n$ **do**
3: $p_{v_i} \leftarrow \frac{||\vec{\theta}_{v_i}||}{\sum_{v_k \in rp_n} ||\vec{\theta}_{v_k}||}$
4: **end for**
5: Randomly choose the candidate s_{RP_n} based on probability p_{v_i}
6: **end for**
7: **for** each $s_{RP_n} \in S$ **do**
8: $\lambda_{s_{RP_n}} \leftarrow RandomSeed$
9: **end for**
10: $l \leftarrow \arg\max_{s_{RP_n}} \lambda_{s_{RP_n}}$

3.3 Threat Models

For an adversary launches an effective attack to compromise PoSecureStake approache at the high level, an attacker must to compromise at least $\frac{1}{3}$ resource pools because of Byzantine fault tolerance [6]. Adversary can achieve this effective attack by compromising the election in more than $\frac{1}{3}$ resource pools or compromising the candidates directly. The adversary needs to ensure that malicious nodes get elected for each round for later attack because candidates are periodic changed.

3.3.1 Network Infrastructure Attack

Message propagation in blockchain network can be delayed by blocking communication between candidates in network infrastructure [88]. The adversary also can disable candidate's communication between requests messages and response messages by launching an effective Denial of Service. Thus, these attacks can conduct two main issues: (1) delay the communication between nodes, and node cannot update their chain immediately if there is a new block produced. (2) failed to elect candidates and leader due to delay or disabled communication between nodes.

3.3.2 Node Election Attack

To launches an effective attack on node election, the adversary need to modify the malicious node's SecureStake score. Adversary can create the fake SecureStake when nodes are completely under the adversary's control. Another approach is that adversary pretends to honestly in blockchain, at same time adversary increases SecureSake score. Eventually, adversary launches attack against blockchain system when malicious nodes are elected as candidates.

3.3.3 Analysis attack in network infrastructure

It is the difficult to deploy attack for targeting network infrastructure. In modern network structure, the network infrastructures will load balance the traffic flow from congested network equipment to other equipment, when network congestion happens in network. Unless, the adversary conducted an attack which disabled communication between all network infrastructures.

3.3.4 Analysis attack the node election

We assume that the number of the nodes are honestly in PoSS approach. Thus, the adversary compromises entire PoSS approach that is difficult. In our approach, probability of being elected as candidates dependents on SecureStake score on each resource pool and probability of being elected as a leader dependents on the number of resource pools. Thus, the adversaries should maintain a high SecureStake score in each resource pool before the attack. We assume all the miners have the same SecureStake in order to simplify security model, and the malicious node maintains a SecureStake in a resources pool, the probability of p_f being elected as a leader across

all the compromised resource pools, attacking the protocol successfully of controlling the block generation is shown in Equation (3.4).

$$(\frac{1}{S} * \frac{S-1}{3}) * (\frac{||\vec{\theta}_{v_i}||}{\sum_{v_k \in rp_n} ||\vec{\theta}_{v_k}||})^{\frac{S-1}{3}} \tag{3.4}$$

For the 4 resource pools, if the probability of malicious node being elected is $p_f = 30\%$, the adversary will have 7.5% probability of control on generation of block . Thus, difficult level of controlling the block generation dependents on the number of resource pools. The number of resource pools are increase, difficult level is increase. However, in reality, to maintain p_f that is difficult due to our election scheme, because (1) the nodes with different levels of SecureStake are mapped to different level of resource pool, that means nodes with similar level of SecureStake score are together. That results that the most of nodes have similar probability of being elected as candidate for each resource pool. (2) the probability of being elected as a leader is pure random, which conducts election process is hard to predict by adversary.

3.4 Experiment and Analysis

We carry several experiments based on a prototype deployment of PoSecureStake approach in order to observe the election scheme for our approach when implementing it blockchain network.

3.4.1 Experiment Setting

We employed one 12 cores Intel XEON E-2696v2 CPU with 128GB memory to deploy Linux system as foundation, then we deployed SecureStake core algorithm in the Docker [89] environment which is an open source software, providing container technology. We run 100 docker containers which represents PoSecureStake participants or nodes. We implement Rancher [90] which is a platform to manage our containers, and also we can easily create Peer-to-Peer networks of different sizes. We suggest that using BFT-SMaRt [91] as consensus box among all candidates for running PBFT to reach an agreement in each round. However, the details of this will not be discussed in this Chapter as it is not a major focus of this Chapter 2.

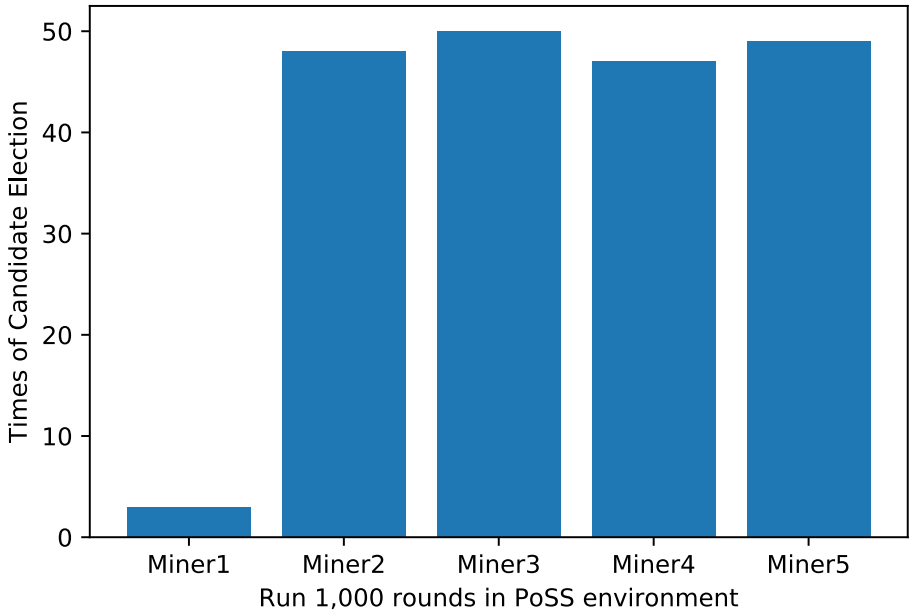


Figure 3.2: Candidate election of the chosen miners .

3.4.2 Election Scheme Experiment

Election frequency between different nodes

At the beginning of experiment, we created 100 instances as nodes. Then, we deployed SecureStake algorithm in each instance and we manually set each instance with different levels of SecureStake's scores. After that, we run the experiment for 1000 times in order to observe the elected times for nodes with different SecureStake's scores as candidates and leader. In specifically, we pre-rank SecureStake's score of each miner from top to bottom, then we choose 5 miners in blockchain. Thus, we choose Miner1 which has lowest score, Miner5 which has highest score, Miner3 which has median score between the Miner1's score and Miner5's score, Miner2 which has median score between the Miner1's score and Miner3's score, Miner4 which has median score between the Miner3's score and Miner5's score. After that, we set the number of resource pools (clusters) as 4 (k=4). Finally, we run the experiment 1000 times in our PoSecureStake's scheme in order to observer that the elected times as candidates and leader for the chosen miners. We also run the experiment 1000 times in PoS environment for chosen miners in order to compare experiment's results.

For candidate's election, Figure 3.2 presents results of the chosen miners in PoSS.

Miner2, Miner3, Miner4 and Miner5 have almost same chance being elected as candidates, excepted Miner1 with lowest level of score in blockchain. The reason times of candidate election for Miner1 is lower than Miner2, Miner3, Miner4 and Miner5 is that Miner1 has lowest level of score in low resource pool. Therefore, probability of Miner1 to get elected is lower than others. The Miner2, Miner3, Miner4 and Miner5 have similar level of score in themselves resource pool. Thus, they have closed probability to get elected. For leader's election, Figure 3.3 demonstrates the leader election result in PoSS and PoS, Miner2, Miner3, Miner4, and Miner5 almost have same chance being elected as leaders in PoSS environment. Also, it proves that Miner5 with highest stake does not get elected as leader all the time, and Miner2, Miner3 and Miner4 get elected frequency is almost same to Miner5. After that, we compare PoSS experiment's results with running same miners in PoS and Nxt experiment's results, the frequency of election is significant increased when miners's stakes are increased in PoS environment. Due to stake is visible on public, so higher stake holders or miners always have higher chance elected as a leader or block producer in order to produce blocks. Thus, potential risks can be happened on this assumption for PoS and Nxt: An attacker can attack small subset of nodes with higher range of amount instead of seeking for entire network or an attack that controls most of the wealth can exclude and modify the ordering of transactions. However, our PoSS approach avoids that small subset of nodes always has higher possibility to be elected, which can mitigate potential risks that brought from wealth.

3.4.3 Leader Election Entropy

We employ information entropy to calculate leader election entropy between PoS and PoSS. In information theory, the entropy of a random variable is the average level of uncertainty inherent in the variable's possible outcomes. In other words, higher entropy is more disorder and random for leader election. The value of leader election entropy for PoS is around 2.49, for Nxt Coin is around 2.32, and value of leader election entropy for PoSS is around 3.24 by calculating elected time for 1000 round between PoS, PoSS and Nxt Coin environment. The reason leader election should be unpredictable (random enough) due to prevent guessing attacks, where an attacker is able to guess or predict a valid leader election session through statistical analysis. Our proposed approach entropy is higher than PoS and Nxt. It means that it provides better random function, and it makes process of leader election is hard to predict than PoS and Nxt.

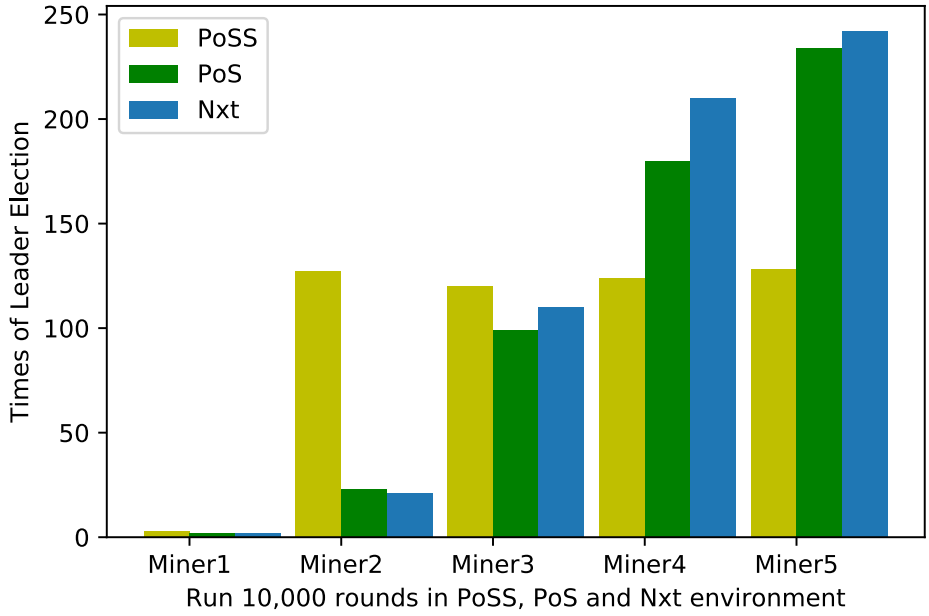


Figure 3.3: Leader election of the chosen miners.

