Role of Game-Theory and Fairness in Designing Blockchain Protocols

Thesis Proposal submitted in partial
fulfillment of the requirements of the degree of

Master of Science
in
Computer Science and Engineering
by Research

by

Anurag Jain
20171021
`anurag.jain@research.iiit.ac.in`

Advised by Dr. Sujit P Gujar

International Institute of Information Technology
Hyderabad - 500 032, INDIA
March, 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Role of Game-Theory and Fairness in Designing Blockchain Protocols" by Anurag Jain, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Prof. Sujit P Gujar

To my parents, Sonia and Sumir Jain

Acknowledgments

I am grateful to Prof. Sujit Gujar for guiding me throughout my research journey. It is only due to him that I have been given immense freedom to explore new ideas and produce the results in this thesis. I am also indebted to fellow researchers with whom I had the opportunity to work with, Prof. Kannan Srinathan, Prof. Dimitris Chatzopoulos (HKUST) and Prof. Emmanuelle Anceaume (INRIA).

I am grateful to the faculties from whom I gained both knowledge and wisdom and every other member of IIIT-H for helping me throughout my long journey toward my graduation. My dear friend, Harsh, for not only being an excellent companion but a well-wishing advisor, Ashit Rustagi, my roommate, Sayantan Jana, Shaunak Badani, Tanuj Garg, and my friends with whom I shared a colorful college life. Seniors who inspired me and guided me through my research journey, Sankarshan Damle, Kumar Abhishek, Moin H. Moti, Shoeb Siddiqui, and Tirth Maniar. My labmates, Samhita Kanaparthy, for her constant support and company, and Debojit Das, for his helpful nature.

I am forever grateful to my parents, who have supported and motivated me through every endeavour of my life.

Abstract

Blockchains lie at the heart of Bitcoin and other cryptocurrencies that have shown great promise to revolutionize finance and commerce. The novel applications of blockchain technology are derived from the unique combination of properties of blockchains: immutability, incorruptibility, consensus, transparency, decentralization and ordering. However, they face technical challenges when it comes to scaling to support greater demand while maintaining their desirable security properties. In an exciting line of recent work, many researchers have proposed various scalable blockchain protocols that demonstrate the potential to solve these challenges. However, many of these protocols come with the assumptions of honest majority and symmetric network access which may not accurately reflect the real world where the participants may be self-interested or rational. In this thesis, we work towards analysing blockchain protocols in settings with rational participants and use game-theory to predict their behaviour. For the first time in literature, we study blockchain protocols in a setting with asymmetric network access and highlight the role of *network fairness* in ensuring the security of a blockchain protocol. We demonstrate via simulations, the effect of lack of network fairness in Bitcoin and the equilibrium behaviour of the participants which results in loss of both security and performance of the protocol. In order to further highlight the importance of network fairness in designing scalable blockchain protocols, we showcase incentive-driven deviations in the OHIE blockchain protocol proposed recently and the loss of security caused by these deviations. We then study a class of blockchain protocols designed by layering two distinct types of consensus protocols and determine the parameters required to ensure game-theoretic soundness and security. We also formally study the reward schemes employed by Proof-of-Work-based blockchain protocols and showcase the lack of game-theoretic soundness. Towards this, we also propose a class of game-theoretically sound reward schemes. Finally, we extend our work by developing a model for layered-blockchain protocols and present a game-theoretic model for analyzing the security of layered-blockchain protocols. Therefore, in this thesis we present novel work that aims to guide the development of fair and game-theoretically sound blockchain protocols.

Research Papers Based on the Thesis Work

Conference Papers

1. Anurag Jain, Shoeb Siddiqui and Sujit Gujar. “We might walk together, but I run faster: Network Fairness and Scalability in Blockchains” In the proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems. **Best Poster Award (AAMAS ’21)**

2. Anurag Jain, Sanidhay Arora, Sankarshan Damle and Sujit Gujar. “Tiramisu: Layering Consensus Protocols for Scalable and Secure Blockchains” **(ICBC ’22)**

Peer-reviewed non-archival Publications

1. Anurag Jain and Sujit Gujar. “Block Rewards, Not Transaction Fees Keep Miners Faithful In Blockchain Protocols” Workshop on Game Theory in Blockchain, 16th Conference on Web and Internet Economics **GTIB@WINE2020**

Other Research Papers

1. Anurag Jain, Sujit Gujar and Kannan Srinathan. "Interlude: Balancing Chaos And Harmony For Fair and Fast Blockchains" arXiv Preprint (arXiv:2209.10125)

2. Dimitris Chatzopoulos, Anurag Jain, Sujit Gujar, Boi Faltings and Pan Hui. "Toward Mobile Distributed Ledgers" IEEE Internet of Things Journal Vol. 9 Issue 11 **(IEEE IoTJ)**

3. Sanidhay Arora, Anurag Jain, Sankarshan Damle and Sujit Gujar. "ASHWAChain: A Fast, Scalable and Strategy-proof Committee-based Blockchain Protocol" Workshop on Game Theory in Blockchain, 16th Conference on Web and Internet Economics **GTIB@WINE2020**

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Bitcoin is the first "killer" blockchain technology app, just like email was the first "killer" internet app. I think the sky's the limit. We're in the very, very early stages here.

Blockchain technology (Nakamoto, 2009) is proposed as the panacea for developing decentralised applications and the underlying technology that powers decentralised finance. It has the potential to be a solution provider for any business, especially in securing and decentralising their data and automating contracts between different entities via *smart contracts* (Ethereum Community). A blockchain is a *decentralised ledger*, i.e., not requiring a central administrator. It has a unique set of properties that enable it to build avant-garde systems that offer unprecedented disruption in nearly every industry, from logistics to supply chains, to finance and insurance. Blockchain technology is poised to become the centrepiece of economies worldwide, transcending the domains of e-commerce, healthcare, and public sectors.

At its core, a blockchain is a simple data structure that stores data / records in the form of blocks chained together via cryptographic hash pointers. Based on its design, it offers the following inherent properties that make it useful for a multitude of applications:

- *Immutability*: Records stored on a blockchain are immutable, i.e., cannot be altered. This makes them suitable for permanently storing asset transactions of property, land, stocks, etc.

- *Incorruptibility*: Records stored on a blockchain are incorruptible, i.e., they cannot be modified by an adversarial party. This makes them suitable for storing records of value, such as money, land records, etc.

- *Consensus*: Blockchains eventually reach consensus with all the honest parties participating in the system. This makes them suitable for creating decentralised, trustless systems.
- *Transparency*: Blockchains enable transparency in record management. This makes them suitable for building trustable and auditable systems.
- *Consistent Ordering*: Records are stored in chronological order and time-stamped.

1.1 Applications of Blockchain Technology

The unique combination of properties in a blockchain enable imaginative applications that are nothing short of a revolution. Let us first see some of the applications of blockchain technology.

1.1.1 Cryptocurrencies and DeFi

One of the most revolutionary applications of blockchain is to have the entire financial system with a decentralised, trustless and transparent alternative. A *cryptocurrency* is a token issued on a blockchain that serves as the currency for performing transactions solely on the blockchain platform. *Decentralised Finance (DeFi)* is a peer-to-peer financial service that operates on top of the blockchain protocol. DeFi builds the entire financial ecosystem on top of the cryptocurrency enabling users to exchange their cryptocurrency, own derivatives, borrow or lend in cryptocurrency denominated loans, etc. Figure 1.1 illustrates the construction of the DeFi stack on top of the blockchain protocol.

The key assumption in building the DeFi stack is that the layers offer a perfect abstraction and operate independently. However, (Daian et al., 2020) demonstrate an example of an application layer flaw percolating to the protocol level and threatening the security of the entire blockchain. Therefore, it is crucial to ensure the game-theoretic soundness of a blockchain protocol. Additionally, the current financial system operates at a breakneck pace, with thousands of transactions every second. For instance, the Visa Network and India's UPI System perform around 1,700 transactions per second (Visa Inc.; upi). For DeFi to replace existing centralised systems, it must offer equivalent performance. Hence, the protocols must scale to high throughputs and offer low *time-to-finality*, i.e., to confirm transactions quickly from the time they are broadcasted.

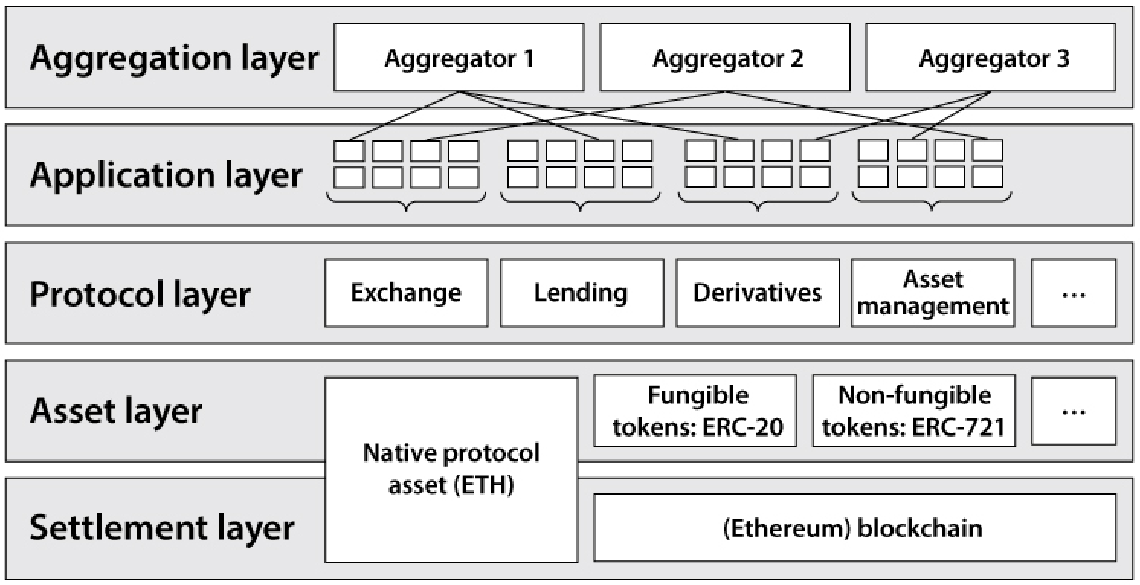

Figure 1.1: Setup of the DeFi Stack (Source: (Schär, 2020))

1.1.2 Central Bank Digital Currency (CBDC)

With most transactions happening online, many countries have realised that it is time to issue currency in a digital form (Auer and Böhme, 2020). CBDCs are cryptocurrencies that are issued and regulated by a central bank. Hence, they are fundamentally different from a decentralised cryptocurrency. CBDCs are extremely efficient for three use cases:

(i) retail payments as a substitute for cash transactions,

(ii) international payments and transfers, and

(iii) efficient exchange of cryptocurrency with fiat currencies.

Every use case associated with a CBDC requires the blockchain to process transactions at a large scale; hence, the protocol must be scalable.

1.1.3 Non-Fungible Tokens (NFTs)

A *Non-Fungible Token* arises from the distinction between regular currency and cryptocurrencies in which tokens are uniquely identifiable, and thus a token can be associated with additional data. An NFT allows a user to establish ownership of rights over a digital asset. They are helpful for artists to reach out to their fans around the world (Wang et al., 2021). For a blockchain to safely store a precious asset like an NFT, the protocol must guarantee security and game-theoretic soundness.

1.1.4 International Remittances

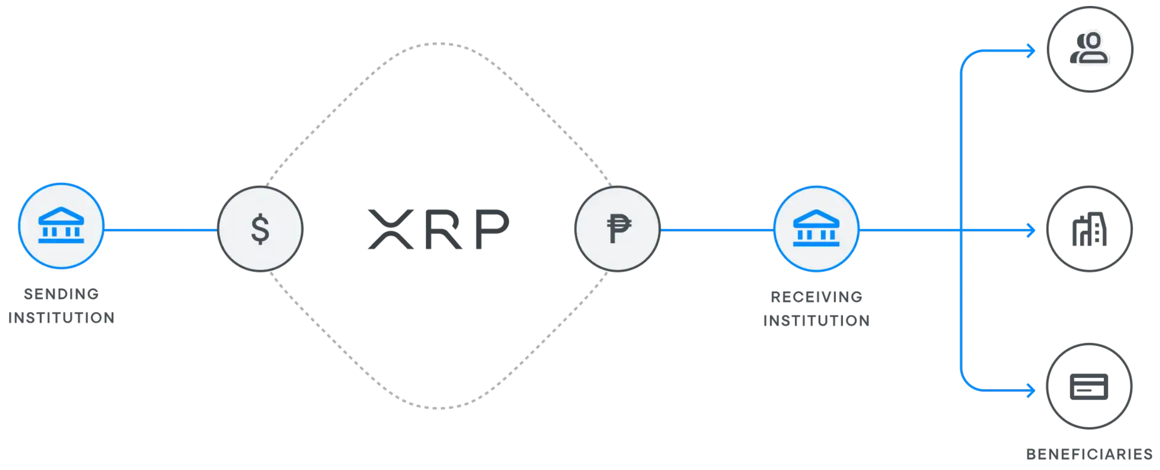

Figure 1.2: Ripple's Settlement System for International Remittances (Source: `ripple.com`)

Typically, transferring money across borders via existing services such as SWIFT is both expensive and slow taking upto 4-5 business days. By coordinating between the banks, blockchains can reduce the friction associated with international remittances. This application has the potential to accelerate global trade but requires the blockchain to be both scalable and highly secure, blockchain-based solutions such as Ripple can reach finality in a few seconds in contrast (Figure 1.2).

