Security Analysis of Cryptocurrencies in Consensus Algorithm

Sun Haokun

(UID: 3035232987)

Apr 14th, 2019
Abstract

The invention of Cryptocurrency is considered one of the greatest virtualization process in the 21st century. The distributed tamper-evident database it adopts makes a great example of decentralization. To maintain the blockchain that makes this database reliable, the study and security analysis of the consensus algorithm is crucial.

This paper is a progress report of project ‘Security Analysis of Cryptocurrencies in Consensus Algorithm’. It provides the general background of consensus algorithm and focuses on a newly-proposed hybrid algorithm called ‘Algorand’. The first two chapters will introduce the universal mechanism of cryptocurrencies to the audience and then move onto the introduction of consensus. The latter part of the paper will illustrate in what way Algorand is innovative. An improvement on the cryptographic sortition algorithm in Algorand will be proposed. A detailed comparative study between Algorand and POW will also be presented to demonstrate the point.
Acknowledgement

I would like to express my most sincere appreciation to all those who have helped me in this project. A special gratitude I give to my final year project supervisor, Dr John Yuen, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project especially in writing this report.

Furthermore I would also like to acknowledge with much appreciation the crucial role of Zhang Qiping, who helped me with mathematical support necessary to complete the computation. A special thanks goes to my CAES lecturer, Ms Mable Choi, who has been patient and caring in guiding me through the writing and presentation.
# List of Tables

## Chapter 2

2.1 Consensus Algorithms and Cryptocurrencies on Permissionless Platform...........8

2.2 Consensus Algorithms on Permissioned Platform...........................................9

## Chapter 6

6.1 VRF makeup..............................................................................................13
List of Figures

Chapter 1

1.1 Time-series Components of Cryptocurrency Market………………………………………1
1.2 Historical market price of Bitcoin……………………………………………………………2

Chapter 2

2.1 Cryptographic hash process…………………………………………………………………5

Chapter 6

6.1 Verifiable Random Function Overview…………………………………………………..22
6.2 Proof generating Overview………………………………………………………………..22
6.3 Detailed proof generating……………………………………………………………………22
6.4 Verification Overview………………………………………………………………………23
6.5 Detailed Verification…………………………………………………………………………23
6.6 The cryptographic sortition algorithm……………………………………………………24
6.7 Running time of cryptographic sortition using binomial distribution…………………25
6.8 Running time of cryptographic sortition using normal distribution…………………26
6.9 Timeout mechanism in voting non-empty block………………………………………27
6.10 Timeout mechanism for voting empty block……………………………………………28
6.11 Hard Forking: Non-upgraded nodes rejecting new rule……………………………29
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BA</td>
<td>Byzantine Agreement</td>
</tr>
<tr>
<td>BFT</td>
<td>Byzantine Fault Tolerance</td>
</tr>
<tr>
<td>CBOE</td>
<td>Chicago Board Options Exchange</td>
</tr>
<tr>
<td>CME</td>
<td>Chicago Mercantile Exchange</td>
</tr>
<tr>
<td>DPOS</td>
<td>Delegated Proof of Stake</td>
</tr>
<tr>
<td>POA</td>
<td>Proof of Authority</td>
</tr>
<tr>
<td>POS</td>
<td>Proof of Stake</td>
</tr>
<tr>
<td>POW</td>
<td>Proof of Work</td>
</tr>
<tr>
<td>VRF</td>
<td>Verifiable Random Functions</td>
</tr>
</tbody>
</table>
# Table of Contents

Abstract .......................................................................................................................... II

Aknowledgement ............................................................................................................. III

List of Tables .................................................................................................................... IV

List of Figures ................................................................................................................... V

List of Abbreviations ....................................................................................................... VI

Chapter 1 introduction ........................................................................................................ 1

1.1 Background ................................................................................................................. 1

1.2 Development History Overview ................................................................................. 1

1.3 Blockchain Overview .................................................................................................. 3

1.4 Scope of Work .............................................................................................................. 3

1.5 Outline of Interim Report ........................................................................................ 4

Chapter 2 Background Theorem ....................................................................................... 5

2.1 Cryptocurrency Mechanism Overview ........................................................................ 5

2.2 Consensus ................................................................................................................... 6

2.2.1 The Byzantine Generals Problem ......................................................................... 6

2.2.2 Byzantine Default Tolerance ................................................................................. 7

2.2.3 Blockchain Platform .......................................................................................... 7

2.2.4 Consensus Algorithm ......................................................................................... 7

Chapter 3 Objective ......................................................................................................... 10

Chapter 4 Methodology .................................................................................................... 11

4.1 Introduction ................................................................................................................. 11

4.2 Qualitative Evaluation ............................................................................................ 11

4.3 Benchmark Selection ............................................................................................... 11
Chapter 5 Comparative Case study ................................................................. 15
  5.1 Study on Algorand .................................................................................. 15
      5.1.1 Cryptographic Sortition ................................................................ 16
      5.1.2 Byzantine Agreement .................................................................... 17
  5.2 Comparative Study between POW and Algorand .................................... 19