Impact of PoSS variation on node election

We created 20 virtual machines (Ubuntu Server 16.04 LTS) on VMware vSphere ESXI 6.5 platform, then we deploy SecureStake core algorithm in each virtual machines. We set each virtual machine has same amount of tokens that deposited in system and also set same CPU Slice to each virtual machine. We assign different size of Memory to each virtual machine in order to observe impact of PoSS election frequency on node. There are 20 virtual machines in total from Miner21 to Miner40. We choose Miner21, Miner23, Miner29, Miner35 and Miner40 to observe times of leader election. The Memory value of Miner29 is treble of Miner23 and Memory value of Miner35 is fivefold of Miner23. Miner21 has lowest amount of Memory value and Miner40 has highest value. After that, we run PoSS algorithm in each virtual machines in order to observe experiment's results. Figure 3.4 illustrates that miner with highest Memory value does not get elected as leader all the time. Furthermore, chance of being elected as leader is almost same as Mine23, Miner29 and Miner35. It means that the impact of Memory in leader election process is similar to Token.

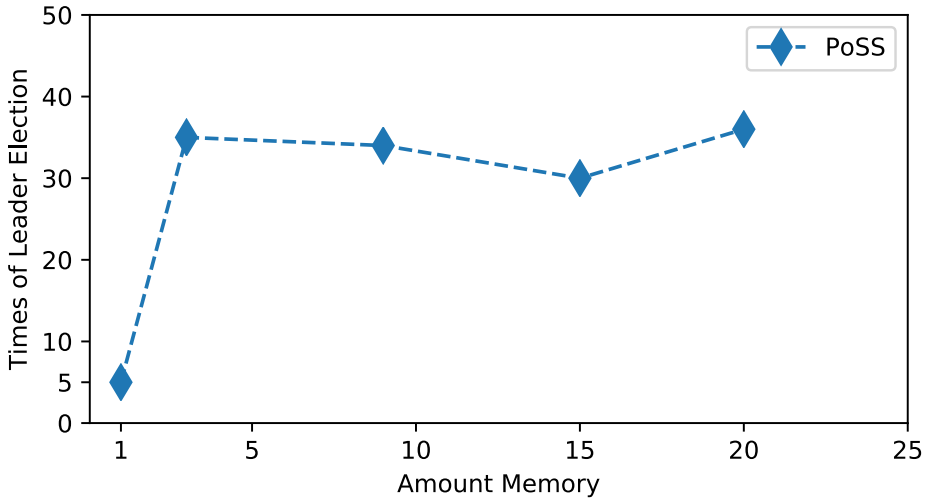


Figure 3.4: Impact of PoSS variation on leader election

3.5 Summary

This chapter introduces PoSS, a novel consensus approaches for blockchain system. PoSS provides multiple election criterion in leader election scheme by combining trustworthiness and performance as comprehensive performance. Also, it mitigates guessing attack by increase information entropy in leader election process. Thus, leader election process is hard to predict by adversary.

This chapter mainly answers the Research Question 1 mentioned in Chapter 1. The next chapter aims to investigate performance in transaction throughput from main blockchains, and provide novel approach to improve that problem.

Chapter 4

Proof of Assurance based blockchain consensus approach

4.1 Intruduction

A consensus approach is critical component to blockchain systems. The design of the consensus approach strongly impacts security, scalability and transaction capacity in the blockchain systems. The traditional consensus approaches, such as PoS and PoW have encountered great challenges and require further investigation, and we already present these challenges in Chapter 2. In this chapter, we propose a novel consensus approach, named Proof of Assurance (PoAssu), which introduces our scheme based on PoW mining process to work with Practical Byzantine Fault Tolerance (PBFT) verification, which together allows for transactions to be confirmed immediately.

The distributed ledger is maintained by each participant, which provides the total order of transactions and runs a distributed consensus mechanism to ensure the consistency of the distributed ledger. Cryptocurrency is the underlying technology behind Bitcoin or other blockchains that have attracted broad attention in both academic and financial industries. Blockchain is designed for a decentralised and secure environment. However, Blockchain faces scalability barriers such as low transaction throughput. Transactions per second (TPS) of Bitcoin is around 7 (TPS) [92] and Transactions per second (TPS) of Ethereum is around 25 (TPS) [93]. Transactions throughput in Bitcoin and Ethereum are far from meeting the high trading demand in actual world. Thus, there are many research studies such as [94, 95, 96, 97] have been investigated to improve the Nakamoto consensus.

Variants of the Nakamoto consensus are still having shortcoming on low transactions

throughput due to decentralisation and probabilistic consistency. It is an inherent problem between performance and security. In contrast, classical BFT protocols offer much shorter period of blocks and lower latency with the deterministic consistency. However, these protocols are suitable to apply in closed environment with fixed and pre-defined set of participants. Moreover, several hybrid protocols have been proposed [98, 99, 100, 101]. However, BFT protocols still have issues on scalability, when increase size of participants. That reduces transaction throughput scales linearly as network size increase [102].

To solve these limitations, we introduce the Proof of Assurance (PoAssu) protocol. It is a novel consensus approach for candidaship election in a permissionless environment. First, the entire blockchain network is pre-partitioned in several areas, and miners are free to join any area they would like to participate in. Secondly, PoAssu has a group to representative validators who elected periodically by adjusted PoW in each area. Then, PoAssu scheme assigns validators to assurance committee members without any honest third party or distributed randomness generation protocol. After that, the assurance committee members will share the transactions they get from each assurance area, and assurance committee members will apply PBFT to determine which transaction involve in the block proposing process. Finally, the assurance committee members broadcast the new block into their own area, and the rest of the miners are going to verify block.

4.2 Outline of Bitcoin Consensus Mechanism

The Bitcoin consensus requires that the significant work must be proofed (Proof of Work) for newly block to appended on chain. A valid PoW in Bitcoin meets the following Equation (4.1).

$$H(PreHash, Difficulty, Time, TxRoot, Nonce) \leq Target \tag{4.1}$$

In above Equation, H represents hash function (Sha-256), Target value is used in mining. It is a number that a block hash must be below for the block to be added to the blockchain. PreHash is hash value of the previous block's header, Difficulty value is a measurement of mining which scales the Target within hash value range, Time is the timestamp, TxRoot is the Merkel root [103] of all transactions in the block, and Nonce is a random number that is typically used once. In the mining

process, the aim is to solve a hash below the current target.

The Hashcash Proof of Work concept [104] is used in the Bitcoin system. The original PoW approach is based on the Sha-256 hash function. This function is used in Merkle-Damgard construction, however, length-extension attack [105] is weakness for using Sha-256 hash function. We suggest that using Blake3 [106] replaces Sha-256, as the Blake3 function uses Subtree-freeness, and Subtree-freeness ensures that generalized length-extension attacks, like the ones that plagued Merkle-Damgard, cannot happen. Also, the time spending on computing Blake3 that is much faster than MD5, Sha-1, Sha-2, Sha-3, and Blake2, at the same time it has the same safety as Sha-3. Therefore, using Blake3 can make the protocol more efficient and robust.

The generation of a block in Bitcoin can be detail as following procedure:

- Each miner is going to collect the transactions into a memory pool when the blockchain network broadcasts the transactions. Miners will choose transactions from the pool, and using chosen transactions to generate a Merkel-tree when mining is beginning.
- Miner produces a temporary block that contains block of header and body. The previous block header hash, difficulty, transaction Merkle root, and timestamp are in the header. The transaction list is in the body.
- A miner keeps putting different Nonce into the temporary block and the miner keep calculating the hash of the block header until the miner finds the solution or receiving a new block from the blockchain. Miner will broadcast the new block when miner finds a valid solution for Nonce. Otherwise, the miner will return to collects the transactions and start over again.

In Bitcoin, each miner is working as an individual. There is a very high probability for miners working in the same transaction list, which conducts wasting of hash computing power. Also, it can lead to multiple valid blocks in the same height when miners generate the same preceding blocks. PoW is tolerance of the possibility of forks, thus miners are able to choose blocks from branches as their preceding block. However, a new block can be committed in the P2P network, when miners chose the block with the longest chain. This is how the Bitcoin system resolved the forks eventually. Figure 4.1 shows an PoW Overview.

In section 4.3, we propose a Proof of Assurance (PoAssu) approach to address the

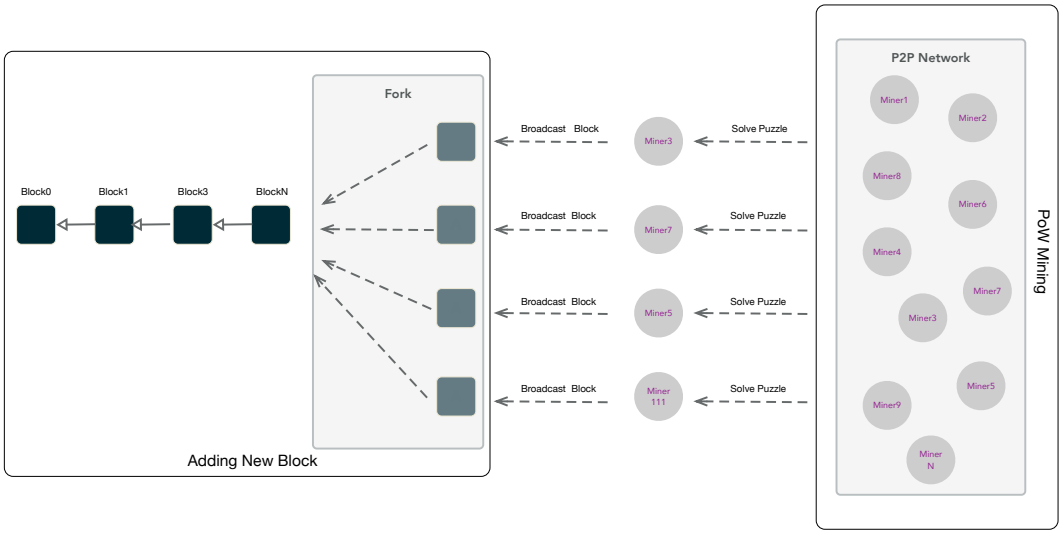


Figure 4.1: PoW Overview

problem in PoW. In Section 4.4, we present how PoAssu is going to ensure the consistency and correctness of the verification. Then, the security problem is discussed in Section 4.6. After that, we present experiment detail for PoAssu in Section 4.6. Finally, a summary of our proposed approach is concluded in the last Section 4.7.

4.3 Proof of Assurance Design

In this section, we describe our proposed PoAssu approach, Proof of Assurance. We present design of the Assurance scheme, Assurance Area, and Assurance Degree.

4.3.1 Assurance

For the PoW mining mechanism, to reach the consensus, the miners are required to solve a certain level of difficulty. It needs to generate at least 32 of leading zeroes to gain block proposal's right [107]. In other words, miners need to keep trying the puzzle individually. The huge workload conducts in a major loss of efficiency. Therefore, Assurance Areas, Assurance Degree and Assurance Committee Members are designed to improve this process.

Assurance Areas

Let us first introduce a concept of a blockchain network in our PoAssu called "Assurance Areas". In the original Proof of Work mining system, all the nodes can be

considered in the same "boat" by the P2P network. In our PoAssu, we partition blockchain network into small subset network called "Assurance Areas". Assurance Areas are pre-partitioned before miners joining. Nodes are free to join any Assurance Areas they would like to participate in. The main reasons partition blockchain network into Assurance Areas in our PoAssu that can be concluded as follows:

- Assurance Area make the process of block broadcasting becomes simplified. Nodes receive update by querying new block from the committee only within Assurance Area instead of broadcasting newly blocks in entire blockchain network. In theory, network convergence time can be reduced.
- Assurance Area optimises the node distribution and reduces the nodes' workload.