1.1.5 e-Voting

Voting is a mechanism for enabling decentralised decision-making, making blockchain the natural platform for conducting e-Voting. In populous countries such as India, the voting process may need input from billions of voters, and hence, the blockchain must scale to match the demand.(Damle et al., 2021) implement voting on blockchains using smart contracts.

1.1.6 Smart Contracts

Smart Contracts are pieces of executable code embedded in real-time on a blockchain, eliminating the need for intermediaries to enforce the code as the contracts. They can be used for identity management, supply chain management, insurance, etc., reducing the probability of fraud. They are written in a Turing-complete language that allows any piece of code to be executed on a shared, trusted computing platform. Given the high stakes associated with some of the use cases, the game-theoretic soundness of blockchain protocols is a must for the widespread deployment of smart contracts.
However, these applications are just the tip of the iceberg with many more to be further explored.

1.2 Challenges

One of the most crucial aspects of blockchain in enabling the desired applications of blockchains is the ability to operate at a large scale and security against adversaries.

1.2.1 Scalability

The biggest challenge faced by existing blockchain protocols is scaling to support high throughputs (Buterin). In a distributed system, scalability refers to the improvement in performance when additional parties join the system. Most of the existing blockchain protocols are not scalable, i.e., the throughput of current blockchain protocols remains the same irrespective of the number of nodes participating in the system. An ideally scalable blockchain protocol should increase its throughput with the number of parties that join the system, i.e., throughput should be $\Omega(n)$ where n refers to the number of parties (Chen et al., 2020). The throughput of existing blockchain protocols like Bitcoin and Ethereum remain constant at around 4-15 transactions per second, irrespective of the number of parties in the system.

1.2.2 Game-Theoretic Soundness

A blockchain relies on the fact that the participating agents are self-serving or rational agents who operate according to the specifications if appropriate incentives are offered. They may further choose to behave strategically to gain more incentive / rewards, which in turn may threaten the security of the blockchain protocol. Hence, a blockchain protocol must be game-theoretically sound to ensure security in a decentralised environment.

1.2.3 Fairness

A key concern of maintaining decentralization is to ensure all participants get equal opportunities to participate in the execution of the system. If the execution favours certain parties, the other parties may deviate from the strategy and behave arbitrarily, threatening a blockchain's security and decentralization aspects.

1.3 The Protocol

Every blockchain application is built on top of the *blockchain protocol*, which is the subject of this thesis. A blockchain protocol, once deployed, is extremely difficult to evolve and modify. Contrary to an algorithm that takes a defined input and produces a defined output which may be treated as a black box, a blockchain protocol is much more critical since it requires the approval of every single stakeholder in the system. A modification to the blockchain protocol is known as a *fork*. For instance, since Bitcoin was initially deployed in 2009, developers have identified several issues in the protocol. Bitcoin has undergone multiple *forks*, many of which failed to garner the approval of all the parties and most of which were abandoned. In contrast, some like the 'SegWit' fork led to splitting users with different versions of the protocol, the 'Bitcoin' and 'Bitcoin Cash. More recently, the Ethereum "Merge" that involved transition from a PoW to PoS protocol, took more than two years to implement. Thus, it is incredibly critical to ensure that the blockchain protocol is secure against today's threats and scalable to meet today's demands. It should also be able to face tomorrow's threats and demands.

The work presented in this thesis aims to provide the imperative foresight for tomorrow's threats and demands and guide the development of scalable blockchain protocols that are both game-theoretically sound and fair.

1.4 Our Contributions and Thesis Outline

With this thesis, we make the following contributions to every chapter:

Chapter 2: Background

We briefly present all the relevant background relevant to the work in this thesis.

Chapter 3: Network Fairness in Blockchains

We discuss our work on Network Fairness in Blockchains.

- In the literature, it is typically assumed that every agent chooses to participate in the blockchain system. To the very best of our knowledge, we are the first to study the blockchain in an environment with asymmetric network access for the participating agents.
- We introduce the notion of *Network Fairness* along with two measures, group fairness, p_f, the probability of frontrunning and, individual fairness, α_f, publishing fairness associated with network fairness.
- We develop a blockchain mining game simulator and present the game-theoretic implications of lack of fairness and show the adverse effect of lack of network fairness on the security of a blockchain protocol.
- We demonstrate some concerning results from the mining game simulator that predict that at the equilibrium, every miner will deviate from the default strategy of the Bitcoin protocol, which could be a potential threat to the security of the blockchain.
- We further extend our analysis to discover a flaw in the OHIE Protocol, a novel blockchain protocol proposed by researchers that claims to scale to desirable throughputs.

Chapter 4: Tiramisu

We generalise the layered approach toward designing blockchain protocols and present a game-theoretic analysis of multi-layered blockchain protocols in the BAR Model (Byzantine, Altruistic and Rational).

- We abstract out the definition of a layered blockchain protocol and develop a framework, *Tiramisu* for game-theoretic analysis.
- We plot the variation of the protocol performance with the protocol parameters in our framework.

- Based on our analysis, we suggest optimal critical security parameters.
- We present a novel blockchain protocol that conforms to the Tiramisu framework, ASHWAChain.

Chapter 5: Game-Theoretically Sound Reward Schemes for PoW Blockchain Protocols

We present a generalized game-theoretic analysis of PoW-based blockchain protocols and highlight the shortcomings in existing reward schemes.

- We show that existing reward schemes, such as the fixed block reward and transaction-fee-only schemes, fail to meet the proposed requirements for game-theoretic soundness for any PoW-based blockchain protocol.
- We present and analyze a class of game-theoretically sound reward schemes, Inflating Block Rewardschemes for any such protocol.

Chapter 6: Conclusion

We discuss further implications of the work presented in the thesis and opportunities for future work to build on top of our work.

Chapter 2

Background

This chapter briefly explains the foundations required for the thesis. We first discuss the notion of distributed consensus and the network model considered in our work. Next, we introduce formal notation associated with the analysis of blockchain protocols and the agent model required for the game-theoretic analysis of blockchain protocols to gradually build up towards the work that is the subject of this thesis.

2.1 Network Model

A blockchain operates via a randomly connected peer-to-peer overlay network of n nodes that different parties may control. There is no structure in this network. The disorder enhances decentralisation and ensures attackers cannot strategically compromise the overlay network and manipulate communication between honest parties. In order to study protocols built on top of networks, we assume three types of network communication models:

Definition 2.1 (Synchronous Model). *There is a fixed delay between the time steps of a node.*

Definition 2.2 (Asynchronous Model). *There can be an arbitrary but bounded delay between the time steps of a node. However, the bound is not fixed.*

Definition 2.3 (Partially Asynchronous Model). *The delay between the time steps of a node is bounded by a fixed constant.*

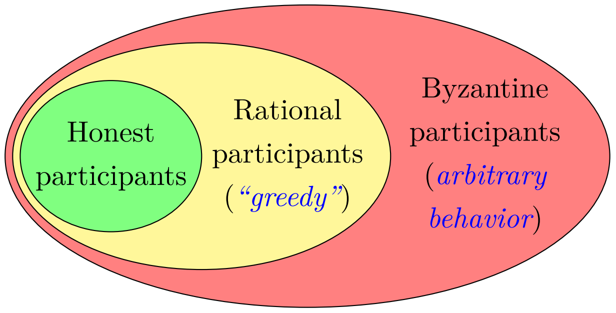
Honest participants
Rational participants ("greedy")
Byzantine participants (arbitrary behavior)

Figure 2.1: Euler diagram of possible behaviors the three types of parties considered

2.2 Agent Model

All parties that participate in the operation of a blockchain may not be honest, i.e., they may have different intentions and incentives that drive their behaviour. Broadly, we may classify parties into three types.

Definition 2.4 (Honest Party). *A party is said to be an* honest party *if and only if it does not deviate from the protocol rules.*

Definition 2.5 (Rational Party). *A party is said to be a* rational party *if it may strategically deviate from the protocol rules if the deviation is expected to yield a higher reward.*

Definition 2.6 (Adversarial Party). *A party is said to be an* adversarial party *(and sometimes also known as "Byzantine" in the literature) who may deviate from the protocol. However, the adversary's goal is to disrupt the operation of the protocol, and it does* ***not*** *try to maximize its reward.*

This model is also often termed as the BAR-model (Byzantine, Altruistic and Rational) in the literature (Aiyer et al., 2005).

2.3 Distributed Consensus

The problem of reaching *agreement* in a distributed system or distributed consensus is widely studied in distributed computing. In essence, *distributed consensus* involves a group of parties trying to agree on the same value(s).

Consider a system of $N \geq 2$ processors $p_1, p_2, \ldots, p_N$ that can communicate by sending messages to one another. Initially, each processor starts with a binary value x_i and eventually, at some time in the computation, all the processes must mutually decide upon a common binary value v.

Definition 2.7 (Distributed Consensus). *A protocol is said to solve the problem of distributed consensus non-trivially if in a setting with n nodes of which at most f might crash, i.e., at least $n - f$ are correct. Node i starts with an input value v_i. The nodes must decide on one of those values, satisfying the following properties:*

1. Termination*: no matter how the system runs, every non-faulty processor decides a finite number of steps,*

2. Agreement*: no matter how the system runs, two different non-faulty processors never decide on different values*

3. Validity*: 0 and 1 are both possible decision values for (possibly different) assignments of initial values. (This condition is needed to avoid the trivial solution where each processor decides 1 regardless of its initial value.)*

If the processors and the communication system are entirely reliable, the existence of consensus protocols is trivial. The problem becomes interesting when the protocol must operate correctly when some processors can be *faulty*. The failure mode studied in this paper is fail-stop, in which a failed processor neither sends nor receives messages. A consensus protocol is t-resilient if it operates correctly when at most t processors fail.

A distributed ledger requires every party[1] to agree on a valid set of transactions to be included permanently in the ledger. Hence, a blockchain aims to achieve distributed consensus in a setting with asynchronous communication that does not guarantee the order of delivery of messages or delivery times and parties that may behave arbitrarily. Unfortunately, Fischer, Lynch and Patterson have shown that achieving all three conditions of distributed consensus in the given setting is impossible (Fischer et al., 1985). Therefore, a blockchain settles for a weaker form of consensus that guarantees *eventual consistency*, i.e., if no new updates are issued, then eventually, the system is in a quiescent state, and the shared state becomes consistent.

[1] which is a node in the above terminology

2.3.1 Practical Byzantine Fault Tolerance (PBFT)

The most widely known and used BFT agreement protocol is the *Practical Byzantine Fault Tolerance* (Castro and Liskov, 1999), commonly known as PBFT. PBFT can achieve distributed consensus in a synchronous setting. One might observe that these protocols can reach consensus with absolute finality fairly quickly if there are fewer participants. However, their speed reduces drastically with every increasing node in the system since the algorithm terminates in $O(n^2)$ in a setting with n nodes.

2.3.2 Blockchain

A blockchain protocol is run perpetually to guarantee eventual consensus. A blockchain protocol relies on a data structure constructed with blocks that are connected via cryptographic hash pointers.

Definition 2.8 (Cryptographic Hash Pointer (H)). *A cryptographic hash pointer H, consists of three parts:*

- *Hash of another block's hash pointer (H')*
- *Data to be referenced (for instance, the root of the merkle tree) (D)*
- *Random nonce (η)*

The hash pointer H is defined as $H = \mathcal{H}(\eta||D||H')$ where $\mathcal{H}$ is a cryptographic hash function.

2.4 The Protocol

A *protocol* is a set of procedures that determine the behaviour of a party. In a blockchain, a protocol determines conflict resolution rules, a reward scheme to incentivise rational parties to follow the protocol and ensures security against byzantine parties. In order to achieve this, a blockchain protocol must derive order from the chaos of an open, decentralised network.

2.4.1 Voting Mechanisms

Voting is a natural solution for deriving order from an open, decentralised network. However, in a blockchain protocol, voting is always implicit in blocks that form structures with hash pointers. For instance, in Bitcoin, the longest chain is selected as the majority chain, giving one block-one vote. However, a party must be selected before being eligible to cast a block or vote by providing proof based on various resources such as Proof-of-Work or Proof-of-Stake.

2.4.2 Mathematical Model Of A Blockchain Protocol

A blockchain $\mathcal{B}$ can be defined by the tuple $(B, \prec, M)$ where B is the set of blocks, $\prec$ is the parent-child relation, M is the mapping of players to the blocks. This abstract model allows us to include all DAG-based blockchain protocols such as IOTA (Popov), SPECTRE (Sompolinsky et al., 2016), etc. Formally,

- $B = \{b_0, b_1, \ldots, \}$ is finite collection of blocks where each block is said to be valid *iff* it has a hash value less than the target.
- $\prec$ is a relation over B that defines the ordering of the blocks. $\prec$ is defined as follows:
    1. if y contains a hash pointer to x, then we say $x \prec y$, and
    2. $x \prec y$ and $y \prec z \Rightarrow x \prec z$, i.e., $\prec$ is a transitive relation.
- $M : B \to \mathcal{P}$ is the function that maps blocks to players, $\mathcal{P}$.