Chapter 6 Result ............................................................................................ 21
  6.1 Study on Algorand .................................................................................. 21
      6.1.1 Cryptographic Sortition ................................................................ 21
          6.1.1.1 Overview ............................................................................. 21
          6.1.1.2 Details of Cryptographic sortition ........................................ 23
              6.1.1.2.1 Normal distribution approximation .................................. 25
      6.1.2 Byzantine Agreement .................................................................... 27
  6.2 Comparative Study between POW and Algorand .................................... 29

Chapter 7 Conclusion .................................................................................... 31
  7.1 Summarization ....................................................................................... 31
  7.2 Future Study .......................................................................................... 31

References ....................................................................................................... 32
Chapter 1. Introduction

1.1 Background

Technology updates have facilitated the virtualization process of financial dealings. Money and ledgers have become increasingly intangible. It has been predicted that in the coming decade, over 80% of the currencies will only exist as entries in the ledger. Despite the supposedly optimistic estimation, transactional operations at this time are still mostly processed by financial institutions acting as a third party. These financial institutions are typically in an advantageous position in occupying resources and manipulating the capital of clients on behalf of their own benefits. They also share privileges to access the insider information, including interest rate flotation, deposit-loan equilibrium and risk management, thus may not be trustworthy to some extent. In an era with an increasing desire for privacy, security and anonymity seem to be mutually exclusive. With the rapid development in the field of Internet communication and cryptography, the need for

1.2 Development History Overview

Cryptocurrencies have provided us with an option to handle financial dealings without any institutions acting as the middle man. The most prevalent cryptocurrency at this moment is Bitcoin. It currently has a market capitalization over US$112 billion (updated: Oct 22th 2018), which is almost equal to IBM. As shown in Figure 1.1, Bitcoin makes up the most of the total market capitalization of cryptocurrencies.
Figure 1.1 Time-series Components of Cryptocurrency Market

However, before the emergence of Bitcoin, there had been a number of virtual-cash protocols using different technologies already. Among which, the most well-know is the e-cash protocol developed by David Chaum and Stefan Brands in 1982. Several e-cash applications such as *Electronic Cash Payment System* were marketed across the world. In 1998, Dr Wei Dai made a breakthrough by proposing a digital asset called b-money and outlining the fundamental features of decentralized cryptocurrency. He also contributed to the open source C++ class library of cryptographic algorithms and schemes called ‘*Crypto++*’, which is known as the cornerstone of blockchain technology.

In January 2009, five months after the registration of the domain named Bitcoin.org, an anonymous developer who identified himself as Satoshi Nakamoto mined the first block of Bitcoin. It was at this historical moment that Bitcoin made its debut to the public. The blockchain technology it adopted supports peer-to-peer transactions without the meddle of any third party, making it the first genuine decentralized cryptocurrency protocol applicable in reality. However, given the finite amount of Bitcoin, it takes some default risk exposing to the market, putting it in a rather weak position against inflation. As we can see in *Figure 1.2*, the price of Bitcoin is
drastically volatile. Therefore, Bitcoin is more of a commodity than actual currency at present.

![Figure 1.2 Historical market price of Bitcoin](image)

During the past decade, we have seen the continuous growth in the scale of cryptocurrency market. It reached the market capitalization peak of over US$813 billion on Jan 8th, 2018, more than 10% of the corresponding stock market capitalization in mainland China. Financial derivatives of cryptocurrency such as Bitcoin futures also started trading in CME and CBOE. Numerous of new cryptocurrencies such as Ethereum and Litecoin were created as well. Now, there are over 1600 types of cryptocurrencies available and growing. This shows an unprecedented prosperity and prospect in the area of currency virtualization.

1.3 Blockchain Overview

The blockchain technology, as its name implies, string millions of verified blocks containing all kinds of information. The architecture design of blockchain makes it impossible to delete or tamper any blocks that have been previously attached to the main chain. In the absence of authorities, cryptocurrencies use blockchain technology
to store ledgers in a distributed tamper-evident database. The transaction records are distributed and stored among users. Whenever there are new blocks coming out, the system implements a consensus algorithm to decide whether to accept these new blocks or not. The consensus algorithm cryptocurrency adopts can have significant effects on its stability. Therefore, security analysis of these consensus algorithms are needed.

1.4 Scope of work

This project focuses solely on the study of consensus algorithm adopted by different cryptocurrencies. The algorithm should work under the assumption that the cryptocurrency uses blockchain technology to store the ledgers in a distributed tamper-evident database. It aims at coordinating users in utilizing their version of distributed ledgers and deciding whether a certain transaction is valid. There are several possible reasons to make transactions fraudulent, such as balance not enough to cover the amount, or the capital exposed to double-spending promises etc. If by this algorithm, users in the network verify the validity of the block containing the transaction, the block will be added to the main blockchain.

Other algorithms or mechanism adopted by cryptocurrencies regarding cryptographic hashing functions, transaction mode such as Unspent Transaction Output protocols and blockchain architectures will not be included in the project.