As mentioned, Assurance Areas are pre-defined before starting PoAssu protocol. Nodes choose any Area they would like to join, however, ideally, we suggest that nodes join the Area where it is physically nearby their location in reality due to latency consideration in the network. Each Assurance Areas selects a node as a committee member to propose the transactions and put them in the block. To increase the transaction throughput, each Assurance Area elects a committee member and caches S committee member in the queue. The purpose of cache is used to next $S-1$ block proposal rounds and Assurance Area does not have to perform the election committee member again.

Assurance Degree

Let us break down mining Target and mining Difficulty, before we introduce the design of our Assurance Degree in our PoAssu.

Target and Difficulty In the original Proof of Work mining, the difficulty level is determined by the amount of time to find such a specific hash value and computed as Equation (4.2).

$$D_{Current} = T_{Genesis}/T_{Current} \tag{4.2}$$

In this expression, $D_{Current}$ presents the current difficult level, $T_{Genesis}$ is the target value of $Genesis$ block, $T_{Current}$ is the current target value of current header

of block and targets are 256-bit numbers with at least 32 leading zeros. Bits of block header marks that the current hash of block header must be less than or equal to Target based on Equation (3). The result of the block header after SHA256 is 256 bits long, equivalent to 32 bytes. Then, Bits can be calculated by Equation (4.3) to get Target.

$$Target = Coefficient * 256^{(exponent-3)} \tag{4.3}$$

For example, Bits of the real block for height 277316 [108] in Bitcoin is 0x1903a30c (decimalise: 419668748), that value has been stored as exponent of number of the hexadecimal for fist of two number (exponent = 0x19) , the rest of six number as the coefficient (coefficient = 0x03a30c). Therefore, we can get this block's Target is 238,348 $*256^{22}$ (0x03a30c $*256^{(0x19-3)}$), as result of the hexadecimal is 0x0000000000000003a30c.... It means that the valid hash of height of block in 277316 must be less than or equal to this value of this Target. In fact, a hash of the height of the block in 277316 is 0x0000000000000001b6b9a..., and it is valid for the requirement. Finally, we can move on to Difficulty. According to Nakamoto's Genesis Block 0 with 0x1d00ffff bits (decimalise as 486604799) [109], and Difficult is 1. According to the Bits, we can get $T_{Current}$ is 0x00ffff $* 256^{26}$, as result of the hexadecimal is 0x00000000ffff00000000.... It means that miners need to keep computing SHA-256 for header of the block until finding a result which is first of 32 bits are all less than or equal to 0. The result of computing for SHA-256 is a uniformly random sequence, and the value in each number of results of SHA-256 is the same probability for either 0 or 1. Thus, to satisfy above Target's requirement (the first of 32 bit are all less than or equal to 0) in order to find a valid block, the probability of the first of 32 bit is equal to 0 that is $\frac{1}{2^{32}}$. In other words, it needs computing time of 2^{32} in average in order to find this value.

In order to better understand this difficulty, we can consider that the computing time in an average of finding required 0 is defined as Difficulty. Thus, 1 Difficulty is required to compute around 2^{32} times ($4.2 * 10^9$ times $\approx 4G$ times). For instance, Bits of the real block height in 501509 [110] in Bitcoin system is 0x18009645 (decimalise:402691653), the $T_{Current}$ is 0x009645 $*256^{21}$. $T_{Genesis}$ is 0x00ffff $*256^{26}$ with Difficulty 1. Difficulty of the current block 501509 is $\frac{(0x00ffff*256^{26})}{(0x009645*256^{21})} \approx 1.87 * 10^{12}$ ($1.87T, 1T = 10^{12}$). Thus, we can get Difficulty depends on the hash of Target of

the block header, Target is smaller and mining is more difficult. Moreover, it affects time of block generation. If we can deal with very heavy or expensive work from the mining process, we can improve transaction throughput in the system.

Design of Assurance Degree As mentioned, we use Blake3 to replace the original Sha-256 hash function in Proof of Work. Blake3 basically provides the same length of output as Sha-256, but it is more secure than Sha-256, and much faster than MD5, Sha-1, Sha-2, Sha-3, and Blake2. From a cryptographic hash functions perspective, no matter Sha-256 or Blake3 that we use in our approach, any input to the hash function will generate a specific value in the output which is random. For example, the probability of a miner finding zero is $\frac{1}{2}$, then for an output of z specific bits has probability of at least $(\frac{1}{2})^z$. Thus, a miner has probability of at least $\frac{1}{2}$ to find z leading zero, and miner is expected trying 2^{z-1} input as an average. Now for the assurance degree, we use Δ as a unit for the assurance degree (equal to difficulty level). For example, 1Δ means that the assurance degree of a miner to consume own computing power on calculating a hash code for 1 bit of leading zero. In the similar way, $z\Delta$ means it needs number of z hash bits to achieve leading zero. Since mining is required the amount of hash operations to obtain the specific number of leading zero, we can get the following expression regarding Δ as follow Equations 4.4:

$$z\Delta = 2^{z-1}\Delta_t \tag{4.4}$$

In Equations (4.4), the mining is successful when there is at least one miner with the expected assurance degree with a certain predetermined number of leading zeros. For example, the degree of assurance is for a miner who spent own power for calculating hash code with z bit of leading zero. Let's say MinerA consumed $\alpha\Delta_t$ power to success mining work if $\alpha\Delta_t$ is equal or lesser than $z\Delta$ with a specific number of z leading zeros. After that, MinerA cloud be considered as one of assurance committee members for the further generation of the block process.

Assurance Committee Members

An assurance committee member is a representative which is selected according to their assurance level within an assurance area. Assume there are three miners X, Y, and Z who are all generated number of z leading zeros in an assurance area. Miners X, Y, and Z with consumed power of $\varpi\Delta_t$, $\rho\Delta_t$, $\varsigma\Delta_t$ where $\varpi\Delta_t < \rho\Delta_t < \varsigma\Delta_t$

and $\varpi\Delta_t, \rho\Delta_t, \varsigma\Delta_t \leq z\Delta$. In this situation, miner X will be selected as an assurance committee member in this area for block generation. An assurance committee member has responsibility to communicate with other peers. For current round, assurance committee members need to work together for transactions that need to be appended on chain.

A certain number of assurance committee members are elected and ordered in a queue for each round. In the above situation, miner Y and Z will be considered as committee members to order in a queue where it is treated as a cache.

4.4 Consistency and Correctness of the Verification

In the last section, we proposed the assurance degree for mining to elect assurance committee members in each area. In our PoAssu, we use an assurance degree scheme to find assurance committee members in order to participate in the process of block generation. The candidate block only can be generated by assurance committee members. In that case, after one of the assurance committee members generates the candidate block, this assurance committee member must submit it to a verifier group grouped by assurance committee members from all areas, assurance committee members are going to run PBFT to ensure the correctness and consistency of the verification in asynchronous network [111]. Figure 4.3 shows an overview of the Proof of Assurance protocol.

The synchronisation is critical to all participants in distributed systems. In PoAssu, we suggest that employing Network Time Protocol (NTP) to achieve synchronisation. 10 millisecond to 100 millisecond is typical delay of NTP, delay is acceptable in synchronisation environment. The time-out scheme is applied to an exception happens. For example, there is a timer set in an assurance area for all miners during the block proposal round, in case of block producer quitting from network. The node are going to request latest updated blocks from other committees, if there is no block update to this round within own assurance area. We employ time-out method to ensure when there is an exception happens in PoAssu. For example, an assurance committee member leaves the blockchain network during proposal round. Thus, timer-set is existing within an area for nodes. The nodes are able to request the updates from the other assurance committee members, when there is no new block is updated in this round after certain of time.