2.4.2.1 Proof-of-Work (PoW)

Proof-of-Work is a selection mechanism that allows a party to prove to another party that a certain computationally challenging problem has been solved by the prover. For instance, Bitcoin requires the prover to find a nonce that, along with the block header, produces a SHA256 hash value smaller than the predetermined target. Due to the nature of the hash function, this is a random process, and miners must repeatedly sample different nonce values to mine a block successfully.

Definition 2.9 (Cryptographic Puzzle). *Formally, a function $\mathcal{F}_d(c, x) \to \{\texttt{true}, \texttt{false}\}$ where d is the difficulty, c the challenge and x a nonce can be used for PoW iff it has the two properties: (i) $\mathcal{F}_d(c, x)$ is fast to compute if d, c and x are known and (ii) given d and c it is computationally intractible to find x. This ensures every miner has to brute-force multiple nonce values in order to find a valid x such that $\mathcal{F}_d(c, x)$ returns $\texttt{true}$.*

Probabilistic Analysis We now determine the probability of a group of parties trying to mine in a PoW blockchain.

Definition 2.10 (Fail Function). *We define fail(ϕ, t) as the probability of ϕ fraction of the network failing to mine a block in t units of time.*

Let us denote p to be the probability of a nonce being successful and q to be the rate of queries to the hash function. In time t, the miner will be able to make $q \times t$ such queries. As the probability of a query being unsuccessful will be $(1 - p)$. The probability of $q \times t$ such queries being unsuccessful will be $(1 - p)^{q \times t}$.
Thus, for n players (the total number of players in the system), the number of queries in the same duration will be $n \times q \times t$. The probability of these nodes failing to mine a block will be $(1 - p)^{n \times q \times t}$. The value $n \times q$ is known as the hash rate of the system, denoted by H. Let us denote the probability of ϕ fraction of the network failing to mine a block by the function $\text{fail}(\phi, t) = (1 - p)^{\phi \times nqt} = (1 - p)^{\phi \times Ht}$.
For convenience, we sometimes use $\text{fail}(\phi) := \text{fail}(\phi, 1)$, the probability of ϕ fraction of the network failing to mine a block in a unit of time.

2.4.2.2 Proof-of-Stake (PoS)

PoW requires massive amounts of computing power and, thus, electricity. Proof-of-Stake offers a way to set up a selection mechanism without requiring as much energy. In PoS, the parties run a lottery based on their stake of the cryptocurrency. Those who win this lottery are selected to cast a block or vote. The Ouroboros Protocol (Kiayias et al., 2017; David et al., 2018; Badertscher et al., 2018b), and Algorand (Gilad et al., 2017) are widely studied examples of PoS protocols.

2.4.2.3 Proof-of-X

Researchers have further developed multiple mechanisms suitable for deploying in different settings of blockchain protocol. For instance, Proof-of-Context is a selection mechanism designed for ad-hoc networks and operates on computationally constrained mobile devices (Chatzopoulos et al., 2021). Many researchers have further developed mechanisms to solve practical computational problems using Proof-of-Work, e.g., generation of Vanity Tor URLs and Bitcoin addresses (Chaurasia et al., 2021), deep learning (Lan et al., 2021), archival storage (Miller et al., 2014), etc.

One of the critical limitations of selection mechanisms is that they need to limit the voting rate to ensure that the network has time to synchronise. If the network goes out of sync, the network may partition into inconsistent states since the blockchain protocol only guarantees eventual consistency. This leads to throttling of throughput in a blockchain.

2.4.3 Scalability Trilemma

The throughput of existing blockchain protocols is being throttled to ensure security. However, this leaves much to be desired in terms of performance. The Visa Network and India's UPI System perform around 1,700 transactions per second which is a hundred times more than what existing blockchain protocols such as Bitcoin and Ethereum support. In an ideal distributed system, the system's performance increases linearly ($\Omega(n)$) concerning the number of parties participating in the system. However, existing blockchain protocols do not scale with the number of parties joining the system. This problem is known as the *Speed-Security Tradeoff* in Blockchain Protocols.

If we also consider centralised distributed ledgers such as traditional databases that offer scalability the tradeoff resolves into the *scalability trilemma* between decentralisation, scalability and security.

2.5 Scalability Related Papers

(Garay et al., 2014) and (Kiayias and Panagiotakos, 2015) show that existing blockchain protocols such as Bitcoin suffer from a loss of security properties as we scale the system. Therefore, scalability has been a consistent concern among blockchain protocol designers over the last few years. This has led to various designs proposed to create scalable and secure blockchains. Two major approaches to designing such protocols are (i) Linear blockchain protocols and (ii) Non-linear blockchain protocols. Linear blockchain protocols are structurally similar to Nakomoto consensus, which resembles a linear chain of blocks that grows with time, while Non-linear blockchain protocols adopt a DAG-based architecture (Directed Acyclic Graph). The most distinctive design feature is the ability to include conflicting blocks in the main chain. While non-linear blockchain protocols can include conflicting blocks in the main chain, a linear blockchain protocol only includes non-conflicting blocks in the main chain by orphaning the rest of the blocks. Thus, non-linear blockchain protocols require an explicit block ordering algorithm while this order is implicit in linear blockchains. Notice that linear blockchains are a sub-class of non-linear blockchain protocols.

2.5.1 Linear Blockchain Protocols

Bitcoin-NG (Eyal et al., 2016) and GHOST (Sompolinsky and Zohar, 2013) are some well-known examples of linear blockchain protocols apart from the original Nakomoto consensus. Interlude also falls in this category. Thus, our design retains this feature from the original Bitcoin design. Linear blockchain protocols are typically challenging to scale due to the formation of forks at high block creation rates or large block sizes.

2.5.2 Non-linear Blockchain Protocols

(Eyal et al., 2016; Sompolinsky and Zohar, 2013; Sompolinsky et al., 2016, 2018; Li et al., 2020; Popov) are examples of some non-linear blockchain protocols. (Chen et al., 2021) use game theory to establish the tradeoff between full verification, scalability, and finality-duration in non-linear blockchains. Their result relies on the computational complexity of verifying a blockchain, while in most distributed systems, the message complexity is considered a more important factor. Hence, their bounds are more than optimistic for most practical applications of blockchain protocol. Their result is interesting since it describes the limit of what these blockchain protocols can achieve. These blockchain protocols can be further categorized into subclasses based on their design features (non-exhaustive).

2.5.2.1 Combining Multiple Chains.

One of the techniques to scale blockchains is to use multiple instances of Nakomoto consensus that parties mine in parallel while linking the chains in a manner that allows PoW in one chain and verify blocks in another chain. Thus, by sharing PoW across the instances, we can distribute mining across the chains without dividing the security. OHIE (Yu et al., 2020) and Chainweb (Martino et al., 2018) use this approach.

2.5.2.2 Sharding.

Another common technique is to use the “divide-and-conquer” paradigm. The parties are distributed across shards with some intersection between the shards. These protocols are typically more complex and do not achieve full verification. Since a transaction is only verified by a subset of parties in a shard, these protocols do not achieve full verification. Ethereum 2.0 (eth, 2021), OmniLedger (Kokoris-Kogias et al., 2018), RapidChain (Zamani et al., 2018), and PolyShard (Li et al., 2021) are examples of such protocols. The non-linear or DAG-based approach suffers from the following drawbacks:

1. Vulnerable to *Balance Attacks* wherein an adversary may partition the network into balanced subgraphs and start mining on one subgraph to invalidate another that contains an accepted transaction (Wang et al., 2020). Interlude is immune to this attack since a party can choose to mine on only one chain as compared to a non-linear blockchain allowing the party to mine on top of *all* the chains at once.

2. Few blockchain protocols such as SPECTRE (Sompolinsky et al., 2016) and IOTA (Popov) can only guarantee pairwise ordering of transactions which may lead to Condorcet cycles in transaction ordering.

We further refer the reader to a comprehensive survey of blockchain-based consensus protocols by (Bano et al., 2019).

2.6 Game-Theory Preliminaries

Blockchain is a distributed ledger that uses a mechanism design to ensure correct functioning. Unlike the Visa Network or centralized banking systems, a blockchain is maintained by a decentralized network of nodes. Instead of using a central authority, it incentivizes participants to run and secure the underlying consensus protocol. Thus, it becomes crucial to ensure that the nodes that maintain the ledger are appropriately incentivized to prevent them from behaving maliciously.

The reward schemes employed in today's blockchain protocols do not guarantee game-theoretic soundness that may leave space for strategic deviations. For instance, (Carlsten et al., 2016c) had shown that when there is a large variance in the fees earned from transactions, it might be profitable to fork blocks containing transactions with high fees.

2.6.1 Game Theory Notation

All the parties that maintain a blockchain system in exchange for a reward can be considered as players in a game. A game Γ is formally defined as a tuple $\Gamma = \langle N, (S_i)_{i \in N}, (u_i)_{i \in N} \rangle$ where N is the set of players, S_i is the strategy set for the player i and $u_i : S_1 \times S_2 \times \cdots \times S_n \to \mathbb{R}$ for $i = 1, 2, \ldots, n$ is the corresponding utility function for each player.

2.6.2 Reward Model for Blockchain Protocols

There are two principal ways of rewarding the nodes participating in a blockchain system:

1. *Block Reward* - The reward parties can assign to themselves for mining a block which the protocol designers can fix. This reward mints new currency, adding to the total amount of currency in circulation. We denote the block reward associated with a block $b \in B$ as $\text{BlockReward}(b)$.

2. *Transaction Fees* - A user can optionally offer a transaction fee to the party that includes the transaction into the blockchain. Typically, the users are allowed to decide the transaction fees they wish to offer while creating a transaction. Although protocol designers may enforce a minimum transaction fee (e.g. BitcoinF (Siddiqui et al., 2020a)), they cannot ensure that there are enough unconfirmed transactions to include in a new block. We denote the transaction fees associated with a block b as $\text{Fee}(b)$.

We further assume that the blockchain protocol additionally specifies a reward function $\mathcal{R} : B \to \mathbb{R}$ that captures both the transaction fees associated with a block and the block reward, hence the function $\mathcal{R}(b) = \text{BlockReward}(b) + \text{Fee}(b)$ specifies the reward assigned to the players.

2.6.3 Solution Concepts

A *solution concept* in game-theory refers to finding a set of strategies in the game that mutually assure each player the greatest reward. There are multiple solution concepts in game theory with varying degrees of conformity. In this work we rely on the Weakly Dominated Strategy Equilibrium and the Mixed Strategy Nash Equilibrium.

Definition 2.11 (Weakly Dominated Strategy). *(Osborne and Rubinstein, 1994) A strategy $s_i \in S_i$ is said to be weakly dominated if $\exists\ s'_i \in S_i$ for an agent i if*
$u_i(s'_i, s_{-i}) \geq u_i(s_i, s_{-i})\ \forall s_{-i} \in S_{-i}$ *and* $u_i(s'_i, s_{-i}) > u_i(s_i, s_{-i})$ *for some* $s_{-i} \in S_{-i}$ *where $u_i(s_i, s_{-i})$ is the payoff for agent i if he/she chooses the strategy s_i and the other agents' vector of strategies is s_{-i}*

Definition 2.12 (Weakly Dominant Strategy). *(Osborne and Rubinstein, 1994) A strategy s_i^* is said to be weakly dominant strategy for an agent i if it weakly dominates* *every other strategy $s_i \in S_i$.*

Definition 2.13 (**Weakly Dominant Strategy Equilibrium**). *Given a game $\Gamma = \langle N, (S_i), (u_i) \rangle$, a strategy profile $(s_1^*, \ldots, s_n^*)$ is called a weakly dominant strategy equilibrium if, $\forall i = 1, \ldots, n$, the strategy s_i^* is a weakly dominant for agent i*

Definition 2.14 (**Pure Strategy Nash Equilibrium**). *A strategy $s^* = (s_1^*, s_2^*, \ldots, s_n^*)$ is called a Pure Strategy Nash Equilibrium (PSNE) for n agents, if for each agent i, s_i^* is the best response to s_{-i}^*. That is, $\forall i \in n$,*

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*), \forall s_i \in S_i$$

where S_i is the set of strategies available to the i^{th} player.

Definition 2.15 (**Mixed Strategy Nash Equilibrium**). *(Osborne and Rubinstein, 1994) A strategy profile $(\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*)$ is called a Mixed Strategy Nash Equilibrium (MSNE) for n agents, if for each agent i, σ_i^* is the best response to σ_{-i}^*. That is, $\forall i \in n$,*

$$u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(\sigma_i, \sigma_{-i}^*), \forall \sigma_i \in \Delta(S_i)$$

where $\sigma_i = (p_1^i, p_2^i, \ldots, p_k^i)$ is the mixed strategy played by the player i in which he/she chooses strategy s_j with probability p_j^i.

Iterated Removal of Dominated Strategies

Iterated removal of dominated strategies is a technique to find solutions to a strategic-form game by iteratively discarding the set of strategies that cannot be part of the solution sets.