1.5 Outline of the Report

This report will first go through the universal mechanism cryptocurrency adopts to support peer to peer transaction in Chapter 2. Relative subjects include generating the private/public keys and temper-evident database. Then it looks into the background
theorem of consensus algorithm and provide the audience with the basic information in understanding the BFT. The objective of the project will be presented in chapter 3. In chapter 4, the report will first make introduction and justification of methodology. Three major properties in security analysis will be introduced as well as the reasoning of choosing POW as a benchmark. Comparative study on POW and Algorand will be shown as preliminary results later in chapter 5. Detailed analysis follows the order of how Algorand outperform POW in energy consumption, class differentiation and dealing with forking attacks.
Chapter 2. Background Theorem

2.1 Cryptocurrency Mechanism Overview

One of the major common features among cryptocurrencies is decentralization. Cryptocurrencies support peer-to-peer transactions, no authorities or any third party is involved in the operations. In Bitcoin, each user is randomly distributed with an account represented by a private key of 256 bits. In binary representation, that is $2^{256}$ possibilities. In contrast, there are $2^{60}$ grains of sand on earth. (Bach, 2018) When considering the scenario where two users are distributed with the same public key, it is even more unlikely than two people randomly picking the same grand of sand on earth. As shown in the cryptographic process in Figure 2.1, the private key later generates another unique and deterministic public key in a one-way hash function. Therefore, there is no way of reverse-engineering the generating process. One can not use public key to come up with the corresponding private key. A private key is needed to withdraw money thus should be kept to oneself. Public key, handles receiving process thus should be offered to the other party for transactions. The remitter uses his private key as digital signature for identity verification, and the public key of the recipient as account address to make transactions.

![Figure 2.1 Cryptographic hash process](image)

- 6 -
Another common feature cryptocurrencies share is anonymity. The whole idea of constructing cryptocurrency is based on anarchism and individualism. There is no need to provide any documentation in opening up an account, no way of revealing the real identity of the users.

2.2 Consensus

Cryptocurrency supports peer-to-peer environment, where ledgers are recorded and added to the blockchain network once verified. The consensus problem depicts a scenario where a supposedly legitimate transaction is proposed, given that faulty processes and deceptive frauds are possible, users are supposed to reach an agreement on whether the block recording that transaction should be added to the blockchain or not. (Ittay,2014) In Bitcoin for example, a consensus needs to be achieved from at least a fifty-percent majority ranked by computational power according to the ledgers in the distributed database.

2.2.1 The Byzantine Generals Problem

The Byzantine Generals Problem was first proposed by Leslie Lamport and his fellow colleagues in 1982.

In the original problem, several Byzantine generals each in charge of a division were in an urgent situation of attacking an enemy city. The only way of an insecure communication is through messengers.(Barber, 2012) The best strategy for them to get out of this battle is through acting on the same pace. Thus, to ensure that every single general acting simultaneously is crucial. The difficulty is that messengers might be captured and never reach the destination, losing the messages permanently. Furthermore, one or more generals might be traitors and motivated to send false or
distort messages. (Tuyet,2017) The communicating mechanism needs to ensure the loyal generals are able to reach an agreement on the attack; traitors and loss of messages shouldn’t sabotage the battle.

In the case of cryptocurrencies, the system should be able to deal with the failure of one or more of its components. These failure will lead to fraudulent transaction and may result from the systematic error in the network or false messages provided by malicious users. In the presence of fraudulent transaction and falsification of ledger records, the honest majority should still be capable of reaching a consensus regarding the validity of a certain block.

2.2.2 Byzantine Fault Tolerance

The Byzantine Fault Tolerance is a category of replication algorithms that are resilient to the vulnerability of the system to reach consensus when arbitrary data are generated by the nodes. (Yossi,2017) BFT can guarantee the smooth running of the system without being influenced by faulty message and malicious falsification.

We will later use these algorithms as a benchmark to evaluate the security of the consensus algorithm. Algorithms that could outperform the Byzantine Fault Tolerance application are considered relatively safe.

2.2.3 Blockchain Platform

Blockchain platforms can be classified into two categories, permissionless and permissioned. Each platform applies a corresponding type of consensus algorithm that fits the architecture of the system.(Baliga,2017)

The permissionless platforms are typically open-end. Cryptocurrencies using these
platforms such as Bitcoin and Ethereum are available to the public, any node in the network could make transactions and join in the process of advancing the blockchain. (Spencer, 2018).

The permissioned platform, on the contrary, are typically close-end. It is still available to the public to join the system, but it is restricted to only a fixed set of nodes representing consortium members to advance the blockchain.

2.2.4 Consensus Algorithm

The consensus algorithms maintain the authenticity of the data recorded on the blockchain. A poor consensus mechanism could compromise the data and endanger the entire system. Therefore, any blockchain system is as secure and robust as its consensus mechanism.

The permissionless platform needs large amount of anonymous and untrusted node to join the network. A metaphor for these consensus algorithms can be a referendum. Each user can vote on a motion to add the new block to the blockchain or not. The weight assigned to each vote and the special decision-making mechanism are what make these algorithms different. Due to the pseudo-anonymity, if consensus algorithm distributes equal voting power to each node, each user is capable of generating several nodes to maliciously manipulating the consensus procedure. (Gabizon, 2017) Therefore, consensus algorithm in permissionless platform will need to deal with regulating voting power. As shown in Table 2.1, such algorithms include proof-of-work, proof-of-stake, etc. In the case of Bitcoin, it applies proof-of-work mechanism to distribute voting power proportional to computational power. The user needs to provide evidence of effort put into mining, the computational power, to gain voting power on advancing blockchain. To avoid collusion, one will need the majority
vote in order to verify a certain block.