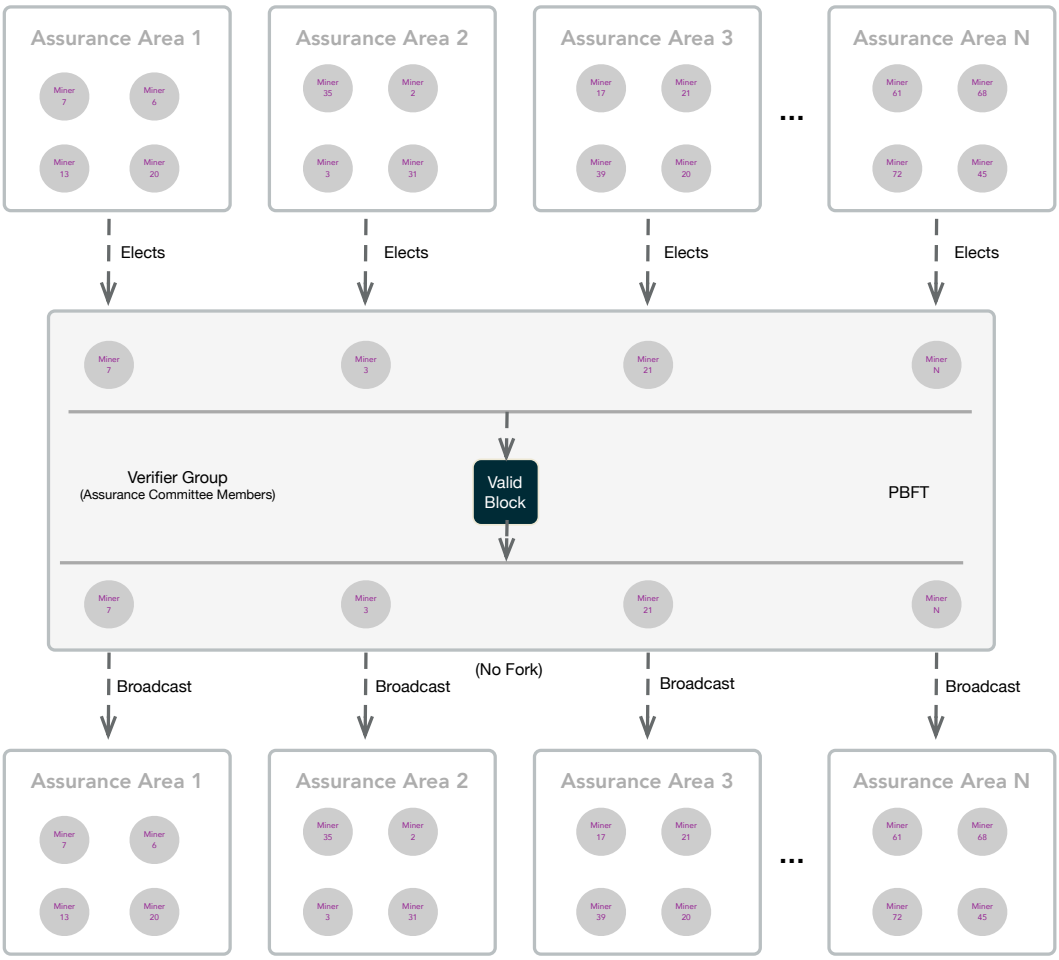


Figure 4.2: PoAssu Overview

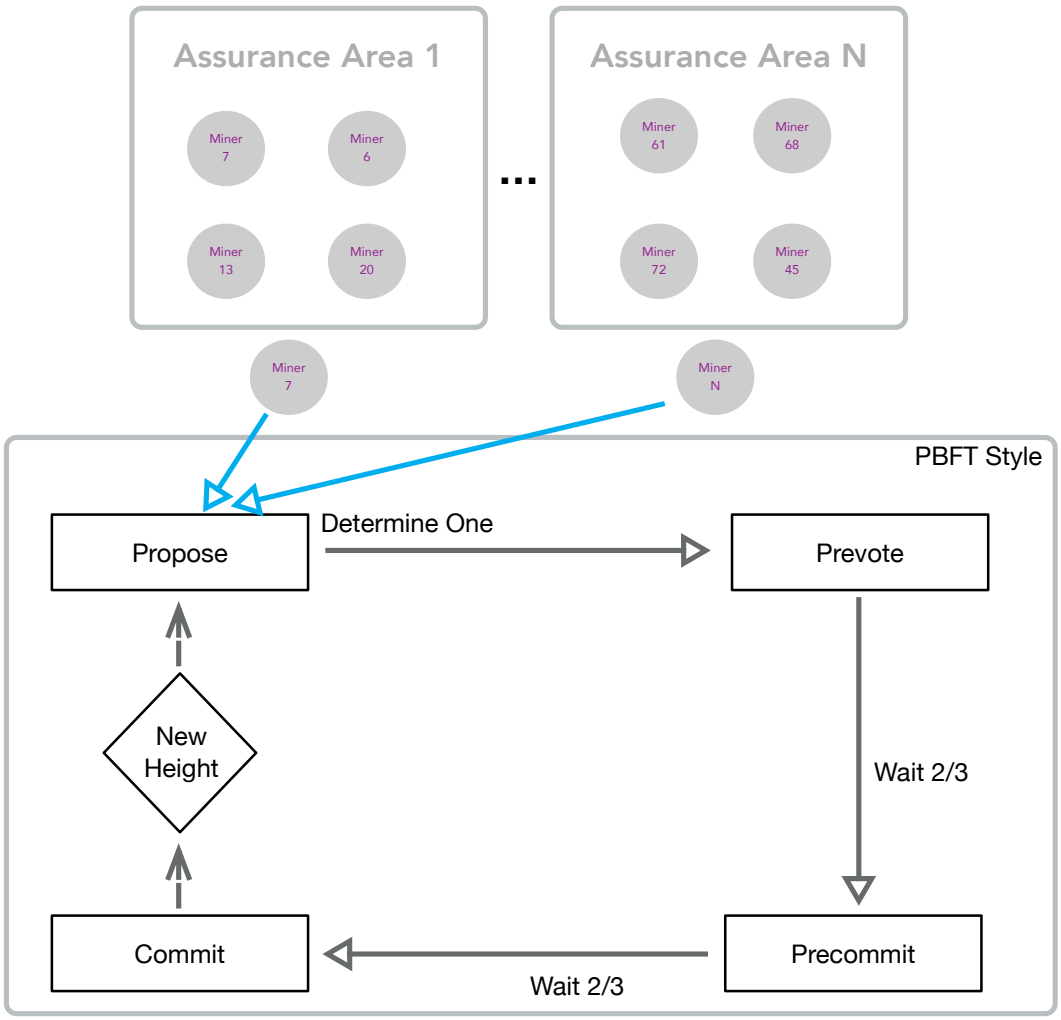


Figure 4.3: PoAssu adapted with PBFT-Style

Nodes are grouped by assurance area. Assurance committee members broadcast the block to assurance committee members. Only one block is allow to verify and broadcast to the network at a time. The assurance committee members are going to verify a certain number of blocks before they are replaced by a new assurance committee members. The time-window within which each chosen assurance committee member performs tasks which are single block verification procedure for each round.

In PBFT protocol [112], the verification process contains four steps, shown in Figure 4.3. Normally, the protocol starts with step one: "Propose". Through the "Prevote" and "Pre-commit" steps, verifiers vote and reach consensus on either a proposed block. After that, they process the "Commit" step to broadcast the verified block in the network. This how blockchain height is increased for each round based on this

cycle.

4.5 Proof of Assurance Interactions

In PoAssu, block proposal has three major phases. In phase one, each assurance area elects a node as an assurance committee member, an assurance committee member is a representative its assurance area to proposes the transactions. In phase two, an assurance committee member decide which transaction needs to put in the chain, and share with peers. There is only one block proposal determined and rest of generated block are in the order. In phase three, all nodes are going to request the latest blocks from the assurance committee member within own area.

In the bootstrapping system stages, we describe the before of system starts and we present how PoAssu elects assurance committee members based on their assurance degree. Then, we discuss how assurance committee members work together to propose blocks in our algorithms.

4.5.1 Bootstrapping system

The initial miners are already existing and working in number of assurance areas before the bootstrapping PoAssu approach in order to kick-off the system for our experiments. There are at least four assurance areas required to set in the protocol in order to the tolerance of one faulty replica, and each assurance area should elect at least one assurance member as a representative for its area; otherwise, the PBFT protocol is not able to achieve an agreement during the block generation. In addition, we mainly focus on the election scheme and how to enhance performance in transaction throughput for the current experiment's environment, thus, the procedure of bootstrapping the new nodes will not be covered in the thesis.

4.5.2 Assurance Committee Member Election

The election process for the initial round is explained in Algorithm 4 for demonstration one area. Firstly, each node starts to mining process as shown in Line 1 Algorithm 4. Miners keep mining until reaching for $z\Delta$ with z leading 0 within its area (Lines 2-9 in Algorithm 4). For miners who obtained z leading 0 with different level of assurance degree $n\Delta_t$, miners' values are sorted (Lines 10-11 in Algorithm 4). After that, an assurance committee member $\Lambda_{committee}$ is elected based on the

smallest value of Δ_t, and let $T_{candidates}$ as the number of candidates for the assurance committee members that need to cache in the queue (Lines 12 in Algorithm 4). Once an assurance member is elected, a verifiable random function (VRF) produces a proof for a committee member π who claims itself as a committee member, and a committee member broadcast hash with π (Lines 14-15 in Algorithm 4). Other miners who are not elect by assurance committee members are able to verify miner who claims to be position in the assurance committee member in this area by identifying of $< hash, \pi >$ (Lines 16-18 in Algorithm 4). In order to do further decisions on clone the new block from this assurance committee member. An assurance committee member is registered to notify other miners within the area.

Algorithm 4 Node Election

```
1: miner.start()
2: notFound = true
3: while notFound do
4:     nΔ_t ← miner.compute(zΔ with z leading 0)
5:     pass ← miner.seal()
6:     if pass = true then
7:         noFind ← false
8:     end if
9: end while
10: for miners with nΔ_t do
11:     nΔ_t ← sort(nΔ_t);
12:     Λ_committee and T_candidates ← elect and catch
13: end for
14: if myself is Λ_committee then
15:     < hash, π > ← VRF.broadcast(hash, π)
16: else
17:     verify.(hash, π)
18: end if
```

4.5.3 Block Proposal

Assurance committee member sorts and proposes a list of the number of transactions based on their area. All assurance committee members should conduct verifying the identity of miners who claim to be assurance committee members before PBFT's procedure is beginning to process transactions. Thus, peers need to check whether the proof which is generated by VRF $(hash, \pi)$ matches with the committee member. It is a similar process to miners checks the identity of the assurance committee within its area. PBFT generates the ordered transactions in the block which are agreed by all assurance committee members, when transactions are given from different

assurance areas. Assurance committee members have to face four majority steps ("Propose", "Prevote", "Pre-commit", and "Commit") in order to sign the current block, ensure that block is valid, and make this block is appended on the chain. If an agreement is not reached in PBFT process, all the nodes can be notified about failure information.

The high-level procedure of implementation PBFT with block proposal presents in Algorithm 5. We trade the PBFT Style algorithm as a consensus box [113], which means that each proposing block with transaction lists needs to go through voting phases by assurance committee members in the PBFT cycle. The function of the consensus box is called *csb* (Line 1 of Algorithm 5). There are initialize the height g and round d (Lines 2-3 of Algorithm 5). It presents as *block.locked* when there is a new preparing block, then block proposal begins the consensus loop. *LimitedStep* is considered a network problem. The valid block contains information about height, round, proposed block, and signature, so assurance committee members must check this signature in information whether it is valid in order to propose it (Lines 7 of Algorithm 5). Assurance committee members need to collect the count votes received from other assurance committee members (Lines 8-9 of Algorithm 5). This process also ensures that vote validity by verifying height, round, and signature. Eventually, if there are more than $\frac{2}{3}$ of votes for the locked block, it will process to Pre-commit. Otherwise, it will return to empty (Lines 10-13 of Algorithm 5). Assurance committee members broadcast their Pre-commit vote to the hash in the Prevote process, then there is work that performed vote count again (Lines 14 of Algorithm 5). If results are more than $\frac{2}{3}$ of votes from Pre-commit and hash matched with locked block, a proposed block can be committed as *new.blcok*, and block height will be appended (Lines 15-17 of Algorithm 5). In other words, a consensus is reached. Otherwise, no consensus is reached in this round, and it will start a new round (Lines 18-19 of Algorithm 5).

4.6 Security Discussion

At the high level, if any adversary wants to launch an effective attack (e.g. *double spend*) against PoAssu, an adversary needs to compromise at $\frac{1}{3}$ of the assurance areas in order to control or dominate assurance committee members, due to Byzantine Fault Tolerance theory [112]. In other words, honest miners never sign signatures twice for a block with the same height. However, the adversary can attack election process

Algorithm 5 Block Proposal with PBFT

```
1: function Consensus Box(csb)
2:     g ← csb.height + 1
3:     d ← 0
4:     ey ← none
5:     block.locked ← ey
6:     while d < LimitedStep do
7:         block.proposed ← Propose(csb, g, r, block.locked)
8:         block.locked ← Prevote(csb, g, r, Hash(block.proposed))
9:         votes(block.locked) ← count
10:        if votes(block.locked) ≤ 2/3 then
11:            empty ← return
12:        else
13:            Hash.block ← Precommit(csb, g, r, block.locked)
14:            votes(Hash.block) ← count
15:            if votes(Hash.block) > 2/3 & Hash.block = Hash(block.locked)
   then
16:                new.block ← Commit(csb, block.locked)
17:                g + +
18:            else
19:                r + +
20:            end if
21:        end if
22:    end while
23: end function
```

and control control more than $\frac{1}{3}$ assurance areas or directly compromising assurance committee members in order to ensure that the block that the adversary proposed is committed.

We assume that the majority of the participants in this protocol are honest. The probability of being elected as a committee member depends on their assurance degree, so the adversary should have the lowest scores of assurance degree with a certain number of leading 0 achieved. Based on assumption, we assume all the miners have the same hash power to simplify security model. The malicious nodes have the probability of ϕ to be elected across all compromised assurance areas. M is a sum of the number of assurance areas. For a node a, the probability of controlling block generation successfully presents in Equation 4.5:

$$1 - \sum_{a=0}^{[\frac{M-1}{3}]} C_{a-M}^{a}(\phi^a * (1-\phi)^{M-a}) \tag{4.5}$$

For 12 assurance areas, the adversary will have 50% probability of compromising the approach successfully, if malicious node has the probability of $\phi = 30\%$ in more than 3 assurance areas. But, an adversary needs to maintain ϕ in each assurance area in order to conduct an efficient attack for the system, so in reality, it is hard to maintain that.

4.6.1 Analysis of Eclipse Attack

In order to launch an Eclipse attack [114], the adversary needs to ensure all the network connections of target nodes are connected with compromised nodes or adversary nodes. The adversary will start the attack by sending its own IP address, flooding to target IP addresses (victim nodes). Then, the victim nodes may reconnect to IP addresses of nodes who are controlled by an adversary in order to isolate them from the network. Finally, adversaries send false information to victim nodes, preventing them to get real information from the network. The result of the Eclipse attack is to partition victim nodes from the network so that victim nodes are isolated. In order to launch a success attack, the adversary should control or deploy enough malicious nodes across most of assurances areas with significant financial support.