Algorithm 1: Iterated Dominance

```
Input: $\Gamma =< N, (S_i)_{i \in N}, (u_i)_{i \in N} >$
Output: $\Gamma' =< N, (Z_i)_{i \in N}, (u_i)_{i \in N} >$
for $i = 1 \rightarrow n$ do
    $Z_i^0 = S_i$
$k = 1$
flag=true
while flag do
    flag=false
    for $i = 1 \rightarrow n$ do
        $Z_i^k = Z_i^{k-1}$
        if $\exists s_i$ dominated by some other strategy then
            $Z_i^k = Z_i^k \setminus s_i$
            flag=true;
    $k = k + 1$
for $i = 1 \rightarrow n$ do
    $Z_i = Z_i^{k-1}$
```

2.7 Game-Theoretic Issues in Blockchain

Blockchain is a distributed ledger that uses mechanism design to ensure correct functioning. Unlike the Visa Network or centralized banking systems, a blockchain is maintained by a decentralized network of nodes. Instead of using a central authority, it incentivizes parties or nodes to run and secure the underlying consensus protocol. Thus, it becomes crucial to ensure that the nodes that maintain the ledger are appropriately incentivized in order to prevent them from behaving maliciously. Liu et al. (2019) provide an overview of the usage of game theory in blockchains.

However, many blockchain protocols come with assumptions that may not accurately reflect the real world. For instance, many protocols require an honest majority rather than a rational one, which may leave them susceptible to incentive-driven deviations and other game-theoretic issues (Grunspan and Perez-Marco, 2020; Neuder et al., 2019; Feng and Niu, 2019; Ritz and Zugenmaier, 2018). Zhang et al. highlight the importance of game-theoretic soundness by demonstrating wide classes of attacks that affect most of these protocols (Zhang and Preneel, 2019).

2.7.1 Game-Theoretic Attacks

2.7.1.1 Incentive-Driven Deviations

A PoW-based blockchain protocol is deployed in an environment with self-interested or rational players wishing to maximise utility. In such a case, it may be possible that some alternative mining strategy may yield a greater reward than following the honest strategy prescribed by the blockchain protocol. For instance, Undercutting or similar attacks.

2.7.1.1.1 Petty mining (Carlsten et al., 2016a) is a strategy in which, given a fork, the petty player picks the fork, which has collected lesser transaction fees and, hence, offers the player an opportunity to include transactions from the other fork and collect more transaction fee.

2.7.1.1.2 Undercutting (Carlsten et al., 2016a) is a strategy in which a player intentionally forks a block with a high reward in order to collect some of the rewards while offering the rest of the reward to the petty players that choose to mine on top of it. Formally, given two maximal blocks b_1 and b_2, a petty player chooses to mine on the block, offering greater transaction fees. Similarly, the undercutting strategy involves mining on a non-maximal block and collecting greater transaction fees.

2.7.1.1.3 Selfish Mining Formally, in a selfish mining strategy, the player does not reveal his/her blocks immediately and keeps mining on a secret chain B' while other nodes do not have access to at least one block in B'. The attack was first described by (Eyal and Sirer, 2014) and involves an attacker mining secretly on top of his/her private chain until the public chain catches up when the attacker broadcasts the private chain. Assuming that the attacker is able to propagate his/her chain faster than the rest of the network, the attacker would be able to grab more than his/her fair share of the reward. Various variants of this attack strategy have been studied by many researchers (Nayak et al., 2016; Eyal and Sirer, 2014; Sapirshtein et al., 2016b).

2.7.1.2 Goldfinger Attack

Kroll et al. (2013) was the first to describe the Goldfinger Attack in which, the attacker bears a high cost of a double-spending attack by purchasing specialised mining equipment and paying electricity costs. The security of a PoW blockchain stems from the belief that parties, without any malicious intent, control the majority of the computational power in the network. A blockchain vulnerable to such an attack would not be able to provide any value to the cryptotoken since no merchant would be willing to exchange goods for the cryptotoken, fearing a double-spend attack rendering the cryptotoken worthless. The attacker subsequently profits from the economic downfall of the cryptotoken by holding short positions on the cryptotoken and related commodities prior to the attack. Performing such an attack should incur an extremely high cost which should outweigh any potential profit the attacker can make. Hence, a rational party would prefer not to attack. Any adversary that wishes to attack the blockchain by performing a double-spending attack would need to acquire at least the same amount of computational power as the honest majority.

Goldfinger Attack differs from other Incentive-Driven Deviations is because the attacker derives his/her utility from outside the system rather than in the form of cryptotoken inside the system. Such an attacker will have an intent to disrupt the system, while the attacker performing incentive-driven deviations only intends to grab a greater mining reward.

2.7.2 Game-Theory Related Papers

Since game theory and Mechanism Design are fundamental to the suitable operation of a PoW-based blockchain protocol, many researchers have analyzed these protocols from the lens of game theory. Researchers have discovered many game-theoretic attacks, for instance, Undercutting (Carlsten et al., 2016a) and Selfish Mining (Eyal and Sirer, 2014). (Alkalay-Houlihan and Shah, 2019) study the feasibility of block withholding attacks. (Hou et al., 2021) employ reinforcement learning techniques to discover incentive-driven deviations in blockchain protocols. Participants in a PoW-based blockchain protocol often form coalitions known as Mining Pools. (Fisch et al., 2017) use game theory to analyze various reward schemes such mining pools adopt. (Siddiqui et al., 2020a) study the transaction-fee-only reward model.

Many papers have explored both (i) attacks on Bitcoin's incentive mechanism (Sapirshtein et al., 2016b; Eyal, 2015; Carlsten et al., 2016a; Liao and Katz, 2017; Nayak et al.,

2016) and (ii) attacks on other blockchain protocols (Grunspan and Perez-Marco, 2020; Neuder et al., 2019; Feng and Niu, 2019; Ritz and Zugenmaier, 2018). In (Zhang and Preneel, 2019), the authors highlight the lack of systematic game-theoretic analysis of recently proposed blockchain protocols. Often, these attacks not only negatively impact the revenue of other parties but also the security of the blockchain against byzantine adversaries. In fact, the claimed scalability results may not be achievable at the game-theoretic equilibrium. Hence, it is pertinent to develop a reward scheme ensuring the scalability and security claims at equilibrium. Towards this, (Badertscher et al., 2018a) use the Rational Protocol Design paradigm to analyze Bitcoin from a lens of game theory. We extend their analysis for incentive-driven deviations to an abstract model of PoW-based blockchains making our results applicable to any general PoW-based protocol.

2.8 Fairness in Blockchains

In a blockchain protocol, each node must be incentivized to act honestly. To ensure this not only the network of nodes as a whole is provided enough incentives, but each node in the network should also be provided the correct incentive.

Fairness related papers

(Pass and Shi, 2017b) defined η-approximate fairness as follows [2]:

Definition 2.16 (**η-approximate fairness**). *(Pass and Shi, 2017b) A blockchain protocol has η-approximate fairness if, with overwhelming probability, any honest subset controlling ϕ fraction of the compute power is guaranteed to get at least a $(1-\eta)\phi$ fraction of the blocks in a sufficiently long window.*

The intuition behind this is that in a fair protocol, an agent that controls a ϕ fraction of the computational resources should receive a ϕ fraction of the rewards. Though η-approximate fairness has its own merit and importance, the definition manages to capture the intuition only if the reward for each block is similar.

[2] In their original paper, Pass et al. used the term δ-approximate fairness however we use a different symbol to distinguish it from the δ delay we use in Chapter 3

Chapter 3

Network Fairness and Scalability in Blockchains

In this chapter, we study Proof-of-Work Blockchains in an environment with asymmetric network access and introduce the concept of network fairness*. As we increase the throughput of the DL system, the underlying peer-to-peer network might face multiple challenges in keeping up with the requirements. Due to varying network capacities, the slower nodes would be at a relative disadvantage compared to the faster ones, which could negatively impact their revenue. In order to quantify their relative advantage or disadvantage, we introduce two measures of network fairness, p_f, the probability of frontrunning and α_f, the publishing fairness. We show that these measures deteriorate as we scale the blockchain, implying that the slower nodes are disadvantaged at higher throughputs. It results in the faster nodes getting more than their fair share of the reward, while the slower nodes (slow in terms of network quality) get less. Thus, fairness and scalability in blockchain systems do not go hand in hand.*

In a setting with rational miners, lack of fairness causes miners to deviate from the "longest chain rule" or undercut*, which would reduce the blockchain's resilience against byzantine adversaries. Hence, fairness is not only a desirable property for a blockchain system but also essential for the security of the blockchain, and any scalable blockchain protocol proposed must ensure fairness.*

We further extend our analysis to demonstrate the negative implications of lack of network fairness and strategic deviations on a contemporary blockchain protocol known as OHIE.

3.1 Different Notions of Fairness

3.1.1 η-approximate fairness

Pass and Shi (Pass and Shi, 2017b) defined η-approximate fairness as follows [1]:

Definition 3.1 (η-approximate fairness). *(Pass and Shi, 2017b) A blockchain protocol has η-approximate fairness if, with overwhelming probability, any honest subset controlling ϕ fraction of the compute power is guaranteed to get at least a $(1-\eta)\phi$ fraction of the blocks in a sufficiently long window.*

The intuition behind this is that in a fair protocol, an agent that controls a ϕ fraction of the computational resources should receive a ϕ fraction of the rewards. Though η-approximate fairness has its own merit and importance, the definition manages to capture the intuition only if the reward for each block is similar. This variation may not be significant in DLs that operate at a slower speed. However, the block reward may vary significantly among the blocks at higher throughputs. In this case, we should also factor in the agents' rewards for their blocks. Secondly, this definition does not capture the disparity among different nodes, i.e., which nodes gain more than their fair share and which nodes get less. The measures we define compare different nodes or sets of nodes to highlight which ones are at a relative advantage and which are at a disadvantage.
Here, we wish to analyze network fairness and establish measures independent of the computational power of the nodes we are comparing. Hence, we base our definitions on network events instead.

3.1.2 Frontrunning (p_f)

In some cases, it might be easier to analyze and quantify "unfairness" rather than fairness. Since only the transaction that comes first is said to be the valid one, any subsequent blocks that include a copy of the transaction lose out on the transaction fees and waste

[1] In their original paper, Pass et al. used the term δ-approximate fairness however we use a different symbol to distinguish it from the δ delay we use throughout the paper

their space, which could have accommodated an unconfirmed transaction. For every transaction, we can imagine a race among the nodes to grab its transaction fees by mining a block that includes it. We can say that the node that manages to win the race by mining a block containing the transaction before everyone else wins the race and has successfully *frontrun* everyone else[2][3]. The system will be fair if every node has an equal probability of winning the race.

We define the event `frontrun_1` for a node L (L here, stands for Loser) as the event in which he loses the race described earlier. In this case, the node does not gain any reward since he failed to mine the block before everyone else.
We define the event `frontrun_2` for L as the event in which some other node manages to win the race even before L starts. Some other node mined a block containing the transaction before L receives the transaction. Clearly, `frontrun_2` $\subseteq$ `frontrun_1`.

To capture it more formally,

Definition 3.2 **(p_f).** *We call $\{p_f\}_m^M$ the probability of the event* `frontrun_2` *between the top M percentile of the nodes and the bottom $1-m$ percentile of the nodes in terms of network delays. That is the probability that some node in the top M percentile (in terms of network speed) manages to* `frontrun_2` *all nodes in the bottom $1-m$ percentile.*

Ideally, $P(\texttt{frontrun\_2}) = 0$ since all nodes should start at the same time (this being a race). However, as we analyze in Section 3.2.1, this probability may become significantly high due to varying network speeds.

3.1.3 Publishing Fairness (α_f)

Faster internet speed can provide an advantage in receiving new transactions and yield an upper hand in broadcasting blocks. Consider two nodes, A and B, that mine a block simultaneously. We define α_f as the ratio of the probability that the majority accepts A's block to the probability that the majority accepts B's block. Thus, it quantifies A's

[2]As we discuss in later sections, this may not be the only way to win the race. In some cases, agents can change the results of the race by *strategically* deviating from the protocol.

[3]We borrow the term from Wall Street jargon, where the term originates from the era when stock market trades were executed via paper carried by hand between trading desks. The routine business of hand-carrying client orders between desks would normally proceed at a walking pace. However, a broker could run before the walking traffic to reach the desk and execute his order first. (Wikipedia contributors, 2020)

advantage in publishing a block and claiming the associated reward. A lack of publishing fairness implies that slower nodes are less likely to receive reward transaction fees in the mined block and less likely to receive the fixed block reward associated with mining a new block. The intuition behind this definition is that over multiple rounds, this would be the ratio of their conflicting blocks getting accepted.

$$\alpha_f(A,B) = \frac{P(\frac{\text{A's block getting accepted}}{\text{A and B mine a block simultaneously}})}{P(\frac{\text{B's block getting accepted}}{\text{A and B mine a block simultaneously}})}$$

Remark 1 Like p_f, α_f is also a measure of "unfairness" rather than fairness, i.e., higher α_f implies lesser fairness in the system. The ideal value of α_f is 1, when the blocks mined simultaneously by both the nodes are equally likely to get accepted.

Remark 2 α_f only accounts for the events in which both A and B mine their blocks simultaneously since both the probabilities are conditioned on the blocks being mined simultaneously.

Remark 3 Ideally α_f should be close to 1. The high value of α_f indicates that the faster agent can collect more block rewards than the slower agent even though they both do exert the same amount of computation.

3.2 Analyzing Network Fairness

A node that can hear new transactions and propagate blocks faster than other nodes gains an unfair advantage over its peers. In this section, we analyze and quantify the advantage. We believe that the results presented in 3.2.1 and 3.2.2 to be a significant insight offered by our paper.