<table>
<thead>
<tr>
<th>Consensus Algorithm</th>
<th>Cryptocurrency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proof of Work</td>
<td>Bitcoin</td>
</tr>
<tr>
<td>Power of Stake</td>
<td>Ethereum</td>
</tr>
<tr>
<td>Delegated Proof of Stake</td>
<td>Bitshare</td>
</tr>
<tr>
<td>Proof of Authority</td>
<td>Ethereum (In process)</td>
</tr>
</tbody>
</table>

Table 2.1 Consensus Algorithms and Cryptocurrencies on Permissionless Platform

Permissioned platform uses consorting mechanism. An appropriate metaphor for these algorithms would be a representative system. Users elect a ‘cabinet’ to make decisions on their behalf. The public aren’t directly involved in the process of advancing blockchains. The consortium members registered and verified as semi-trusted members are anonymous to the public. But within the group, the public keys are shared among consortium members. As shown in Table 2.2, such algorithms, including Paxos, RAFT, and various Byzantine Fault Tolerance, should focus on resilience to collusion within the consortium group mainly.

<table>
<thead>
<tr>
<th>Consensus Algorithm</th>
<th>Cryptocurrency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paxos</td>
<td>Paxos standard token</td>
</tr>
<tr>
<td>RAFT</td>
<td>Logcoin</td>
</tr>
</tbody>
</table>

Table 2.2 Consensus Algorithms on Permissioned Platform
Chapter 3. Objective

This project aims at providing security analysis as well as possible improvements to the consensus algorithm used by both permissioned and permissionless platforms, and to some extent, the variant rendered as the mixture of the both kinds.

3.1 Comparative study

During the first phase, the project was purely theoretical. The goal at this stage was to provide comparative security analysis of typical consensus algorithms. The comparison includes algorithms applied at the same platform as well as hybrid consensus mechanism. The project at this stage focused on the algorithmic steps taken by each of the consensus algorithms, how these algorithms reward each validator for their efforts put into verifying the blocks as well as the scalability of the algorithms. Explanatory illustration on consensus failure and poor performance in latency will also be provided.

3.2 Algorithm improvement

During the second phase, the focus of the project shifted to the study of Algorand, a hybrid consensus algorithm. First, it shows the security risk within the algorithm and came up with a possible improvement in the cryptographic sortition procedure. The project at this stage focused on testing the immutability and auditability, that is, whether the verified blocks with underlying assumptions are indeed valid, and make sure fraudulent transactions such as double-spending are thrown away, and in permissionless platform specifically, no arbitrage opportunity in gaining extra reward from validators. The corresponding attacks include blockchain fork, dominance by individual or group of entities as well as cheating by maintaining false parallel forks.
Improvements made in this process should also abide with principles such as decentralization and pseudo-anonymity.
Chapter 4. Methodology

4.1 Introduction

Theoretically, the protocols applied to cryptocurrencies are secured on condition that the system run smoothly with tolerable systematic faults and each user abides with the regulations. However, technologies could never defeat human nature and act in accordance with it. In this sense, consensus algorithm should have some protective mechanism against malicious collusion and conspiracy. The evaluation of the security on each cryptocurrency should be as good as the scale of maximum collusion and conspiracy it can bear. Theoretical deduction of attacks will be carried out to test the vulnerability of the algorithm.

4.2 Qualititative Evaluation

As the security evaluation approach stated in 4.1 are not quantifiable, we need to focus on the vulnerability of different protocols based on their performance under all possible attacks. Security analysis in consensus algorithm will focus on testing three major properties as follows.

a. Safety: All nodes show consistency output and the output is proved valid. This suggests the nodes act on the same pace and the input/output algorithm can run smoothly.

b. Liveness: Guaranteed liveness if all non-faulty nodes join the consensus-making process. This means a block containing transaction information can be verified smoothly if all the honest users contribute to reach consensus. If each user were to share the real information stored in their share of the ledger, the system would be able to run without any congestion or failure in the absence
of the involvement from any authorities. (Marko, 2016)

c. Fault Tolerance: The ability to recover from a failure state. This property measures the maximum fault a consensus algorithm can bear. If malicious users collude and conspire with one another, the algorithm should be able to identify and recover from the attack.

4.3 Benchmark Selection

The most important step for running qualitative analysis is the selection of a benchmark. We need to use the benchmark as a contrast to evaluate the performance of the algorithm. Ordinarily, the benchmark consensus algorithm we use is the POW. The reasons behind are as follows.

First of all, POW is the most widely used consensus algorithm around the world, the working mechanism behind POW is relatively clear and straightforward. Therefore, it may mitigate the potential risks in lack of the study materials.

Secondly, the POW algorithm is a prototype that rigorously satisfy the basic need of cryptocurrencies. In other words, the performance of POW is considered the minimum requirement for any cryptocurrencies to function smoothly. Any algorithm that outperform POW could be considered a progress.