4.6.2 Analysis of Sybil Attack

In order to achieve Sybil attack [115], adversary needs to create plenty of fake or Sybil identities (malicious actors) in blockchain, then using these identities to win the voting in consensus. In this circumstance, these malicious actors are able to reject blocks of receiving or broadcasting in order to prevent other nodes from joining the network. Sybil attack is quite similar to an Eclipse attack, but an Eclipse attack is targeted at a single party; whereas a Sybil attack is network targeted. To launch a success attack, the adversary should control or deploy enough malicious nodes across most of assurances areas with significant financial support.

4.6.3 Analysis of Quantum Computing Attack

In our perspective, the biggest concerns are not either of Double Spend attack, Eclipse attack and Sybil attack or 51% tolerance in PoW or 30% tolerance in BFT-style based Blockchain. The biggest concern is Quantum Computing. Quantum computers, which substantially exceed traditional computing speed and data processing capacity, are gradually moving from theory toward practice. The tremendous computing power of quantum computers will bring fundamental challenges to the current information encryption mechanism [116]. In fact, Quantum Computing with Grover [117] and Shor [118] algorithms are the majority threats that brought into the blockchain.

Threat of Grover algorithm

The first threat of quantum supremacy is brought from the Grover algorithm, it is a searching algorithm which can significantly increase searching for Quantum of function inversion.

- Search Hash conflict: In Quantum computing, the Grover algorithm can increase searching hash conflict. Thus, the Grover algorithm inserts the modified block into the chain and at the same time, it does not affect the sequence consisting of the block. This type of attack modifies the authenticity of the block content while ensuring that the integrity of the blockchain does not conflict, and ultimately destroys the trustless environment of the blockchain.
- Reduce Mining Time: For quantum computers, the speed of generating new

blocks on the chain can far exceed the computing power of classic computers, which means that the time it takes to mine is much shorter than traditional computers. Therefore, miners with quantum computers can obtain more computing power. At the same time, if the random number (nonce) generation speed is fast enough, a new branch chain can be reconstructed in a very short time by using a quantum computer. When the length of the branch chain exceeds the main chain, the branch chain can replace the main chain and become a real one.

Threat of Shor algorithm

The second threat of quantum supremacy is brought from the Shor algorithm, it can be used to destroy the RSA encryption adopted by the blockchain [119]. Shor's algorithm can quickly find the two prime factors of a composite number. In the RSA encryption algorithm, the composite number will be used as the public key, and these two prime factors will be used as the private key. For classical computers, it is very difficult to factorize a composite number, but it is a simple task for quantum computers. Therefore, in the process of users verifying and exchanging public and private keys, the adversary can use Shor's algorithm to obtain and crack users' key pair (public and private keys), forging information and signatures. It means that any signed content by public or private key in the blockchain are able to be forged and passed to consensus verification . In addition, not only the transaction information between users will be attacked, but any encrypted communication used in the infrastructure of the blockchain will be attacked, and the reliability of communication encryption will be lost, and the blockchain environment will no longer be secure.

Countermeasure of Quantum Computing Attack

The detail for the countermeasure of Quantum computing attack is not discussed in this thesis, but we provide an ideal outline of methods to mitigate Quantum computing attack for blockchain in our approach or main blockchains for further works. In our perspective, there are following two methods, either one of them can mitigate Quantum computing attack:

- Employ Rainbow or Falcon signatures scheme instead of using ECDSA or Schnorr signature scheme.

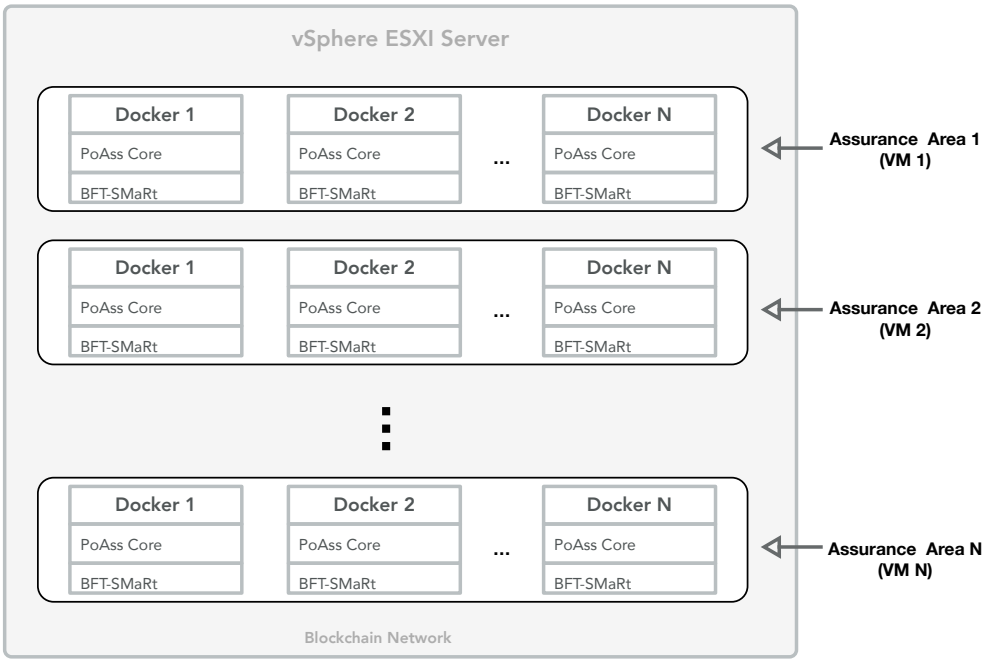


Figure 4.4: PoAssu deployment architecture

- Employ Quantum Communication for quantum key distribution (QKD) and quantum teleportation (QT) instead of traditional communication in blockchains.

4.7 Experiment

In this section, we deployed experiments to demonstrate PoAssu approach for performance. We deployed our approach using Python interacted with open-source project BFT-SMaRt [120]. The PoAssu architecture which is deployed in ESXI Server shown in Figure 4.4. We deploy vSphere ESXI Server 6.0 [121] as a fundamental virtualization system in a physical server, then we create virtual machines (VMs) and install Ubuntu Server OS to them, each VM can be represented one assurance area. Each virtual machine is running docker instances that contain PoAssu core algorithm interacted with BFT-SMaRt to represent PoAssu participates or nodes.

4.7.1 PoAssu with Blake3 for Per Puzzle

First, we tested the efficiency of PoAssu scheme with the Blake3 hash function for solving a puzzle by comparing it with the original Sha256 hash function scheme. Thus, we set two group experiments. In Group 1 (PoAssu with Blake3), we create

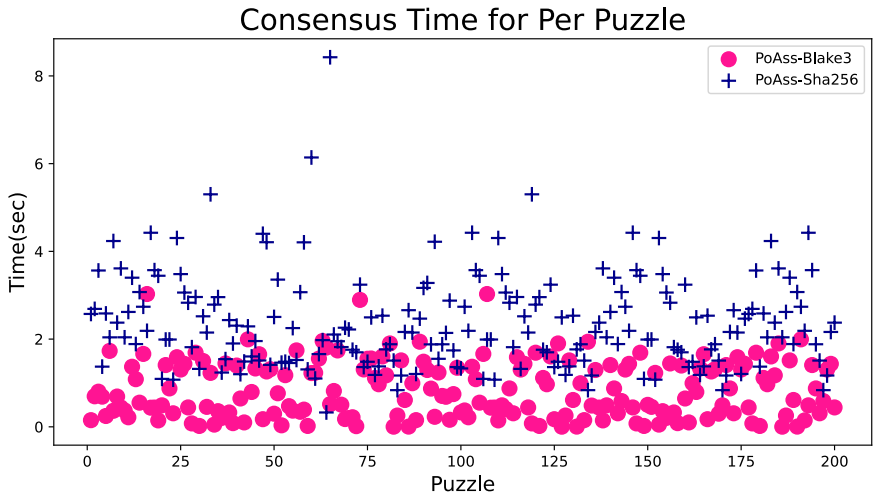


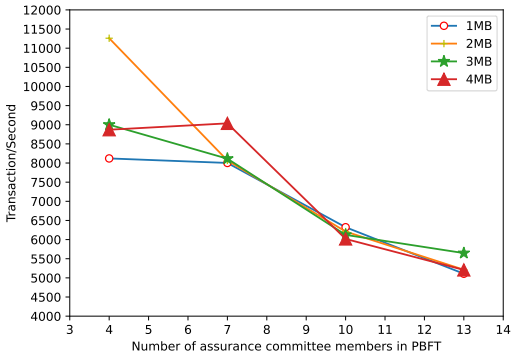
Figure 4.5: PoAssu for per puzzle

one assurance area with 10 docker instances as PoAssu participates in order to observe the time of solving assurance degree's puzzles. In Group 2 (PoAssu with Sha-256), we use the same docker instances to solving the same assurance degree of the puzzle. We recorded the time spent to solve puzzles and we run the experiment for 200 hash puzzles for each group.

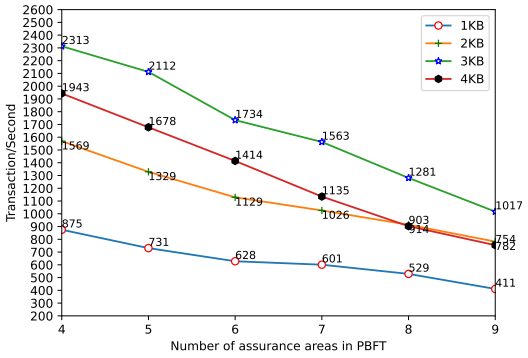
The times spent to achieve the proof of assurance for all 200 hash puzzles that is presented in Figure 4.5. It can be observed that the Group1 with Blake3 was faster than Group2 with Sha256. Thus, we suggest that using the Blake3 crypto hash function to replace the Sha256 hash function in blockchain platforms.

4.7.2 PoAssu simulation transaction throughput

In order to evaluate the performance of PoAssu proposal process in PBFT network, we start to create the number of nodes from 4 to 13. We also create several clients distributed across 2 additional machines in order to submit messages to nodes. These nodes are in the same LAN environment in order to communicate with each other. Then, size of each transaction is set as 256 Byte. After that, we test the impact on different block sizes from 1MB to 4MB for this PBFT environment. Moreover, we manually set one node as a fault for all the transaction throughput tests. The result is shown in Figure 4.6 (a). According to experiment results, transaction throughput dropped when the number of nodes increased. The result can be found that transaction throughput with a block size of 4MB is around 8.9K TPS when assurance



(a)



(b)

Figure 4.6: Transaction Throughput in PBFT

committee members are 4 nodes. Transaction throughput with a block size of 1MB is around 8K TPS when assurance committee members are 4 nodes. It shows that transaction throughput with a larger block size is higher than a smaller block size. This because a larger block size can contain more transactions than a smaller block, but it requires more time to handle blocks within PBFT environment. Thus, PoAssu can achieve best performance in transaction throughput when a network has 7 nodes with block size of 4MB. For a network has 10 nodes with block size of 1MB, PoAssu can achieve best performance in transaction throughput. For a network that has 13 nodes, block size of 3MB is more suitable in PoAss.