3.2.1 Analyzing Frontrunning

Consider a node with a poor network connection that receives a transaction with a significant delay compared to others that do not. We now analyze the probability of the event `frontrun_2` happening for the node. For the Bitcoin Network, according to (bit, 2017) it takes less than 4 seconds for a transaction to reach the 50^{th} percentile but more than 15 seconds to reach the 90^{th} percentile. It is likely that the slower nodes only hear about a transaction once a faster node has already confirmed it. Let us consider the

probability of 50% nodes in the top percentile in terms of network speed being able to confirm the transaction before the bottom 10% of the nodes receive it.[4]
Let d be the advantage offered to the top M fraction of the nodes in terms of time (in our example, this amounts to 11 seconds), p is the probability of query being successful, and H be the hash rate of the network.

Theorem 3.1 (Lower bound of $\{p_f\}_m^M$). $\{p_f\}_m^M > M\lambda d - \frac{1}{2}\big(M\lambda d\big)^2$

Proof. We show that the probability of the event that M fraction of the network manages to mine a block in time d increases with the block creation rate λ. Let `frontrun_2`(M, m) denote the event that top M fraction of nodes succeed in mining the block before the transaction reaches the bottom $1 - m$ fraction.

$$\begin{aligned}
&P(\texttt{frontrun\_2}(M, m)) \\
&= 1 - P(M \text{ fraction of nodes fail to mine a block in time } d) \\
&= \ 1 - \text{fail}(M, d) \\
&= 1 - (1-p)^{M \times Hd} \\
&> pMHd - \frac{1}{2}\bigg(pMHd\bigg)^2 \\
&= M\lambda d - \frac{1}{2}\bigg(M\lambda d\bigg)^2
\end{aligned}$$

□

Theorem 3.1 shows that $\{p_f\}_m^M$ increases monotonically with increasing the block creation rate λ since its lower bound increases monotonically. One can observe that in order to keep the probability of this event sufficiently low, d must be reduced while increasing λ to scale the blockchain. Although it may seem that the lower bound is independent of the bottom m percentile selected, this would have been incorporated in d since d increases as m decreases.

Practical Significance of Theorem 3.1

As of October 2020, $\{p_f\}_{0.9}^{0.5}$ approximates to 0.01 for the Bitcoin Network. However, suppose we were to scale the Bitcoin Network to a throughput similar to that offered by the

[4]We make an additional assumption here that all nodes should have equal computing power. This is ideal from the expectation that the system should be decentralized. Hence, computational power should be ideally distributed equally among the nodes.

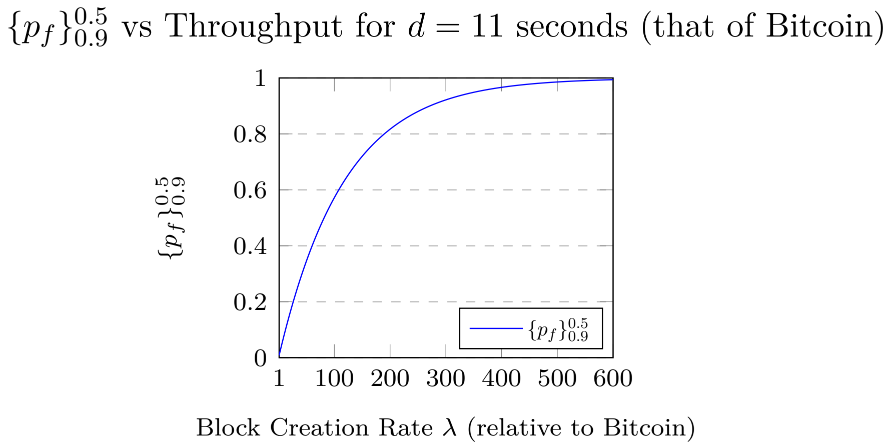

Figure 3.1: Variation of $\{p_f\}^{0.5}_{0.9}$ as we scale the blockchain

likes of Visa Network by increasing the block creation rate to 566 blocks every 10 minutes (by reducing the difficulty and keeping the block size the same). In that case, $\{p_f\}^{0.5}_{0.9}$ goes up to 0.99 [5]. Unfortunately, since, at the time of writing, statistics on transaction propagation times in Ethereum were not published, we estimate $\{p_f\}^{0.5}_{0.9}$ for Ethereum by using the same delays as Bitcoin. We find that the estimated value of $\{p_f\}^{0.5}_{0.9}$ (as of October 2020) is around 0.36, which is considerably higher than that of Bitcoin. In Figure 3.1, we plot the variation of $\{p_f\}^{0.5}_{0.9}$ with increase in the throughput.

Variation in Block Rewards

Our analysis in Section 3.2.1 shows that as we scale the blockchain to higher throughputs, some agents would be able to grab high-value transactions before others consistently. This means that some agents would produce blocks with higher rewards, while others would produce blocks with lower rewards. Hence, a lack of network fairness may be able to induce a greater variation in block rewards. We discuss further implications of this variation in Section 3.3.

Lack of incentive for information propagation: The inability of the peer-to-peer network to keep up with the ledger's desired throughput is further exacerbated by the fact that the nodes do not have any incentive to participate in broadcasting information. In fact, they are incentivised to keep the knowledge of transactions to themselves, as shown by Babaioff et al. (Babaioff et al., 2012). By broadcasting a transaction to other nodes,

[5]We assume there are no significant improvements in the peer-to-peer overlay network

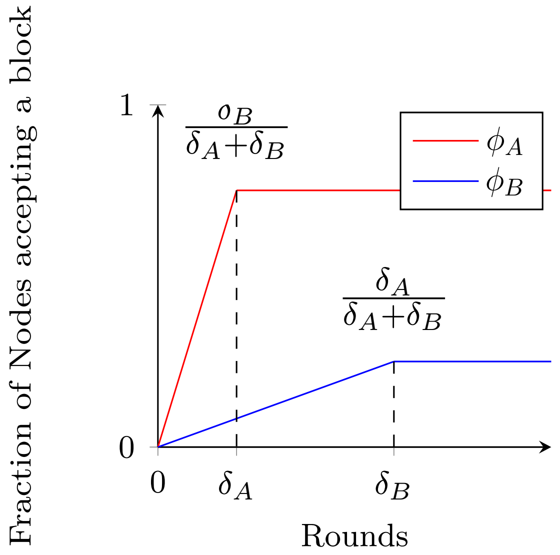

Figure 3.2: Fraction of nodes accepting a block vs rounds

an agent potentially increases the number of nodes competing to include the transaction in their blocks and collect the corresponding transaction fees. Thus, offering an additional incentive to agents for propagating transactions may speed up the broadcast of a transaction.

3.2.2 Publishing Fairness in Broadcasting Blocks

Let us assume that the execution happens in “rounds” in which the nodes make q queries each to the Hash Function. At the boundaries of rounds, the nodes can communicate with their neighboring nodes. In our example here, we assume the duration of a round to be 1 second. If a node succeeds in mining a block in a round, it will begin broadcasting it to its neighbors when the round ends. Similarly, if a node receives a block at the beginning of a round, it will broadcast it to its neighbors at the end of that round.

We assume that there are n nodes, and all nodes are honest. Hence, they follow the Bitcoin protocol's strategy of picking the oldest block in case of a tie and publishing a block as soon as it is mined.

Consider the event in which exactly two nodes, A and B mine a block simultaneously in the same round. Let δ_A and δ_B ($\delta_A \leq \delta_B$, without loss of generality) be the delay associated with broadcasting the blocks to the entire network. At a time $t > \delta_B$, the network will be split into two fractions:

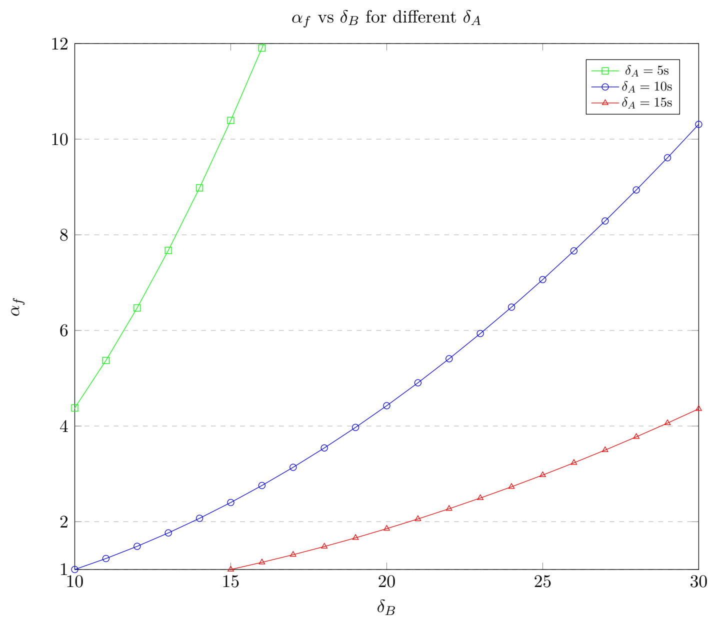

Figure 3.3: Variation of α_f with varying network speeds
(λ = 566x Bitcoin Throughput)

- ϕ_A: The fraction of nodes in the network which claim to have received the block mined by A before the block mined by B.
- ϕ_B: The other fraction of nodes in the network which claim to have received the block mined by B before the block mined by A.

Since by δ_B, all nodes in the network would have received the blocks mined by A and B, $\phi_A + \phi_B = 1$. Let ϕ_A^i and ϕ_B^i be the fraction of network accepting A and B at the i^{th} round. Then, α_f can be approximated by Theorem 3.2.

Theorem 3.2 (Approximation of α_f).

$$\alpha_f = \frac{\psi}{1-\psi} \tag{3.1}$$

where
$$\psi = \sum_{i=1}^{\infty}\left[\prod_{j=0}^{i-1}[(1-fail(\phi_A^j))(1-fail(\phi_B^j)) + fail(\phi_A^j)fail(\phi_B^j)] \times (1-fail(\phi_A^i))fail(\phi_B^i)\right]$$

Proof. The proof of follows from first finding the probability of A being successful to getting its block accepted in i^{th} round conditioned on the probability that neither of the blocks

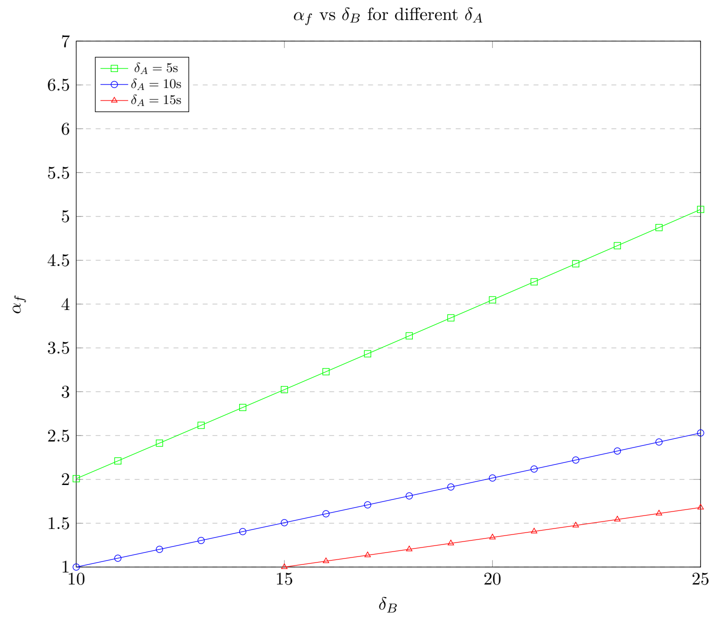

Figure 3.4: Variation of α_f with varying network speeds (λ = 1x Bitcoin Throughput)

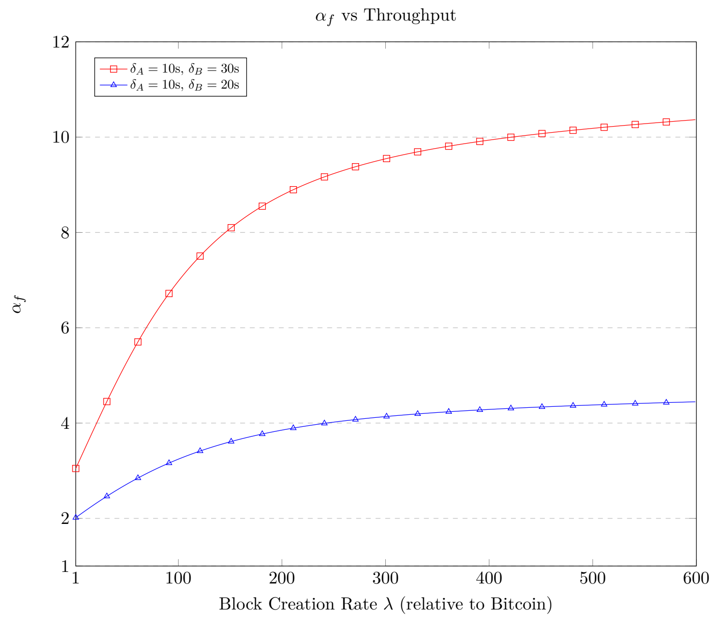

Figure 3.5: Variation of α_f as we scale the blockchain

gain majority till the $(i-1)^{\text{th}}$ round. We then use *Baye's Theorem* to find out the total probability.