Lastly, the POW algorithm has several major flaws. The later proposal of the hybrid algorithm can thus focus on making modifications to these flaws. Therefore, it may also mitigate the potential risks in dealing with complexity.
Chapter 5. Comparative Case Study

In this chapter, we will list the result of a comparative case study on the POW and BFT. The paper will first list and define the qualities to be compared that include consensus finality, scalability, performance, network synchrony and correctness proof. Then it goes into detail and compare the qualitative differences of the two types of consensus algorithms.

5.1 Consensus Finality.

Generally speaking, consensus finality, sometimes known as the forward security (Decker, 2016), is often informally referred to the property that a valid block proposed in time, and appended to the blockchain at some point, could never be removed from the blockchain. This goes to the very core of blockchain features, to ensure a tamper-evident database that is safe and secure to store all the information of transactions made by the users. The formal definition translated to the blockchain terminology can be described as below:

*Definition 1 Consensus Finality.* No valid node $p$ could attach block $b$ to its copy of the chain if any valid node $p^*$ has tried to append block $b^*$ to its copy of the blockchain.

It is obvious that the consensus finality is not meet by the PoW-based blockchains. For starter, we need to focus on the identity management mechanism of POW. To explain why, we must also acknowledge the fact that the concurrency control mechanism of POW is randomized, which suggests that the frequency of the block proposed is adjusted so as to put the odds of collisions, that is, the simultaneous attempts to append different blocks to the blockchain, under reasonable likelihood.
However, the concurrency control is not deterministic, since the propagation over the network is considered quite a challenge in practice, which is also why the propagation time is frequently used as a criteria for determining how good a consensus algorithm is on a relative term. Therefore, the collision happen from time to time. The result of collision is usually temporary forks that could not be avoided even if all the users were proved to be honest. In Bitcoin, such problem is dealt with under the longest fork rule (Nakamoto, 2009), or the GHOST rule (Sompolinsky, 2015). Only the longest rule is entitled to append to the main blockchain. Nevertheless, the very existence of these temporary forks suggests the absence of consensus finality, in lack of which, the probability that a valid transaction is left pruned and removed from the mainchain will increase and thus causing the entire system inefficient.

In comparison, BFT could easily meet the requirement of consensus finality, since the validity of each proposed block is verified within the consortium group. Unless the consortium reaches an agreement on whether append or dessert the block, the main chain will not proceed to add another different temporary chain. Compared to POW, this is clearly a advantage of the BFT-based blockchains. In practice, such advantage could be put into application, where users and smart contracts could immediately confirm the validity of a transaction to be recorded in the blockchain.

5.2 Scalability.

The scalability of the blockchain is graded upon the number of nodes and clients in the system that could support efficiency and authenticity. It has some overlap area with the performance of the blockchain. Although it is not possible to completely seperate the issue of blockchain scalability from the performance, we first take a glance over the constraints on nodes and clients that POW-based and BFT-based
blockchain that have already been proven to work properly in practice.

For POW-based network, we’ll use Bitcoin as example for illustration. The Bitcoin spans its network over thousands of nodes, which goes to show the node scalability of POW-based blockchains in application. Be that as it may, the class diversification in Bitcoin that classifies miners, with incentives to split mining rewards and making financial profit, into the mining pools harm the decentralization characteristic of cryptocurrency to some extent. It is also worth mentioning that the mining pool centralization is not a unique thing in the Bitcoin system. In fact, it is the POW they deploy that should be held accountable. Other POW-based blockchain network such as Ethereum also suffer from this damage.

For BFT and its similar state-machine replication, however, are considered consensus algorithm with poor scalability. Although there has been quite a lot of applications built under the BFT premises, the scalability of these BFT protocols were never really tested thoroughly. One reason that could account for the lack of test is the complexity of messages (i.e. \(O(n^2)\)) per block (Castro, 2002), which made the BFT protocols not scalable in the database and systems communities. (Mickens,2014) Other replication protocols of BFT, a lot of which are crash-tolerant counterparts, such as Paxos (Lamport,1998), Zab (Junqueira,2011) and Raft (Ongaro,2014), are put into application in some large scale system, but never across more than 5 of the replicas.(Corbett,2013).

In terms of the scalability with the number of clients, on the other hand, both PoW and BFT protocols have a relatively good performance that support thousands of clients and scale well.
5.3 Performance.

Apart from the relatively poor performance of Bitcoin has in terms of the efficiency in making transactions (7 transactions/second) and latency of block confirmation (1 hour for every 6 blocks), POW-based blockchains have met with other performance difficulties in boosting efficiency.

Since the two major parameter that influence the performance of a POW-based blockchain are block size and block frequency, we now go into the detail about how these two parameter affect the performance. To begin with, the desire to boost throughput through increasing the size of the blocks also come at the expense of larger latency. The larger size of the proposed block, the longer of POW propagations delay. Due to the possible temporary chain forks and lack of consensus finality in POW, these longer delays may increase the probability of forks and double-spending attacks (Karame, 2015), as a very negative implications on the security of cryptocurrency.