4.7.3 PoAssu Demonstration

We deploy the 9 virtual machines (VM) representing 9 assurance areas in vSphere ESXI Server as a demonstration. We run 4 dockers in each virtual machine, each

docker instance contain PoAssu core algorithm interacted with PBFT to represent PoAssu participates or nodes. To deploy PoAssu, we used BFT-SMaRt (open-source project) which is Byzantine fault-tolerant state machine replication to run PBFT between assurance committee members. We modified BFT-SMaRt, and using Python interface interacted BFT-SMaRt into PoAssu. In our current stage, committee members' Queue have not deployed in our experiment. Thus, there are a certain amount of delay that is used to confirm new committee members, once assurance committee members are going to change in the next round. Assurance committee members cache scheme is next goal of our future work. The current works, we just demonstrate the PoAssu's feasibility.

For transaction throughput test experiment in PoAssu, each assurance area has one more additional docker instance in order to represent as PoAssu client, and clients interact with PoAssu by the interface layer of PoAssu. PoAssu client sends transactions to PoAssu nodes and puts them on the chain in the blockchain. Firstly, we still set size of the transaction as 256 Byte. Then, we start our PBFT network from 4 to 9 assurance areas, and we set different block sizes between 1KB to 4KB for a PBFT network in order to observe impacts on transaction throughput. The result is presented in Figure 4.6 (b). It can be found that transaction throughput is automatically dropped when the number of assurance areas is increased. Block size is growing, transaction throughput increased. Because large block size can contain more transactions. However, it is required more time to process. Transaction throughput can achieve around 2.3K TPS for a block size of 3KB when there are 4 assurance areas, which is higher than other sizes of the block. Transaction throughput can achieve around 0.9K TPS for a block size of 2KB or 4KB when there are 8 assurance areas. The lowest transaction throughput is around 1.0K TPS and 0.4K TPS for a block size of 3KB and 1KB when the number of assurance areas is 9. Based on experiment, it spends 1.5 seconds to complete confirmation of assurance committee members in 4 assurance areas. It spends 5 seconds to complete confirmation of assurance committee members in 7 assurance areas, and 7 seconds in 9 assurance areas.

The transaction throughput difference between the PoAssu simulation and the demonstration PoAssu system can be explained as follows: (1)we did not use high-performance infrastructure for our implementation of the PoAssu system in the docker environment. (2)PoAssu system is not optimized when we implemented it. Instead of deployment native library for BFT in PoAssu simulation. (3)committee member

cache queue have not deployed in experiment. So, waiting time is required to complete confirmation of committee member across all the assurance areas before block generation. Thus, these limitations will be in our future works.

TPS, block size, and assurance areas

To further investigate the relationship between TPS, block size, and the number of assurance areas in our system, we build a multivariable linear regression model to analyze our experiment so that we can predict TPS with a suitable block size in our system. Then, we use the least squares approximation to find the solution of our built model. Furthermore, the relationship between dependent variable: y (TPS) and independent variables: $x_2, x_2, x_3, ...x_n$ can be presented in Equation (4.6). The w_0, w_1, w_2, w_n are regression coefficient in our model.

$$y = w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n \tag{4.6}$$

In Equation (4.7), m is a number of samples in the data set that needs to bring in the above Equation. After that, we use the matrix to covert our data in Equation (4.8). We can get the transformation matrix in Equation (4.9). Eventually, we can get Equation (4.10) for our model.

$$\begin{aligned} y_1 &= w_0 + w_1x_{11} + w_2x_{12} + ... + w_nx_{n1} \\ y_2 &= w_0 + w_1x_{21} + w_2x_{22} + ... + w_nx_{n2} \\ y_3 &= w_0 + w_1x_{31} + w_2x_{32} + ... + w_nx_{n3} \\ &... \\ y_m &= w_0 + w_1x_{n1} + w_2x_{n2} + ... + w_nx_{nm} \end{aligned} \tag{4.7}$$

$$\begin{bmatrix} y_1 \\ y_1 \\ . \\ . \\ . \\ y_m \end{bmatrix} = w_0 + w_1 \begin{bmatrix} x_{11} \\ x_{12} \\ . \\ . \\ . \\ x_{1m} \end{bmatrix} + w_2 \begin{bmatrix} x_{21} \\ x_{22} \\ . \\ . \\ . \\ x_{2m} \end{bmatrix} + ...w_n \begin{bmatrix} x_{n1} \\ x_{n2} \\ . \\ . \\ . \\ x_{nm} \end{bmatrix} \tag{4.8}$$

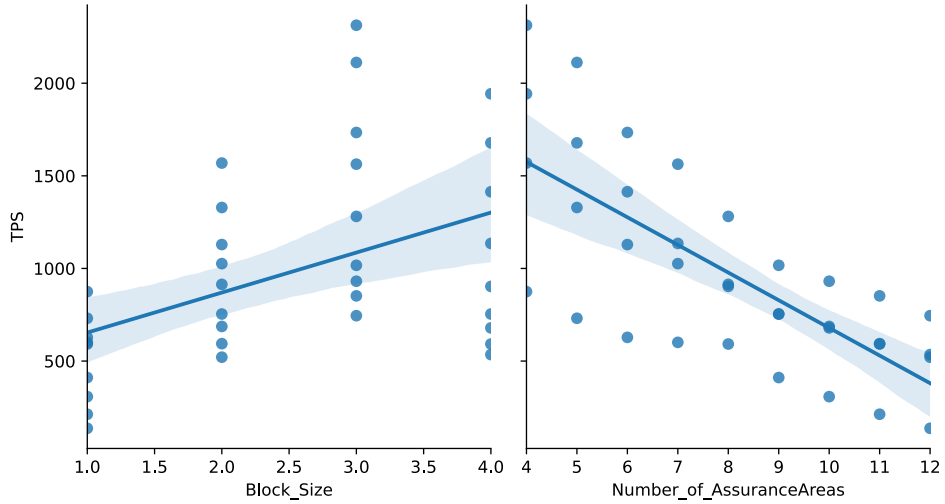


Figure 4.7: Result of multivariable linear regression model

$$\vec{y} = \begin{bmatrix} y_1 \\ y_1 \\ . \\ . \\ . \\ y_m \end{bmatrix}, X = \begin{bmatrix} 1 & x_{11}...x_{n1} \\ 1 & x_{12}...x_{n2} \\ . & \\ . & \\ . & \\ 1 & x_{1m}...x_{nm} \end{bmatrix}, \vec{w} \begin{bmatrix} w_0 \\ w_1 \\ . \\ . \\ . \\ w_n \end{bmatrix} \tag{4.9}$$

$$\vec{y} = X\vec{w} \tag{4.10}$$

According to our equations, we can find the solution of the regression coefficient for $\vec{w}$ in Equation 4.11. X^T represents the transpose of a matrix X, -1 of $(X^TX)^{-1}$ represents inverse matrix for that matrix.

$$\vec{w} = (X^TX)^{-1}X^T\vec{y} \tag{4.11}$$

We used our experiment data from the demonstration PoAssu system and bring them to our model. Figure 4.7 shows the experiment result of multivariable linear regression model. We find solution of w_0, w_1 and w_2 and our multivariable linear regression model can be presented in Equation (4.12)

$$y = 713.7111 * x_1 + (-88.9333 * x_2) + 497.3111 \tag{4.12}$$

In Equation (4.12), y is represented as predict TPS in our system, x_1 is represented as block size, x_2 is represented as the number of assurance areas. Thus, for given any block size and the number of assurance areas we are able to predict TPS in our system. Moreover, we also calculate coefficient of determination (R^2) of our model. The coefficient of determination is the proportion of the dependent variable's variance that can be predicted from the independent variable [122].

A data set has n values marked $z_1, ...z_n$ (z_i or vector $\vec{z} = [z_1, ..., z_n]^T$), each associated with a fitted value $g_1, ..., g_n$ (g_i or vector $\vec{g}$). Then, we can employ the residuals in Equation (4.13).

$$\vec{e_i} = z_i - gi \tag{4.13}$$

If $\bar{z}$ is the mean of the observed data, then $\bar{z}$ can be presented in Equation (4.14),

$$\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i \tag{4.14}$$

then the variability of the data set are able to measure by two sums of squares formulas in Equation (4.15). The sum of squares of residuals can be present in Equation (4.16):

$$SS_{tot} = \sum_i (z_i - \bar{z})^2 \tag{4.15}$$

$$SS_{res} = \sum_i (z_i - g_i)^2 = \sum_i e_i^2 \tag{4.16}$$

Eventually, the definition of the coefficient of determination can be presented in Equation (4.17). If the value of the coefficient of determination (R^2) equal to 1. It indicates that the fitted model explains all variability in y, and the modeled values exactly match the observed values. Thus, the value of the coefficient of determination (R^2) is more closed to 1 which has the goodness of fit for predictive data [123]. We calculated R^2 in our model that is 0.9911632594186066. It means our model has the goodness of fit for predictive data. In our current model, we only considered block size and the number of assurance areas as factors to impact TPS. However, in reality, many factors can impact results in the blockchain system, such as network delay. It is our future work to optimize our model.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{4.17}$$

TPS under different consensus approaches

To compare TPS under the PoW, PoS, PoAssu, and Raft consensus approaches, we define the same transaction size and block size for them. Then, we set up 5 groups (Group 1: PoW-Bitcoin, Group 2: PoW-Ethereum, Group 3: PoS-Nxt, Group 4: PoAssu, Group 5: Raft), and each group has the same 16 docker instances as nodes. We use Bitcoin and Ethereum applications as the typical PoW approach. Thus, TPS value of PoW approach is tested by counting block data in Ethereum and Bitcoin. Detailed of transaction data are counted by using the RPC interface to get blockchain information. TPS average is calculated by The number of transactions in each block and newly generated blocks. TPS value of Bitcoin is 7 and Ethereum is 10. We use Nxt application as the typical PoS approach. The test network is implemented to get the number of transactions in the block in Nxt. TPS value of Nxt is 51. We take advantage from simpleRaft (open-source) [124] as Raft approach. The TPS of the Raft approach is around 1492. The TPS values of various approaches are presented in Figure 4.7. The Raft has closed performance to PoAssu, however, Raft approach does not provide fault tolerance in system. Thus, our PoAssu provides better performance in TPS.

4.8 Summary

In this Chapter, we presents a Proof of Assurance based consensus approach, which is a hybrid approach that describes an assurance election scheme and combines it with PBFT verification. There are some outstanding issues in mainstream blockchain consensus protocols. PoAssu is able to achieve a high transaction throughput. The 9.0K TPS throughput demonstrates that PoAssu simulation transaction throughput are to work with the high-demanded trading market. We also build a multivariable linear regression model to analyse and predict TPS between block size and the number of assurance areas.

This chapter mainly answers the Research Question 2 mentioned in Chapter 1. The conclusions of the thesis and future research directions will be given in the next chapter.