The block mined by A will get accepted if the chain that includes A's block becomes longer than the one that includes B. As soon as the longer chain is received by a node that had previously accepted B, the node must reject B and accept A instead. We assume that as soon as the chain mined by either fraction becomes longer than that of their counterpart, their counterpart will switch to the longer chain. Let ϕ_A^i and ϕ_B^i be the fraction of network accepting A and B at the i^{th} round. We slightly abuse notation here to use A and B to refer to the blocks mined by A and B, respectively.

$$P\Big(\frac{\text{A gets accepted in } i^{\text{th}} \text{ round}}{\text{The network is undecided in } (i-1)^{\text{th}} \text{ round}}\Big) = (1-\text{fail}(\phi_A^i)) \times \text{fail}(\phi_B^i) \tag{3.2}$$

The network fails to decide in the i^{th} round if the length of the chains remain equal, i.e., either both fractions mine a block (which is highly unlikely), or both fractions fail to mine a block.

$$P\Big(\frac{\text{The network fails to decide in } i^{\text{th}} \text{ round}}{\text{The network is undecided in } (i-1)^{\text{th}} \text{ round}}\Big) = (1-\text{fail}(\phi_A^i))\times(1-\text{fail}(\phi_B^i))+\text{fail}(\phi_A^i)\times\text{fail}(\phi_B^i) \tag{3.3}$$

$$\begin{aligned}
&P(\text{The network is undecided in } i^{\text{th}} \text{ round})\\
&= [(1-\text{fail}(\phi_A^i))(1-\text{fail}(\phi_B^i)) + \text{fail}(\phi_A^i)\text{fail}(\phi_B^i)]\\
&\qquad \times P(\text{The network is undecided in } (i-1)^{\text{th}} \text{ round})\\
&= \prod_{j=0}^{i}(1-\text{fail}(\phi_A^j))(1-\text{fail}(\phi_B^j)) + \text{fail}(\phi_A^j)\text{fail}(\phi_B^j)\\
&= \sum_{i=1}^{\infty} P(\text{The network is undecided in } (i-1)^{\text{th}} \text{ round} \cap \text{A gets accepted in } i^{\text{th}} \text{ round})\\
&= \sum_{i=1}^{\infty}\Bigg[P(\text{The network is undecided in } (i-1)^{\text{th}} \text{ round})\\
&\qquad\quad \times P\Big(\frac{\text{A gets accepted in } i^{\text{th}} \text{ round}}{\text{The network is undecided in } (i-1)^{\text{th}} \text{ round}}\Big)\Bigg]\\
&= \sum_{i=1}^{\infty}\Bigg[\prod_{j=0}^{i-1}[(1-\text{fail}(\phi_A^j))(1-\text{fail}(\phi_B^j)) + \text{fail}(\phi_A^j)\text{fail}(\phi_B^j)] \times (1-\text{fail}(\phi_A^i))\text{fail}(\phi_B^i)\Bigg]
\end{aligned}$$

$$\alpha_f = \frac{P(\text{A eventually gets accepted})}{P(\text{B eventually gets accepted})}$$
$$\alpha_f = \frac{P(\text{A eventually gets accepted})}{1 - P(\text{A eventually gets accepted})}$$
$$\therefore \alpha_f = \frac{\psi}{1-\psi}$$

□

Calculating α_f

If we assume the propagation of the block in the network to be linear[6], then by the end of δ_B, $\phi_A = \frac{\delta_B}{\delta_A+\delta_B}$ and $\phi_B = \frac{\delta_A}{\delta_A+\delta_B}$. Accordingly, the plot of ϕ_A and ϕ_B will be as shown in Figure 3.2. We then calculate α_f according to Theorem 3.2 and plot for varying network delays as well as varying throughputs.

Results on α_f

In Figures 3.5 and 3.5, we plot the variation of α_f with the increase in δ_B for different values of δ_A. We find that α_f grows exponentially with an increase in δ_B or a decrease in δ_A, which implies that even small differences in network delays can make the system unfair.

In Figure 3.5, we plot the variation of α_f as we increase throughput for a fixed δ_A and δ_B. We find that α_f grows with an increase in the block creation rate λ, which implies that as we scale the system, it becomes more unfair (from Remark 3.1.3). However, the α_f increases slowly after a certain λ. We must note that the probability of simultaneously mining a block still keeps increasing exponentially, which is not factored in α_f (from Remark 3.1.3). Hence, the overall ratio of blocks that A is able to include in the blockchain as compared to B still keeps increasing.

3.2.3 A Few Tips and Tricks

Improving the delays associated with broadcasting transactions and blocks for a node may not be as simple as improving the quality of a node's internet connection. A significant factor of the delay also depends upon the structure of the peer-to-peer network, the number of neighbors, and their delays. Decker and Wattenhofer (Decker and Wattenhofer,

[6]Although, we expect it to be exponential in practice due to the GOSSIP algorithm, a linear assumption is rather optimistic and our results would be more pronounced in the exponential case.

2013) suggest pipelining of information propagation to reduce the latency. However, we are more concerned with the changes a single node can implement to reduce its delay. Stathakopoulou et al. (Stathakopoulou, 2015) suggest implementing a Content Distribution Network that connects to nodes based on their geographic proximity. There have also been a few high-speed alternative "relay" networks developed to broadcast information quickly among a subset of nodes part of the network ((Corallo), (Basu et al.) and (Klarman et al., 2018)). However, there are many concerns raised about the centralized nature of these networks. A node that is a part of such a network could undoubtedly gain an advantage over others. However, doing so may lead to further centralization, defeating the purpose of having a distributed ledger.

3.3 Fairness and Strategic Deviations

A lack of fairness causes some agents to gain more than their fair share of rewards, whereas some agents gain less than their fair share. Since all agents have similar underlying costs related to mining, it would make mining less profitable or even loss-making for some agents. We believe this might open up pandora's box of strategic deviations that might be unfair to the honest players and detrimental to the health of the blockchain. Until now, we had considered that all players were honest and would not deviate from the protocol. However, the players may choose to deviate from the protocol if the deviation cannot be detected or penalized. When mining is not fair to some agents, they may be incentivized not to accept the consensus and depart from the honest strategy. This disagreement could be reflected by forks that cause the agents in the system to split their votes. These deviations might harm a blockchain's health and make it less secure against adversaries. We now briefly discuss a few possibilities if they act rationally to maximize their expected reward.

A slow node that does not have enough high-value transactions in its mempool might have an incentive to either fork the block mined by a frontrunner (undercutting) or given a fork pick the fork that offers an opportunity to collect a higher transaction fee (petty mining). If we assume that all agents are rational and hence petty miners since this strategy strictly dominates honest mining. Undercutting would allow a slower node to overcome both frontrunning and lack of publishing fairness. A slower node could fork a block mined by a faster node containing many high-value transactions due to frontrunning and include those transactions in its own block while leaving some transactions for others to include. Secondly, even if a node receives the block mined by a slower node later, it would drop the previous block and mine on top of this since it offers a higher reward. [7] In this case, it might not be in the best interest of the frontrunner to always pick the transactions offering higher rewards.

3.3.1 Modelling Bitcoin Mining As A Game

We divide the network into two portions: *slow* and *fast*. The fast nodes can receive messages broadcasted by any node in the previous round, but the slow nodes have higher communication delays with certain nodes. Each node can choose from the following strategies:

1. `petty`: The petty mining strategy described in (Carlsten et al., 2016c), it is same as the default strategy in case of no forks. It weakly dominates the default compliant strategy prescribed by Bitcoin but it is not harmful to the security of the blockchain on its own.

2. `minor_undercutting`: A node will undercut if the longest chain's reward is below a certain threshold. However, it would leave out a small constant reward as an incentive for the subsequent agents that pick the block.

3. `major_undercutting`(κ): A node will undercut if the longest chain's reward is below a certain threshold. However, it would leave out a significant portion of the reward (κ) as an incentive for the subsequent agents that pick the block.

The resulting game could be analyzed as a two-player bi-matrix game.

3.3.2 Simulation-based Analysis

Simulator Details

In order to study undercutting based strategic deviations in blockchains at high throughputs, we developed a lightweight simulator (described in Algorithm 2).

[7] In fact, undercutting might even occur in case of an unintentional fork between a faster node and a slower node (since the block mined by a faster node will contain more high-value transactions). The probability of such forks increases along with p_f and λ

Algorithm 2: Simulator

```
Input: Distance matrix $D_{n\times n}$, Number of rounds $R$, Strategy $s_i \in S \forall i \in [n]$
Result: Expected reward of each player $P_i \quad \forall i \in [n]$
/**                                                                    */
[h]Set of mined blocks
Initialize $B = $ genesis_block
Initialize longest_chain $= 1$
for $r = 1$ to $R$ do
    for $i = 1$ to $n$ do
        if $P_i$ wins a lottery in round $R$ then
            $B' = \emptyset$
            max_reward $= 0$
            for $b \in B$ do
                if $b$ satisfies strategy $s_i$ and $r - b.$round $\geq D[b.$miner$][i]$ then
                    $B' = B' \cup b$
                    max_reward $= \max(r - b.$round $+ b.$leftover, max_reward$)$
            for $b \in B'$ do
                if $r - b.$round $+ b.$leftover $\geq$ max_reward then
                    $\hat{b}.$parent $= b$
                    $\hat{b}.$miner $= i$
                    $\hat{b}.$height $= b.$height $+ 1$
                    $\hat{b}.$reward $= \min(r - b.$round $+ b.$leftover, max_block_size$)$
                    if $s_i \in$ major_undercut then
                        $\hat{b}.$leftover $= \kappa$
                        $\hat{b}.$reward $= \hat{b}.$reward $- \kappa$

                    else if $s_i \in$ minor_undercut then
                        /**                                                    */
                        [h]A small quantity $d$
                        $\hat{b}.$leftover $= d$
                        $\hat{b}.$reward $= \hat{b}.$reward $- d$
                    else
                        $\hat{b}.$leftover $= 0$
                    $B = B \cup \hat{b}$
                    longest_chain $= \max(\hat{b}.$height, longest_chain$)$
Return: Expected reward of each player averaged over all chains with length
        = longest_chain
```

	S_1	S_2	S_3	S_4
S_1	(75.22, 19.13)	(71.72, 23.12)	(74.29, 21.54)	(73.75, 22.17)
S_2	(75.22, 19.86)	(76.05, 19.56)	(72.17, 24.24)	(74.99, 21.74)
S_3	(63.30, 33.17)	(63.35, 33.84)	(66.90, 30.68)	(74.02, 23.6)
S_4	(63.41, 33.06)	(64.04, 33.29)	(56.66, 40.78)	(67.54, 30.26)

Table 3.1: Payoff Matrix averaged over 100 simulations
The utilities shown here are the percentage of total reward grabbed by the fast and slow sets collectively. (They do not add up to 100% since some blocks may be underutilized)
The strategies which would be removed by iterated removal of dominant strategies (with tolerance of ± 1) are shown in red.

We tested the following strategies: (a) `major_undercutting`(1.5) (S_1), (b) `major_undercutting`(1) (S_2), (c) `minor_undercutting` (S_3), and (d) `petty` (S_4). We assigned different strategies to slow and fast nodes to produce the payoff matrix in Table 3.1[8].

Equilibrium Analysis

By applying *Iterated Removal of Dominated Strategies* on the Payoff Matrix, we discard strategies S_3 and S_4 from the solution set of fast agents and discard S_4 from the solution set of slow agents. We then find the following two mixed strategy nash equilibria among the remaining set of strategies:

1. Fast agents choose S_1 with probability of 0.74 and S_2 rest of the time, while slow agents choose S_2 with probability of 0.32 and S_3 rest of the time.

2. Fast agents choose S_1 with probability of 0.06 and S_2 rest of the time, while slow agents always pick S_1.

We make the following additional remarks based on the payoff matrix:

1. The blockchain system would have been secure against any strategic deviations if (`petty`, `petty`) had been an equilibrium strategy. However, we observe that it is dominated by nearly all undercutting strategies for slower nodes. Hence, it would

[8]Due to computational constraints, we produced results for $\lambda = 300$ times that of Bitcoin which would yield a throughput 60% that of the Visa Network but we expect the results to be even more pronounced as we scale the blockchain further.

always be more profitable for the slower nodes to undercut. A lack of publishing fairness could explain this.

2. If the slow nodes choose `minor_undercutting`, it would be more profitable for the faster nodes to choose `major_undercutting`.

3. If all the nodes choose `major_undercutting` the total revenue gathered by the network reduces to roughly 94.3% from 97.8% indicating that fewer transactions are being added to the longest chain. This means that the throughput is being underutilized.

4. The equilibria strategies are even worse for the slower nodes in terms of fairness since they grab an even smaller share of reward as compared to the strategy where all players act honestly.

Thus, if agents act rationally not only would security of the blockchain be adversely affected, the lack of fairness among the rewards received by the slower miners would be exacerbated.

3.4 Frontrunning in OHIE

In this section, we describe a strategic deviation for the OHIE Protocol based on frontrunning. OHIE is a permissionless blockchain protocol that aims to achieve high throughput while tolerating up to 50% of the computational power being controlled by the adversary.

OHIE composes k (e.g., $k = 1000$) parallel instances or "chains" of Nakamoto consensus. Each chain has a distinct genesis block, and the chains have ids from 0 to $k - 1$ (which can come from the lexicographic order of all the genesis blocks). For each chain, Bitcoin's longest-chain-rule is followed. The miners in OHIE extend the k chains concurrently. They are forced to split their computational power across all chains evenly.