Increase in the block frequency could to some extent mitigate the latency problem caused by multi-block confirmation. However, it could also cause the similar insecurity as the increase of block size. The security implication is that the calibration of both block frequency and block size in POW-based blockchain are indeed a zero-sum game, and should be dealt carefully (Sompolinsky, 2015). Therefore, the poor performance of the POW-based blockchain is inherent from the nature of the mechanism, and thus should not be treated as a normality in application.

In comparison, it has been proved that the modern BFT protocols could sustain
transactions with network latency of the practical speed in the practical system. That is, the tolerance of the latency in the BFT protocols rest in the feature of the system itself, not the efficiency of the consensus algorithm.

5.4 Network Synchrony

Synchrony is crucial in the blockchain system when it comes to proposing and verifying the blocks. The Bitcoin uses the local time of each note to stamp a block. The general logic is that the transactions contained in the block is only accepted as valid if the time stamped on the block is larger than the median of the last 11 blocks. Moreover, the timestamps could also function as a parameter in estimating the difficulty of solving puzzles in the mining process, as well as managing the block frequency.

As a consequence, it is required by the POW-based blockchain to have a synchronized system to maintain the liveness of the chain. However, this could also give rise to the timestamp manipulation attacks that could hurt the consistency of the blockchain. Even if the odds for malicious users to carry out such attacks are very little in large POW-based blockchains such as Bitcoin, other cryptocurrency that deploy POW such as altcoin have suffer from such attacks.

BFT protocols on the other hand, don’t usually require any physical clock. The synchronous communication required to keep the liveness of the system in BFT-based block chain are relatively synchronous. That is, BFT only requires the same unit of measure in time among all the users as a estimation of the running time, but not any local timestamps to mark the exact time of the proposed blocks. The safety properties of consensus, including consensus finality, are maintained despite global
communication outages and arbitrarily long asynchrony periods (Dwork, 1988).

5.5 Correctness Proof

The state-machine replication protocols, in terms of the BFT variants, are considered quite challenging to design and implement (Castro, 2002 & Bessani, 2014 & Chandra, 2007). As a consequence, new protocols prone to receive detailed academic scrutiny, thus sometimes come with a detailed proof on the correctness (Prisco, 1999 & Losa, 2014).

As an example, Bitcoin came into market without any security scrutiny, it is quite surprising that POW-based blockchains rarely come up with a detailed security analysis and distributed protocol correctness proof, which is also why some people question the authenticity in the characteristics of decentralized cryptocurrency.
Chapter 6. Result

This chapter is a demonstration of the security analysis on Algorand. It will first go through the innovative solutions that Algorand come up with in electing consortium and reaching consensus within it. The

Then use POW as benchmark, and conducts a comparative study to show how Algorand outperforms the POW.

6.1 Study on Algorand

Algorand was first proposed in April, 2018 by Turing Award Winner Professor Sivio Micali. The word 'Algorand' comes from 'algorithm' and 'random'. As its name implies, Algorand is based on public ledger calculated and distributed by random algorithm. It aims at improving some major weakness in the Proof of Work algorithm applied in Bitcoin.

6.1.1 Cryptographic Sortition

6.1.1.1 Overview

In POW, since it is based on a permissionless platform, there is no need for selecting a subset of users as consortium. Algorand, on the other hand, performs on a permissioned platform, which requires the consensus algorithm to randomly select a subset of the users. Cryptographic Sortition is the most innovative mechanism proposed in Algorand. It uses a set of verifiable random functions (VRF) to randomly pick the node in the blockchain instead of blindly throwing darts as they do in POW. VRF can be seen as a predictor, as shown in Figure 6.1, it requires an input and evenly distributed the output among the given range. The output is deterministic,
namely, given the same input, VRF can only come up with one output. The makeup of VRF and their functionality are shown in Table 6.2.

<table>
<thead>
<tr>
<th>VRF</th>
<th>Function Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VRF\text{\text{GEN}}</td>
<td>Generates pair of keys</td>
</tr>
<tr>
<td>VRF\text{\text{VAL}}</td>
<td>Generates pseudorandom variable</td>
</tr>
<tr>
<td>VRF\text{\text{PROVE}}</td>
<td>Calculate proof</td>
</tr>
<tr>
<td>VRF\text{\text{VER}}</td>
<td>For anyone with public key to verify</td>
</tr>
</tbody>
</table>

VRF introduces a concept of ‘seed’ that is left for the user to select. If users are picked to become consortium member, it gives the proof to VRF, and the VRF will then come up a credential to certify the identity of this consortium member and a value for non-consortium member to verify his identification. The overall process and the details are presented in Figure 6.2 and Figure 6.3. As shown in Figure 6.3, the generating of value follows binomial distribution.
Figure 6.4 and 6.5 show the overview and detailed verification process respectively. Since binomial distribution can be reversed calculated, there is no need for providing

the private key of the chosen consortium members. The users could simply compare the input of the binomial hash function \( j \) and \( j' \) to decide the verification result.

6.1.1.2 Details of Cryptographic Sortition

Cryptographic sortition is an algorithm for choosing a random subset of users according to per-user weights; that is, given a set of weights \( w_i \) and the weight of all users \( W = \sum w_i \), the probability that user \( i \) is selected is proportional to \( w_i / W \).