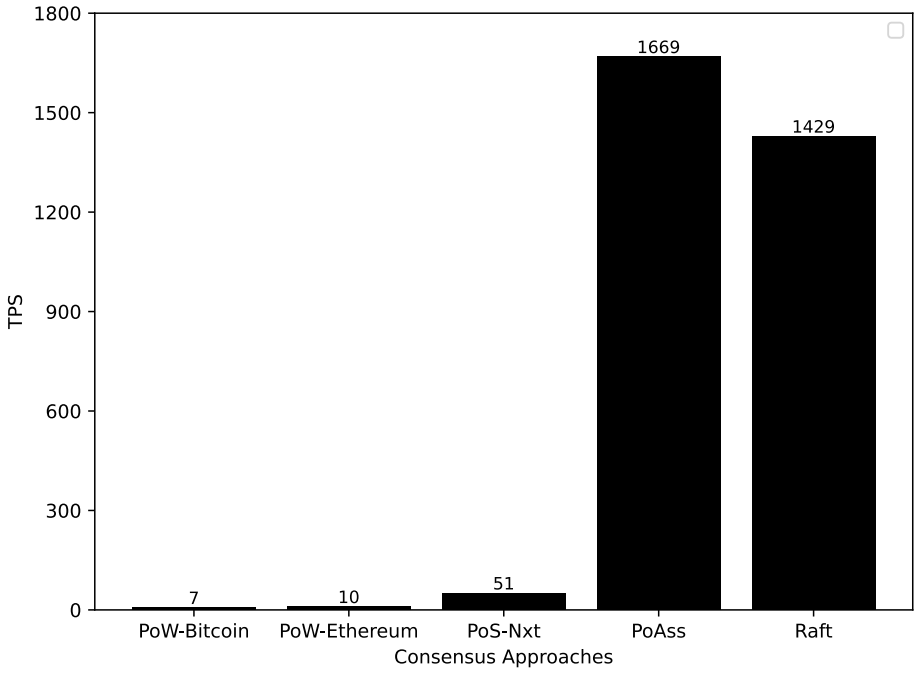


Figure 4.8: TPS under different consensus approaches

Chapter 5

Conclusion

This chapter concludes the thesis and addresses the limitations and future work directions. The thesis contributions are summarised in Section 5.1. The limitations of the approaches and possible directions for future work are discussed in Section 5.2.

5.1 Summary of Thesis Contribution

This thesis contributes to the field from the following three aspects.

5.1.1 Blockchain Literature Review

I provided reviews for most commonly used consensus approaches in blockchain systems, and provided landscape into two important components in Chapter 2: candidateship election and consensus mechanism. The candidateship election determines a committee or leader of nodes that participate in the consensus; the consensus mechanism is responsible for deciding on the next block, run by the elected committee or leader of nodes.

5.1.2 Proof of SecureStake based blockchain consensus approach

I proposed a Proof of SecureStake (PoSS) based blockchain consensus approach, focusing on trustworthiness and performance as multiple criteria for election scheme. PoSS is that we are not only using token as single "stake" as the measurement for nodes participating over the block generation, but also we take node's multiple criteria as the comprehensive performance (e.g., CPU core and amount of memory) into a

score for consideration. The Concentration of wealth that conducts "wealthy" nodes have full control over the block generation process is mitigated by PoSecureStake. Election process is predictable, which brings vulnerability for guessing attacks that can be mitigate by PoSS.

5.1.3 Proof of Assurance based blockchain consensus approach

I proposed a Proof of Assurance (PoAssu) based blockchain consensus approach, focusing on transaction throughput. In order to achieve that, I construct a novel approach which combines the PoW and PBFT algorithms so that forks can be avoided. PoAssu can allow transactions to be confirmed immediately, achieving high transaction throughput in our system. PoAssu is able to improve efficiency of high deemed trading environment. I also proposed a multivariable linear regression model to further analyse the relationship between TPS, block size, and the number of assurance areas, and this model also can be used to predict TPS in our approach in order to make PoAssu more adaptive and comprehensive itself.

5.2 Limitations and Future Work

In my Master's study, I developed a basic consensus approach for comprehensive blockchain consensus mechanisms. There are various potential directions to investigate blockchain systems and make improvements and future work for proposed approaches.

- PoSS presented in Chapter 3, nodes are mapped into different resource pools by k-Medoids clustering function. The time complexity is high in this function which conducts slow performance for mapping new nodes to resource pools. Thus, during the dynamic network, it requires lower time to partition resource pool and map nodes into pools. In future work, I would like to dive deeper into this problem and aim at developing a more efficient function to mitigate this problem.
- PoAssu developed in Chapter 4, PoAssu simulation of transaction throughput is able to work with the high-demanded trading market, and a multivariable linear regression model is built in our system. However, there are a lot of works to improve our PoAssu. Firstly, in the future, we will optimise architecture

of PoAssu and implement PoAssu in a production environment. Secondly, we can employ the reputation model for election process which allows to increase the difficulty of compromising committee members, and it can enhence security. Thirdly, we will optimize our multivariable linear regression model by considering more factors into the model.

Besides, in the studies of blockchain systems, I noticed that the fundamental of blockchain is cryptographic, however, the tremendous computing power of quantum computers will bring fundamental challenges to the current information encryption mechanism. Thus, another further work is to investigate how to effectively mitigate quantum computing attacks for blockchain systems.

BIBLIOGRAPHY

[1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[2] Eleftherios Kokoris Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. Enhancing bitcoin security and performance with strong consistency via collective signing. In *25th {usenix} security symposium ({usenix} security 16)*, pages 279–296, 2016.

[3] Colin LeMahieu. Raiblocks: A feeless distributed cryptocurrency network. *URL https://raiblocks.net/media/RaiBlocks_ Whitepaper_ _English.pdf*, 2017.

[4] Stuart Haber and W Scott Stornetta. How to time-stamp a digital document. In *Conference on the Theory and Application of Cryptography*, pages 437–455. Springer, 1990.

[5] Fred B Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys (CSUR)*, 22(4):299–319, 1990.

[6] Leslie Lamport. Password authentication with insecure communication. *Communications of the ACM*, 24(11):770–772, 1981.

[7] Gavin Wood et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151(2014):1–32, 2014.

[8] Elad Elrom. Neo blockchain and smart contracts. In *The Blockchain Developer*, pages 257–298. Springer, 2019.

[9] Wiebe Koerhuis, Tahar Kechadi, and Nhien-An Le-Khac. Forensic analysis of privacy-oriented cryptocurrencies. *Forensic Science International: Digital Investigation*, page 200891, 2020.

[10] Ren Zhang and Bart Preneel. Lay down the common metrics: Evaluating proof-of-work consensus protocols' security. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 175–192. IEEE, 2019.

[11] Ghassan Karame and Wenting Li. Blockchain beyond bitcoin by ghassan karame and wenting li.

[12] Shai Halevi and Hugo Krawczyk. Public-key cryptography and password protocols. *ACM Transactions on Information and System Security (TISSEC)*, 2(3):230–268, 1999.

[13] Bitcoin Wiki. Elliptic curve digital signature algorithm. *Elliptic curve digital signature algorithm. See https://en. bitcoin. it/wiki/Elliptic_ Curve_ Digital_ Signature_ Algorithm*, 2017.

[14] Gideon Greenspan. Multichain private blockchain-white paper. *URl: http://www. multichain. com/download/MultiChain-White-Paper. pdf*, 2015.

[15] Walter L Heimerdinger and Charles B Weinstock. A conceptual framework for system fault tolerance. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1992.

[16] John R Douceur. The sybil attack. In *International workshop on peer-to-peer systems*, pages 251–260. Springer, 2002.

[17] Marko Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. bft replication. In *International workshop on open problems in network security*, pages 112–125. Springer, 2015.

[18] Christopher Natoli, Jiangshan Yu, Vincent Gramoli, and Paulo Esteves-Verissimo. Deconstructing blockchains: A comprehensive survey on consensus, membership and structure. *arXiv preprint arXiv:1908.08316*, 2019.

[19] Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The honey badger of bft protocols. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 31–42, 2016.

[20] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Annual International Cryptology Conference*, pages 139–147. Springer, 1992.

[21] Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August*, 19:1, 2012.

[22] Nicolas Houy. It will cost you nothing to "kill" a proof-of-stake crypto-currency. *Available at SSRN 2393940*, 2014.

[23] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[24] Christian Cachin et al. Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, volume 310, 2016.

[25] Harish Sukhwani, José M Martínez, Xiaolin Chang, Kishor S Trivedi, and Andy Rindos. Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 253–255. IEEE, 2017.

[26] Session management owasp. URL https://www.blockchain.com/btc/block/.

[27] Theodore Garland Jr. The scientific method as an ongoing process. *U C Riverside. Archived from the original on 19 August 2016*, 2016.

[28] Theodore Garland Jr. The scientific method as an ongoing process. *U C Riverside. Archived from the original on 19 August 2016*, 2016.

[29] Lina Alsahan, Noureddine Lasla, and Mohamed Abdallah. Local bitcoin network simulator for performance evaluation using lightweight virtualization. In *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*, pages 355–360. IEEE, 2020.

[30] Machwork. URL https://www.mathworks.com/discovery.

[31] Michael Myer, Maria L Paget, and Robert D Lingard. Performance of t12 and t8 fluorescent lamps and troffers and led linear replacement lamps caliper benchmark report. Technical report, Pacific Northwest National Lab.(PNNL), Richland, WA (United States), 2009.

[32] Aad van Moorsel. Benchmarks and models for blockchain: consensus algorithms. *ACM SIGMETRICS Performance Evaluation Review*, 46(3):113–113, 2019.

[33] Wattana Viriyasitavat and Danupol Hoonsopon. Blockchain characteristics and consensus in modern business processes. *Journal of Industrial Information Integration*, 13:32–39, 2019.

[34] Hyungmin Cho. Asic-resistance of multi-hash proof-of-work mechanisms for blockchain consensus protocols. *IEEE Access*, 6:66210–66222, 2018.

[35] Iddo Bentov, Ariel Gabizon, and Alex Mizrahi. Cryptocurrencies without proof of work. In *International conference on financial cryptography and data security*, pages 142–157. Springer, 2016.

[36] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[37] Adam Back et al. Hashcash-a denial of service counter-measure. 2002.

[38] Cynthia Dwork, Andrew Goldberg, and Moni Naor. On memory-bound functions for fighting spam. In *Annual International Cryptology Conference*, pages 426–444. Springer, 2003.

[39] Martin Abadi, Mike Burrows, Mark Manasse, and Ted Wobber. Moderately hard, memory-bound functions. *ACM Transactions on Internet Technology (TOIT)*, 5(2):299–327, 2005.

[40] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM (JACM)*, 35(2):288–323, 1988.

[41] Nicolas Van Saberhagen. Cryptonote v 2.0, 2013.

[42] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE, 2014.

[43] Harris Brakmić. What is bitcoin? In *Bitcoin and Lightning Network on Raspberry Pi*, pages 3–31. Springer, 2019.

[44] Alex Biryukov and Dmitry Khovratovich. Equihash: Asymmetric proof-of-work based on the generalized birthday problem (full version). Technical report, Cryptology ePrint Archive: Report 2015/946. Last revised October 27, 2016 âĂę, 2020.

[45] Hanna Halaburda and Neil Gandal. Competition in the cryptocurrency market. *updated version published as" Can we predict the winner in a market with network effects*, pages 14–17, 2016.

[46] Yujin Kwon, Jian Liu, Minjeong Kim, Dawn Song, and Yongdae Kim. Impossibility of full decentralization in permissionless blockchains. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, pages 110–123, 2019.

[47] Lars Brünjes, Aggelos Kiayias, Elias Koutsoupias, and Aikaterini-Panagiota Stouka. Reward sharing schemes for stake pools. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 256–275. IEEE, 2020.

[48] Wei Dai. B-money. *Consulted*, 1:2012, 1998.

[49] Bitcoin Forum Proof of Stake instead of Proof of Work. Cryptonote v 2.0. *URL https://bitcointalk.org/index.php?topic=27787.0*, 2011.

[50] Nxt Wiki. Whitepaper: nxtâĂŤnxt wiki. 2016.