The total block ordering in OHIE is generated according to the increasing order of ranks and breaking ties by picking one with a lower chain id earlier. The miner of a block picks the rank of the block that follows it in a chain, i.e., the *next_rank*. Notice that the chains may not be equal in length and might have different *next_rank*s at their last positions. This implies that if a block is mined in a chain having lower *next_rank* than another chain, the block might end up earlier in the Total Block Ordering than a block that has already been mined.

According to the specifications of the protocol, a miner should pick the highest possible *next_rank* in order to ensure that the chain the block becomes a part of, catches up to the longest chain. A miner can also choose which blocks to mine on top of. If a rational agent wishes to insert his block earlier in the total ordering, he can choose a block that has picked a lower *next_rank* over one with a higher *next_rank*. We describe a rational deviation based on this fact.

This deviation is similar to undercutting in Bitcoin, described by Carlsten et al. (Carlsten et al., 2016c) We assume that at least some agents are rational and follow a *petty compliant* strategy. The agents still choose to mine on top of the longest chains, but in case of a tie or fork, they pick the block offering lower *next_rank*. Doing so provides an opportunity for *strategic frontrunning* by possibly achieving a lower rank (and hence include the high-value transactions of already published blocks of higher rank).

3.4.1 Expected Reward of Including Transactions

In our case, the frontrunning is not guaranteed to be successful. The block could end up on the chain from which the transaction was included. In which case, it will end up losing the transaction fees since it will not precede the original block from which the transaction was included. The block is equally likely to become a part of the k chains. Therefore we define the expected reward of including a transaction as follows:

$$\mathbb{E}[R] = P(\text{Successful frontrunning}) \times R$$

The probability of the succeeding to frontrun a block b will be:

$$P(\text{Frontrun } b) = \sum_{i=0}^{k-1} P(\text{Block mining on the } i^{\text{th}} \text{ chain} \cap \text{Block preceding block } b \text{ in TBO}) \tag{3.4}$$

We can expect a rational agent to include the transactions from the mempool as well as transactions from other blocks that offer the highest expected reward.

Similar to (Carlsten et al., 2016c) we find that for a *petty compliant* miner *frontrunning* is a better strategy since it guarantees as much reward as the honest node's strategy. As we try to scale the system to higher throuputs by increasing the number of chains, the number of possible blocks to frontrun will also increase (due to a higher probability of some chains being longer in length). Thus, the probability of *strategic frontrunning* also increases as we try to scale the system.

3.4.2 Undercutting Agents

Now consider if a more aggressive agent does not mind forking a chain. In the following example (modified version from the original paper (Yu et al., 2020)):

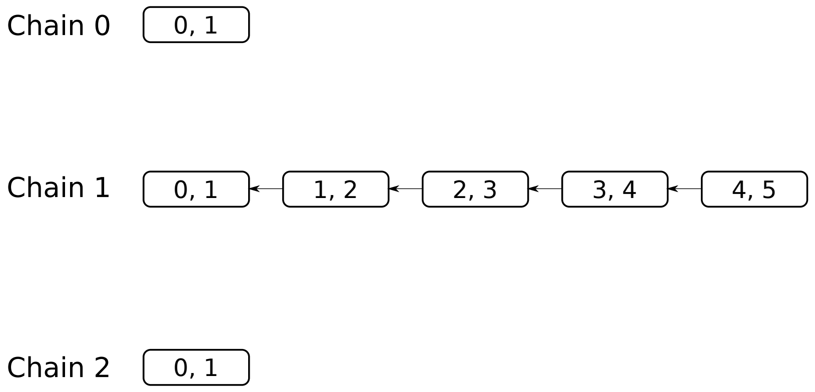

Figure 3.6: Example initial state of an OHIE execution with $k = 3$. Each block is marked with a tuple $(rank, next\_rank)$.

In the example initial state, the chains are of unequal length. (Such an event is possible since the blocks extend chains at random, some chains can receive more blocks than others)

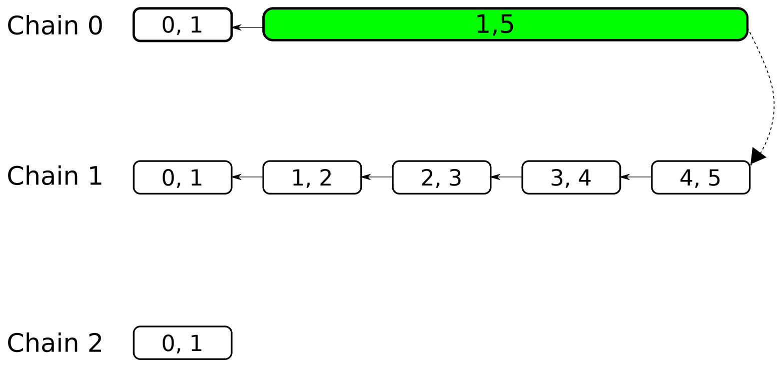

Figure 3.7: The state after the honest node extends Chain 0. A dotted arrow denotes the trailing pointer.

Let us say that an honest node mines the next block on Chain 0. Since the node follows the default strategy of setting the $next\_rank$ to be the maximum $next\_rank$ among all chains, it sets the $next\_rank$ to be 5. The mechanism by which the $next\_rank$ is specified is by including a trailing pointer to the last block on Chain 1. Hence, the *next_rank* of this block is implicitly set to the *next_rank* of the block that the trailing pointer points to.

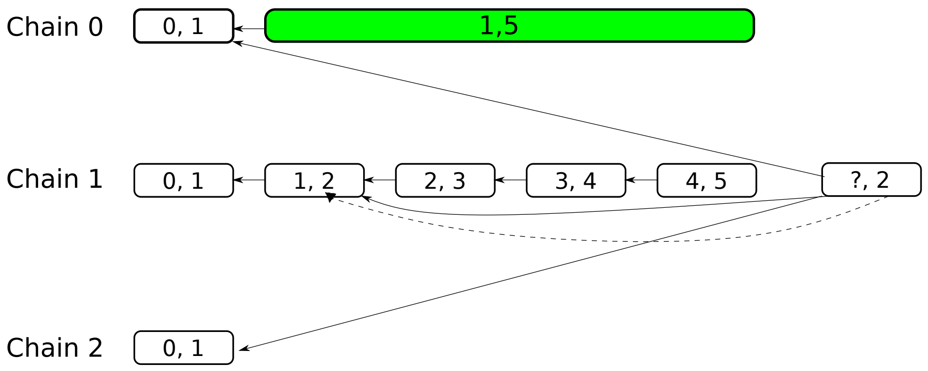

Figure 3.8: The block formulated by an aggressive miner for undercutting. The dotted arrow denotes the trailing pointer. The three solid arrows point to the Merkle tree of pointers to preceding blocks.

Consider a agent that tries to undercut aggressively. It picks the blocks it wants to drop (in this case $(1, 5)$ on Chain 0 and $(2, 3)$, $(3, 4)$, and $(4, 5)$ on Chain 1) and then picks the trailing pointer to $(1, 2)$. However, it could have picked $(0, 1)$ on any chain as the trailing pointer, in which case it would have been assigned a *next_rank* of $rank + 1$. This deviation would have easily been detected since its trailing pointer would have lesser *next_rank* than the block it precedes, indicating the deviation. In our case, the agent tries to deviate in a manner that is not distinguishable from a fork. In order to do so, it selects a *next_rank* that is greatest among the blocks in its Merkle tree.

If the undercutter is able to mine the block successfully, it may end up on one of the three chains depending upon the last $\log_2 k$ bits of the hash. We consider these three cases separately:

- If the block ends up on Chain 0, the new block forks the chain. Since the two forks are equal it is upto the nodes in the network to choose the fork they wish to extend. Choosing the undercutter's block in this case would be a better option for the *petty compliant* agents since it offers a lower *next_rank*. If the majority of the agents are *petty compliant*, then the undercutter is successful.

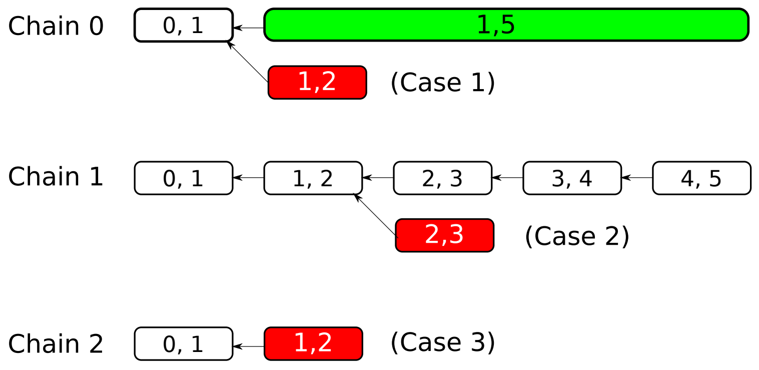

Figure 3.9: The three possible cases that could arise if the undercutter's mining is successful

- If the block ends up on Chain 1, the new block forks the chain. Since the undercutter's fork is shorter than the original chain, it would be orphaned by all agents. In this case, the undercutter is unsuccessful.

- If the block ends up on Chain 2, the new block extends the original chain. All agents will prefer to mine on top of undercutter's block. In this case, the undercutter is successful.

Hence, in 2 out of 3 cases, i.e., with a probability of 2/3, the undercutter is successful. Therefore, the agent may undercut if the reward obtained by picking transactions from the blocks it drops is 3/2 times more than the reward by mining honestly. That is, if the $\mathbb{E}[\text{Reward obtained by undercutting}] >$ Reward obtained by mining honestly then undercutting may be a better strategy.

3.5 Conclusion

In this chapter, we introduced a notion of *network fairness*, investigated the factors that influence network fairness, and studied its impact on the agent' revenue. We assumed a model in which players have asymmetric network connections, i.e., some players may have a faster connection while others may have a slower connection and described two mechanisms via which this asymmetry could result in a loss of fairness in the system. We considered two events associated with the mining process, frontrunning and block publishing, and measured fairness for these events. We showed that fairness can be quantified via p_f and α_f. p_f is a measure of *frontrunning* due to network delays in broadcasting a transaction. α_f or publishing fairness is the ratio of blocks of one node that get accepted due to network delays over blocks of another node. We found that both of them deteriorate as we increase the throughput of existing protocols. Hence, even though it might look like the agents walk together (or the system is fair) while the throughput is low, at higher speeds, some agents may run faster (or gain more than their fair share of rewards).

We also discussed that not only does a lack of fairness impacts the revenue of some agents, it might also create an incentive for them to deviate from the honest mining strategy, which might impact the security of the blockchain system and further exacerbate lack of fairness in rewards.

Thus, we conclude that even though blockchain is an ambitious technology, its potential is hindered by the underlying network infrastructure.

Chapter 4

Tiramisu: Layering Consensus Protocols for Scalable and Secure Blockchains

Researchers have developed several multi-layered blockchain protocols that combine both permissioned and permissionless blockchain protocols to achieve high performance along with decentralization known as layered *blockchain protocols. The key idea with existing layered blockchain protocols in literature is to divide blockchain operations into two layers and use different types of blockchain protocols to manage each layer. However, many such works come with the assumptions of the honest majority, which may not accurately reflect the real world where the participants may be self-interested or rational. These assumptions may render the protocols susceptible to security threats in the real world, as highlighted by the literature that explores game-theoretic attacks on these protocols. In this chapter, we generalize the "layered" approach taken by existing protocols in the literature, present a framework to analyze the system in the BAR Model, and provide a generalized game-theoretic analysis of such protocols. Our analysis identifies the critical system parameters required for a distributed ledger's secure operation in a more realistic setting.*

Researchers have proposed several multi-layered protocols to improve blockchain technology's practical performance for better applicability (Abraham et al., 2018; Timo Hanke and Williams, 2018; Decker et al., 2016; Pass and Shi, 2017a). These protocols combine both permissioned and permissionless protocols to achieve scalability while maintaining decentralization. We take inspiration from these works, abstract out this layered approach, and formalize these layers for different system functions. These essentials lead us to a general framework we term as "Tiramisu".

We highlight that these works do not provide a game-theoretic analysis to ensure incentive compatibility, i.e., all the above protocols assume miners are either honest or byzantine. However, the miners could also be *strategic players*. That is, they may deviate from the prescribed protocol to gain additional *rewards* because it may not be a strategic miner's *best response* to follow the protocol honestly. Selfish mining, petty mining, and undercutting are some of the strategies that may lead to greater rewards for the miners (Carlsten et al., 2016b; Sapirshtein et al., 2016a). In the game-theory literature, a system is said to be *incentive compatible* if it rewards each player more for playing truthfully compared to all other possible strategies. We remark that designing incentive-compatible blockchain protocols robust to strategic deviations is a significant challenge. It is also a new area of research with limited prior work (Siddiqui et al., 2020b).

Overall, we observe that building a scalable, consistent, and fully decentralized practical blockchain protocol remains elusive. Cryptocurrencies like EOS (eos, 2020), DFINITY (Timo Hanke and Williams, 2018), Solida (Abraham et al., 2018), etc. have a committee-based blockchain protocol. Among these, PeerCensus (Decker et al., 2016) and Hybrid Consensus (Pass and Shi, 2017a) inherently use a layered approach that demonstrates its potential benefits by achieving higher throughput and faster block confirmations. We highlight none of these works provide a game-theoretic analysis to ensure incentive compatibility and thus, security in the presence of *rational* miners – the ones maximing their rewards. Hence, the scalability guarantees provided by these protocols might not hold if the protocol is not incentive compatible.