(Micali, 2017) To ensure the randomness in the sortition algorithm, Algorand introduced another variable that comes from a publicly known random seed. To prove a specific user is chosen, sortition requires each of the user to use the standard private and public key \( (sk_i, pk_i) \).

Sortition is implemented using verifiable random functions (VRFs). Informally, on any
input string $x$, $\text{VRFsk}(x)$ will return two values: that is a hash and a proof. The hash is a hashlenbit-long value that is uniquely determined by $sk$ and $x$, but is indistinguishable from random to anyone that does not know $sk$. The proof $\pi$ uses the typical key-pair verification, where it enables anyone who possesses the $pk$ to check that the hash indeed corresponds to $x$, without having to know the $sk$. For security reasons, Algorand requires that the VRF provides these properties even if $pk$ and $sk$ are chosen by an attacker.

Using VRFs, Algorand implements cryptographic sortition as shown in Figure 6.6.

```
procedure Sortition(sk, seed, $\tau$, role, w, W):
    $\langle \text{hash}, \pi \rangle \leftarrow \text{VRFsk}(\text{seed}||\text{role})$
    $p \leftarrow \frac{1}{W}$
    $j \leftarrow 0$
    while $\text{hash} \neq \sum_{k=0}^{j} B(k; w, p), \sum_{k=0}^{j+1} B(k; w, p)$ do
        $j \leftarrow j + 1$
    return $\langle \text{hash}, \pi, j \rangle$
```

Figure 6.6 The cryptographic sortition algorithm.

Sortition requires a parameter that stands for the role of the selected user. The “role” parameter help to distinguish the different roles that a user may be selected for. It is crucial that the sortition process selects users in proportion to their weight assigned, otherwise, the sortition would not defend against Sybil attacks as we have mentioned in chapter 5. One observation regarding the cryptographic sortition is that users may be chosen for more than once. This could be accounted by the higher weight some users may have. Therefore, algorand uses a parameter $j$ to indicate how many times users was chosen. Being chosen for $j$ times suggests that the user gets to participate as $j$ different sub-users. If user $i$ owns $w_i$ (integral) units of Algorand, then simulated user $(i,j)$ with $j \in \{1, \ldots, w_i\}$ represents the $j$th unit of currency $i$ owns, and is selected with probability $p = \frac{W}{\tau^w}$, where $W$ is the total amount of currency units in
Algorand.

As shown in Figure 6.6, a user performs sortition by computing \( \mathbf{hash, \pi} \leftarrow \text{VRFsk(seed}\|\text{role}) \), where \( sk \) is the user’s secret key. The pseudo-random hash determines how many sub-users are selected, as follows. The probability that exactly \( k \) out of the \( w \) (the user’s weight) sub-users are selected follows the binomial distribution, \( B(k;w,p) = \binom{w}{k}p^k(1-p)^{w-k} \).

6.1.1.3 Normal distribution approximation

The flaw with this algorithm is that, the scalability could be hurt badly, since the computation of binomial distribution requires quite a lot of time, and the algorithm itself is rated as \( O(n^2) \) in terms of complexity, where \( n \) stands for the potential number of users in the Algorand-based network. As we can see in Figure 6.7, the running time of the cryptographic sortition using binomial distribution grows approximately similar to a quadratic function.

![Figure 6.7 Running time of cryptographic sortition using binomial distribution](image)

To make improvements regarding the complexity of the algorithm, we suggest the approximation of a normal distribution as a substitute. In stead of the while loop at the
end of the algorithm, we could use the pdf function of the normal distribution to indicate the value of $j$. That is $j = \text{normpdf} \left( n, \frac{\text{hash}}{2^{\text{length}(\text{hash})}}, np, np(1 - p) \right)$. Where we simulate the normal distribution with a mean of $np$, a standard deviation of $np(1-p)$. And the pdf function can directly use the transformation of N(0,1) to directly locate the parameter $j$ that satisfy the requirement. The new process save the time of a while loop, and in practice could largely reduce the running time and complexity of the cryptographic sortition, as shown in Figure 6.8.

![Figure 6.8 Running time of cryptographic sortition using normal distribution](image-url)
5.1.2 Byzantine Agreement

The Byzantine Agreement is the actual algorithm for the consortium group to reach consensus. To simplify the voting process, BA uses hash values to represent each proposed block. As highlighted in Figure 5.6 and Figure 5.7, the consortium can either vote the hash value of the block valid or simply propose the empty block. Given the relatively limited number of members in the consortium chamber, which by custom is usually kept under twenty, we could assume the vote by the honest majority solid. Therefore, BA only needs to deal with the possible latency in the network.

```plaintext
while step < MAXSTEPS do
    CommitteeVote(ctx, round, step, τSTEP, r)
    r ← CountVotes(ctx, round, step, τSTEP, λSTEP)
    if r = TIMEOUT then
        r ← block_hash
    else if r ≠ empty_hash then
        for step < s' ≤ step + 3 do
            CommitteeVote(ctx, round, s', τSTEP, r)
        if step = 1 then
            CommitteeVote(ctx, round, FINAL, τFINAL, r)
    return r
step++
```