[51] Iddo Bentov, Ariel Gabizon, and Alex Mizrahi. Cryptocurrencies without proof of work. In *International conference on financial cryptography and data security*, pages 142–157. Springer, 2016.

[52] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In *Annual International Cryptology Conference*, pages 357–388. Springer, 2017.

[53] D Larimer. Delegated proof-of-stake white paper. *2014 http://www.bts.hk/dpos-baipishu.html*, 2014.

[54] Giang-Truong Nguyen and Kyungbaek Kim. A survey about consensus algorithms used in blockchain. *Journal of Information processing systems*, 14(1), 2018.

[55] Sergei Tikhomirov. Ethereum: state of knowledge and research perspectives. In *International Symposium on Foundations and Practice of Security*, pages 206–221. Springer, 2017.

[56] Runchao Han, Zhimei Sui, Jiangshan Yu, Joseph Liu, and Shiping Chen. Fact and fiction: Challenging the honest majority assumption of permissionless blockchains. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 817–831, 2021.

[57] Jae Kwon. Tendermint: Consensus without mining. *Draft v. 0.6, fall*, 1(11), 2014.

[58] Gavin Wood. Proof-of-authority private chains, 2015.

[59] Stefan Dziembowski, Sebastian Faust, Vladimir Kolmogorov, and Krzysztof Pietrzak. Proofs of space. In *Annual Cryptology Conference*, pages 585–605. Springer, 2015.

[60] Giacomo Brambilla, Michele Amoretti, and Francesco Zanichelli. Using blockchain for peer-to-peer proof-of-location. *arXiv*, pages arXiv–1607, 2016.

[61] Debayan Gupta, Benjamin Mood, Joan Feigenbaum, Kevin Butler, and Patrick Traynor. Using intel software guard extensions for efficient two-party secure function evaluation. In *International Conference on Financial Cryptography and Data Security*, pages 302–318. Springer, 2016.

[62] Architecure ARM. Security technology building a secure system using trustzone technology (white paper). *ARM Limited*, 2009.

[63] Saidalavi Kalady, PG Dileep, Krishanu Sikdar, BS Sreejith, Vinaya Surya, and P Ezudheen. Implementation of a purely hardware-assistedvmm for x86 architecture. In *Proceedings of the World Congress on Engineering*, volume 1, 2009.

[64] Jiangshan Yu, David Kozhaya, Jeremie Decouchant, and Paulo Esteves-Verissimo. Repucoin: Your reputation is your power. *IEEE Transactions on Computers*, 68(8):1225–1237, 2019.

[65] Bin Yu, Joseph Liu, Surya Nepal, Jiangshan Yu, and Paul Rimba. Proof-of-qos: Qos based blockchain consensus protocol. *Computers & Security*, 87:101580, 2019.

[66] Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. Hotstuff: Bft consensus in the lens of blockchain. *arXiv preprint arXiv:1803.05069*, 2018.

[67] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th symposium on operating systems principles*, pages 51–68, 2017.

[68] Dan Boneh, Saba Eskandarian, Lucjan Hanzlik, and Nicola Greco. Single secret leader election. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*, pages 12–24, 2020.

[69] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OSDI*, volume 99, pages 173–186, 1999.

[70] IBM. Hyperledgersawtooth. URL https://www.hyperledger.org/category/hyperledger-sawtooth, 2020.

[71] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. In *Concurrency: the Works of Leslie Lamport*, pages 203–226. 2019.

[72] Litecoin. Litecoin. *URL https: //litecoin.info/index.php/Litecoin*, 2011.

[73] Elaine Shi. Analysis of deterministic longest-chain protocols. In *2019 IEEE 32nd Computer Security Foundations Symposium (CSF)*, pages 122–12213. IEEE, 2019.

[74] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 507–527. Springer, 2015.

[75] Joao Sousa, Alysson Bessani, and Marko Vukolic. A byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In *2018 48th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pages 51–58. IEEE, 2018.

[76] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)*, 20(4):398–461, 2002.

[77] June Ma, Joshua S Gans, and Rabee Tourky. Market structure in bitcoin mining. Technical report, National Bureau of Economic Research, 2018.

[78] Rui Qin, Yong Yuan, Shuai Wang, and Fei-Yue Wang. Economic issues in bitcoin mining and blockchain research. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 268–273. IEEE, 2018.

[79] Suhyeon Lee and Seungjoo Kim. Proof-of-stake at stake: predatory, destructive attack on pos cryptocurrencies. In *Proceedings of the 3rd Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, pages 7–11, 2020.

[80] Suhyeon Lee and Seungjoo Kim. Short selling attack: A self-destructive but profitable 51% attack on pos blockchains. *IACR Cryptol. ePrint Arch.*, 2020:19, 2020.

[81] Critique of buterin's "a proof of stake design philosophy" by tuur demeester medium. URL https://tuurdemeester.medium.com/critique-of-buterins-a-proof-of-stake-design-philosophy-49fc9ebb36c6.

[82] David Shrier, Weige Wu, and Alex Pentland. Blockchain & infrastructure (identity, data security). *Massachusetts Institute of Technology-Connection Science*, 1(3):1–19, 2016.

[83] JA Núñez, PM Cincotta, and FC Wachlin. Information entropy. In *Chaos in Gravitational N-Body Systems*, pages 43–53. Springer, 1996.

[84] Ml|k-medoids clustering. URL https://www.geeksforgeeks.org/.

[85] Weiguo Sheng and Xiaohui Liu. A genetic k-medoids clustering algorithm. *Journal of Heuristics*, 12(6):447–466, 2006.

[86] Machine learning on big data. URL http://machinelearningbigdata.blogspot.com.

[87] Pranava Madhyastha and Rishabh Jain. On model stability as a function of random seed. *arXiv preprint arXiv:1909.10447*, 2019.

[88] Joanna Moubarak, Eric Filiol, and Maroun Chamoun. On blockchain security and relevant attacks. In *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)*, pages 1–6. IEEE, 2018.

[89] Docker whitepaper. *URL https://docker.com*, 2020.

[90] Rancher. *https://rancher.com*, 2020.

[91] Alysson Bessani, Joao Sousa, and Eduardo EP Alchieri. State machine replication for the masses with bft-smart. In *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 355–362. IEEE, 2014.

[92] Patrick McCorry, Malte Mser, Siamak F Shahandasti, and Feng Hao. Towards bitcoin payment networks. In *Australasian Conference on Information Security and Privacy*, pages 57–76. Springer, 2016.

[93] Dejan Vuji, Dijana Jagodi, and Sini Ran. Blockchain technology, bitcoin, and ethereum: A brief overview. In *2018 17th international symposium infoteh-jahorina (infoteh)*, pages 1–6. IEEE, 2018.

[94] Yonatan Sompolinsky and Aviv Zohar. Secure high-rate transaction processing in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 507–527. Springer, 2015.

[95] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. Bitcoin-ng: A scalable blockchain protocol. In *13th {USENIX} symposium on networked systems design and implementation ({NSDI} 16)*, pages 45–59, 2016.

[96] Yonatan Sompolinsky and Aviv Zohar. Phantom. *IACR Cryptology ePrint Archive, Report 2018/104*, 2018.

[97] Chenxing Li, Peilun Li, Dong Zhou, Wei Xu, Fan Long, and Andrew Yao. Scaling nakamoto consensus to thousands of transactions per second. *arXiv preprint arXiv:1805.03870*, 2018.

[98] Eleftherios Kokoris Kogias, Philipp Jovanovic, Nicolas Gailly, Ismail Khoffi, Linus Gasser, and Bryan Ford. Enhancing bitcoin security and performance with strong consistency via collective signing. In *25th {usenix} security symposium ({usenix} security 16)*, pages 279–296, 2016.

[99] Ittai Abraham, Dahlia Malkhi, Kartik Nayak, Ling Ren, and Alexander Spiegelman. Solida: A blockchain protocol based on reconfigurable byzantine consensus. *arXiv preprint arXiv:1612.02916*, 2016.

[100] Rafael Pass and Elaine Shi. Hybrid consensus: Efficient consensus in the permissionless model. In *31st International Symposium on Distributed Computing (DISC 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[101] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th Symposium on Operating Systems Principles*, pages 51–68, 2017.

[102] Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, and Prateek Saxena. A secure sharding protocol for open blockchains. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 17–30, 2016.

[103] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.

[104] Adam Back et al. Hashcash-a denial of service counter-measure. 2002.

[105] Danilo Gligoroski. Length extension attack on narrow-pipe sha-3 candidates. In *International Conference on ICT Innovations*, pages 5–10. Springer, 2010.

[106] Blake3-specs/blake3.pdf at master blake3 team/blake3 specs github. URL https://github.com/BLAKE3-team/BLAKE3-specs/blob/master/blake3.pdf.

[107] Alireza Beikverdi and JooSeok Song. Trend of centralization in bitcoin's distributed network. In *2015 IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 1–6. IEEE, 2015.

[108] Block: 277316 | blockchain explorer. URL https://www.blockchain.com/btc/block/.

[109] Block: 0 | blockchain explorer. URL https://www.blockchain.com/btc/block/.

[110] Block: 501509 | blockchain explorer. URL https://www.blockchain.com/btc/block/.

[111] David L Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on communications*, 39(10):1482–1493, 1991.

[112] Harish Sukhwani, Jos M Martnez, Xiaolin Chang, Kishor S Trivedi, and Andy Rindos. Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric). In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 253–255. IEEE, 2017.

[113] Guangquan Xu, Yihua Liu, Jun Xing, Tao Luo, Yonghao Gu, Shaoying Liu, Xi Zheng, and Athanasios V Vasilakos. Sg-pbft: a secure and highly efficient blockchain pbft consensus algorithm for internet of vehicles. *arXiv preprint arXiv:2101.01306*, 2021.

[114] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. Eclipse attacks on bitcoin's peer-to-peer network. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*, pages 129–144, 2015.

[115] Haifeng Yu, Michael Kaminsky, Phillip B Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 267–278, 2006.

[116] Tiago M Fernndez Carams and Paula Fraga Lamas. Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks. *IEEE Access*, 8:21091–21116, 2020.

[117] PG Kwiat, JR Mitchell, PDD Schwindt, and AG White. Grover's search algorithm: an optical approach. *Journal of Modern Optics*, 47(2-3):257–266, 2000.

[118] Ben P Lanyon, Till J Weinhold, Nathan K Langford, Marco Barbieri, Daniel FV James, Alexei Gilchrist, and Andrew G White. Experimental demonstration of a compiled version of shora's algorithm with quantum entanglement. *Physical Review Letters*, 99(25):250505, 2007.

[119] Munmun Saha, Sanjaya Kumar Panda, and Suvasini Panigrahi. Distributed computing security: issues and challenges. *Cyber security in parallel and distributed computing: concepts, techniques, applications and case studies*, pages 129–138, 2019.

[120] Alysson Bessani, Joao Sousa, and Eduardo EP Alchieri. State machine replication for the masses with bft-smart. In *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 355–362. IEEE, 2014.

[121] Forbes Guthrie, Scott Lowe, and Kendrick Coleman. *VMware vSphere design*. John Wiley & Sons, 2013.

[122] Laureano Gonzalez-Vega and Ioana Necula. Efficient topology determination of implicitly defined algebraic plane curves. *Computer aided geometric design*, 19(9):719–743, 2002.

[123] Scott Menard. Coefficients of determination for multiple logistic regression analysis. *The American Statistician*, 54(1):17–24, 2000.

[124] simpleraft: Raft in pure python. URL https://github.com/streed/simpleRaft.