Figure 6.9 Timeout mechanism in voting non-empty block

A timeout mechanism is proposed by Algorand to deal with the latency problem in a network. As shown in both Figure 5.6 and Figure 5.7, it applies a three-step gossip to broadcast the vote within the consortium group. At each round of the BA, the consortium members will either reach consensus on the validity of the block within three steps, or attach a empty block to the chain as a signal of the failed consensus.

```plaintext
r ← CountVotes(ctx, round, step, τSTEP, λSTEP)
if r = TIMEOUT then
    r ← empty_hash
else if r = empty_hash then
    for step < s' ≤ step + 3 do
        CommitteeVote(ctx, round, s', τSTEP, r)
    return r
step++
```

Figure 6.10 Timeout mechanism for voting empty block
5.2 Comparative Study between POW and Algorand

With the unique cryptographic sortition mechanism Pro. Macali came up with, Algorand has outperformed POW in many ways.

To begin with, Algorand save users the energy spent on mining. According to Digiconomist.com, it takes $1.18 \times 10^{22}$ times of hash function computation to mine a unit of Bitcoin on average. It’s equivalent to rolling a dice with $1.18 \times 10^{22}$ dimensions and wish to get a specific number. The electricity consumed per annum on the blockchain system of Bitcoin is roughly the same as annual consumption in Peru and still growing. The energy spent on these meaningless mining has contributed nothing valuable to society. Algorand, on the other hand, activates the Variable Random functions to perform the ‘throwing dice’. The electricity spent on Algorand are not even remotely of the same scale as POW.

Additionally, to maintain the healthy status of Bitcoin blockchain, it requires that honest users should possess the majority (at least more than 50%) of the computational power. Once malicious users collude with a majority, the validity of the block will not be guaranteed. The ‘referendum’ system initially plays an important role in preventing users from class differentiation. (Miller,2017) However, as the transactions proceed, the computational power is limited to several large ‘mining pool’, putting the corresponding miners in an most advantageous position of gaining rewards. Algorand doesn’t have a mining mechanism. Therefore, it implements the idea of ‘absolute democracy’ throughout the whole consensus making process. The concept of ‘decentralization’ is also guaranteed in Algorand while miners in POW may somehow become the new authorities.
Furthermore, the blockchain system of Bitcoin is more likely to generate 'forking' problem. In the white paper Nakomoto Stoshi published in 2009, when a transaction is recorded in a block and the block is added to the temporary chain, POW will need six extra blocks attached to the first one to confirm the validity and put the transaction information from the first block into public ledger. (Satoshi, 2009) However, if some malicious attacker deliberately hides several extra blocks he mined and attach one or several forged blocks to satisfy the number of six, then POS will get into a common forking trouble as shown in Figure 5.7. In this case, the main chain would accept the longer chain of blocks. Although the transactions recorded in the block would be put into public ledger only after verification, the malicious attacker would still benefit from the rewards of the forged block.

![Figure 6.11 Hard Forking: Non-upgraded nodes rejecting new rules](image)

In this comparative case study on Algorand, we use POW as a benchmark to evaluate the performance of Algorand as well as the drawbacks shown by the POW. This comparative study shows that the hybrid consensus algorithm such as Algorand has a better chance in both surviving and recovering from the possible attacks. Major flaws of POW including high energy consumption, class differentiation and forking attacks are also mitigated.
6. Conclusion

6.1 Summarization

The report has introduced the mechanism of cryptocurrency and the consensus algorithm implemented in blockchain technology. To some extent, Algorand is a hybrid algorithm. On the one hand, it adopts the consortium chamber to cast the vote as algorithms on the permissioned platforms do. On the other hand, it also assign different weight on each vote cast by the consortium members as algorithms on the permissionless platforms do. The comparative study conducted in phase one has pointed out how Algorand outperform the POW algorithm implemented in Bitcoin and justified the practicality of Algorand. The paper has shown the overall advantages of Algorand in energy conservation, decentralization preservation and the protective mechanism against attacks such as forking.

The security analysis of cryptocurrency is an entry point for improving the consensus algorithm. The consensus algorithm is vital in that it carries the responsibility of maintaining the blockchain credibility. Further improvements on consensus algorithm can be most helpful in raising the security level of cryptocurrency as well.

6.2 Future Study

Since consensus algorithm is nothing more than a different combination of minor algorithms, how to make sure each part function as they are supposed to is the key to the proposal of a new hybrid consensus algorithm.

The project has shown the improvement of changing the binomial distribution into normal distribution in cryptographic sortition algorithm and ran a security analysis to Algorand.
Future study also include possible changes on minor algorithms may include change of VRF and introducing another round of voting in BA system. Alternative of such distribution includes normal distribution, poisson distribution etc.
References:


the Conference on Dependable Systems and Networks (DSN), pages 245-256, 2011.


27. Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is


34. Klaus Kursawe and Victor Shoup. Optimistic asynchronous atomic broadcast.
In Automata, Languages and Programming, 32nd International Colloquium, ICALP, pages 204-215, 2005.


57. The 'Zeitgeist attack'.


66. Zsolt István, David Sidler, Gustavo Alonso, and Marko Vukočić. Consensus in