Tiramisu Overview

We believe that with a layered approach, one can create protocols that offer the best of both worlds: throughput and decentralization. We dub this approach as "Tiramisu" after the layered dessert. At a high level, we illustrate Tiramisu with Figure 4.1.

More concretely, Tiramisu consists of two layers: Access Control Layer (ACL) and Consensus Later (CSL). In ACL, we run a permissionless consensus protocol to obtain authorized nodes. Then, with CSL, these nodes form a committee to run a BFT consensus protocol on the system's state. As our framework is general, one can use any permissioned blockchain protocol in CSL, as long as it satisfies certain conditions (Section 4.3). As standard, our security analysis assumes that ACL is secure. This assumption is based on

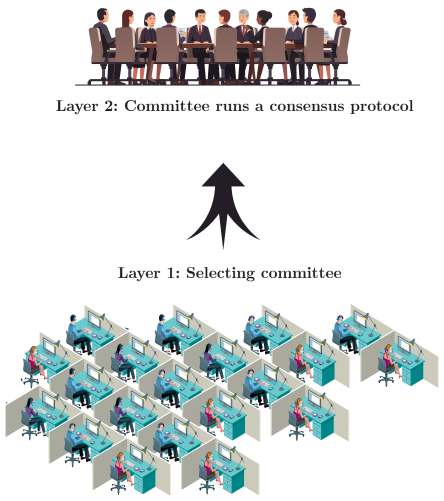
Layer 2: Committee runs a consensus protocol
Layer 1: Selecting committee

Figure 4.1: A high-level view of Tiramisu

inherent security guarantees of the underlying consensus protocol that must be carefully employed in the design. Hence, we do not discuss security attacks on it and analyze the security of Tiramisu through CSL (Section 4.3.2). We also identify three conditions in our game-theoretic analysis (Section 4.3.1) that ensure incentive-compatible implementation of any Tiramisu protocol.

4.1 Preliminaries

Network Model

Here, we formally describe our network model. We consider a partially synchronous peer-to-peer network consisting of participants who control identities in the network. These identities are denoted by their public-private (pk, sk) key pair and hence, are only pseudonyms since they do not leak any real-world information about the participant[1]. Participants connect by a broadcast network over which they can send messages to everyone.

Agent Model

We follow the BAR Model introduced by (Aiyer et al., 2005) that consists of Byzantine, Altruistic (or Honest) and Rational Parties.

[1]For brevity, we use identities in place of pseudonymous identities.

Notation	Meaning
$\alpha_{participant}$	Fraction of network resources owned by a participant
α_A	Fraction of network resources owned by adversary A
Π_{ACL}	Underlying Access Control Layer Protocol
Π_{CSL}	Underlying Consensus Layer Protocol
$\Pi_{\mathcal{TM}}$	Tiramisu Protocol: $\{\Pi_{ACL}, \Pi_{CSL}\}$
n_{CSL}	Total number of committee nodes in CSL.
n_f	Minimum nodes needed to compromise Π_{CSL}.

Table 4.1: Protocol specific notations

Term	Definition
$\mathcal{N}$	Set of nodes, $\{1, 2, \ldots, n_{CSL}\}$
τ_i	Node i's type, i.e.,$\tau_i \in \{\mathcal{H}, \mathcal{R}, \mathcal{B}\}$
s_i^1	Node i signs the block without validating
s_i^2	Node i signs the block only if its valid
s_i^3	Node i signs and proposes only invalid blocks
S_i	Set of pure strategies of nodes i, $\{s_i^1, s_i^2, s_i^3\}$
TR_i	Total rewards of node i for a transaction block.
u_i	Utility of node i; $u_i : \tau_i \times TR_i \times S \rightarrow \mathbb{R}$

Table 4.2: Game constituents

Definition 4.1 (**$\mathcal{H}$ Honest Party**). *A party is said to be honest if and only if it strictly follows the protocol.*

Definition 4.2 (**$\mathcal{R}$ Rational Party**). *A party is said to be rational or self-interested if it may strategically deviate from the protocol if the deviation is expected to yield a higher reward.*

Definition 4.3 (**$\mathcal{B}$ Byzantine Party**). *A party is said to be an adversarial or byzantine if it may or may not follow the protocol. The adversary's goal is to disrupt the operation of the protocol, and it does* ***not*** *try to maximize its reward.*

With this background, we now present Tiramisu in the next section.

Term	Definition
c_{mine}	The cost of mining a block in ACL.
c_{val}	The cost to validate one byte of data.
n_{TX}	Number of transaction blocks for a given committee session.
κ_i	Cost incurred by a rational node i if an invalid transaction block is accepted.
$\kappa_{\mathcal{R}}$	κ_i for the rational node with minimum stake invested.
ϕ	Number of bytes of shared state to be validated for a single block.
TR	Minimum TR_i for the transaction block.
$P_{invalid}$	Belief probability of invalid block being accepted, $P_{invalid} : \mathcal{N}^{n_{CSL}} \times S_i \rightarrow [0, 1]$.

Table 4.3: Relevant Game Notations

Name	State
PowChain	Proof of Work Blockchain
T	List of transaction validators
ComChain	Committee Blockchain
TxChain	Transaction Blockchain
O	Operation Log
B	Account Balances

Table 4.4: Shared state constituents

Name	Operation
`powBlock`(b)	Adds PoW block b to PowChain
`txBlock`(t)	Adds transaction block t to T
`comBlock`(c)	Adds committee block c to ComChain

Table 4.5: Shared state operations

4.2 Tiramisu: A Layered Approach

In Tiramisu, we split the operation of the blockchain into two independent layers. The first and the second layers are called Access Control Layer (ACL) and Consensus Layer (CSL) respectively. ACL manages nodes in the network via a permissionless consensus protocol. Whereas, CSL employs a permissioned consensus protocol among the authorized nodes from the first layer. These nodes will run a Byzantine agreement protocol to verify transactions and reach a consensus on the state of the system.

Let $\Pi_{\mathcal{TM}} = \{\Pi_{ACL}, \Pi_{CSL}\}$ be a Tiramisu protocol where Π_{ACL} and Π_{CSL} refer to the underlying consensus protocols running in the Access Control Layer and the Consensus Layer respectively.

4.2.1 Access Control Layer (ACL)

ACL is responsible for providing sybil-resistant node identities to the Consensus Layer. The identity of these *nodes* are simply a derivative of the public key of a public-secret-key pair $\langle pk, sk \rangle$, owned by a participant. Each node is identified by its public address.

Participants maintain a separate blockchain in this layer just to obtain Sybil-resistant identities. Participants in this layer run any permissionless blockchain consensus protocol by choice of protocol design, denoted by Π_{ACL}. Note that Π_{ACL} must satisfy some pre-defined conditions. Towards stating these pre-defined conditions, we first describe a *democratized resource*. A democratized resource is essential to run the protocol Π_{ACL}. It can be captured as a real-world resource and must be directly responsible for: first, the basis of sybil-resistance in Π_{ACL}; second, for obtaining control over specific network resources. These network resources must be directly proportional to the fraction of control they gain over the protocol Π_{ACL}. Typically Π_{ACL} can be a proof-of-X base protocol like Proof-of-Work. We assume that ACL is securely based on the inherent security of Π_{ACL}. Formally, Π_{ACL} must satisfy the following properties:

- There must exist a protocol parameter that is directly able to express the democratized resource present in the network.

- The probability of a node adding a new block to its blockchain must be linearly dependent on the democratized resource.

For instance, Π_{ACL} could be a Proof-of-Work protocol where the democratized resource is *computational power* and the protocol parameter could be the *hash-rate* of the network.

Another Π_{ACL} could be a Proof-of-Stake protocol where both the resource and this parameter are the stakes invested in the system.

Operation. Π_{ACL} is run to determine the identities of the nodes participating in the Consensus Layer. Each block in the blockchain of this layer must contain a single public address representing the identity of the node that wishes to join the CSL. Once a block reaches finality (Section 4.1), ACL uses a pre-defined interface to interact with CSL and propose the identity present in this block to join the committee in CSL. Observe that any participant will only invest their network resources for the identities of the nodes they wish to be promoted in the Consensus Layer. The key insight behind the sybil-resistant nodes is that the democratized resources are hard to obtain and may not be scaled at will. Observe that the probability of a node joining the CSL is directly proportional to the amount of democratized resources owned by the participant, denoted by $\alpha_{participant}$. This is a key observation that will be used in our security analysis (Section 4.3).

In Section 4.3, we show that for the system to function correctly, the fraction of resources controlled by the adversary must be less than $\frac{n_f}{n_{CSL}}$. Here, n_{CSL} and n_f are the total number of nodes in the Consensus Layer and the minimum number of nodes needed to compromise Π_{CSL}, respectively. Then we show that α_A is a reasonable bound for practical systems. We now move on to the second layer, called the Consensus Layer (CSL), in which the committee of nodes from this layer handles the transactions.

4.2.2 Consensus Layer (CSL)

This layer is responsible for handling transactions and reaching a consensus on the system's state. Participants control the nodes in this layer. This layer maintains a shared state of the system by running any Byzantine agreement protocol or BFT protocol, denoted by Π_{CSL}. One can use any BFT protocol suited for the permissioned blockchain setting as long as it satisfies the network model. Pre-determined operations can only modify the shared state. This shared-state can be anything related to the protocol's purpose, like running a cryptocurrency, notarization platform, smart-contract platform, etc. For a cryptocurrency, this state may contain account balances, a transaction blockchain, etc. Note that, however, for the sake of simplicity in this paper, we use the term *transaction blockchain* to denote the shared state, which represents the purpose of the Tiramisu protocol, $\Pi_{\mathcal{TM}}$. Similarly, we use the term *transaction* throughout this paper to denote a state change in the transaction blockchain.

We now claim that the Tiramisu protocol satisfies the properties of distributed ledgers described in Section **??**. BFT protocols structure their execution into a sequence of views, each with a designated leader process. These protocols guarantee Safety and liveness by ensuring that all correct nodes eventually overlap in a single view, with the right leader, for enough time to reach consensus.

Claim 4.1. *A Tiramisu protocol session achieves Safety and liveliness if* Π_{CSL} *does so.*

At any time, this layer is operated by n_{CSL} nodes, collectively called the *?* of this layer. The committee size can be fixed or variable, depending on network parameters. Fixed size can be achieved through various techniques. For example, allowing only the latest n_{CSL} nodes from the ACL blockchain to join the system. Note that the shared-state must store a copy of the ACL blockchain, or an equivalent, to ensure verification of the nodes. The nodes in the committee are decided only after a BFT agreement on the identity of these nodes. This agreement ensures that any behaviour of the ACL blockchain, like a blockchain fork, is irrelevant in this layer. Since we use Byzantine agreement protocols, CSL inherently satisfies the properties of Safety and liveness. This justifies Claim 4.1.

Now we discuss our analysis of Tiramisu, in which we also identify three conditions on protocol parameters that ensure incentive compatible implementation of a Tiramisu protocol.

4.3 Tiramisu: Analysis

It is important to note that each node is independently selected to be on the committee. The probability of this selection is proportional to the number of democratic resources, as required for participating in ACL which is owned by the node in the network.

The BAR model (described in Section 4.1) allows us to analyze the game-theoretic and security aspects of Tiramisu as presented next.

4.3.1 Game-theoretic Analysis

As stated in our player model, the reward structure of any blockchain consensus protocol induces a game among the participants. We highlight that we account for the computational costs of validating the system's state. This consideration forms the basis of our game model, making it more suited for a realistic setting. To analyze this induced game, we define the following notations: Consider a set of nodes $\mathcal{N} = \{1, \ldots, n_{CSL}\}$. This set includes all the information about a node, including its strategy and type. Let each participant i's strategy be s_i and its type be τ_i, where $\tau_i \in \{\mathcal{H}, \mathcal{R}, \mathcal{B}\}$. Note that the strategy will depend on the participant's type. We have $\vec{s} = (s_1, \ldots, s_m)$ as the strategy vector and $\vec{s}_{-i}$ as the vector without node i. Let TR_i denote participant i's reward for one transaction block. Note that this reward is left for the design of the protocol. With this, $u_i(\tau_i, TR_i, \vec{s})$ represents a participant i's utility from its participation. We assume that rational participants do not collude with each other based on the premise that the distributed environment will make it resistant to collusion.

In Tiramisu, we game-theoretically analyze the node's behavior at equilibrium using the following notion.

Definition 4.4 (Pure Strategy Nash Equilibrium (PSNE)). *A strategy vector* $\vec{s^*} = \{s_1^*, \ldots, s_m^*\}$ *is said to be a Pure Strategy Nash Equilibrium (PSNE) if for every node i, it maximizes its utility* $u_i(\vec{s}^*, \tau_i, r_i)$ *i.e.,* $\forall i \in \mathcal{M},\ \forall j$,

$$u_i(\vec{s^*}, \tau_i, r_i) \geq u_i(s_i, \mathcal{S}_{-i}^*, \tau_i, r_i);\ \forall s_i. \tag{4.1}$$

Intuitively, PSNE states that it is the best response for a node to follow $\vec{s^*}$, given that every other node is following it.

Equilibrium. Towards this, we assume that a rational player $\mathcal{R}$ incurs a cost $\kappa_{\mathcal{R}}$ if any malicious transaction block is committed in CSL, similar to (Amoussou-Guenou et al., 2020). This assumption is based on the premise that the entire ecosystem of the currency inflicts harm when an invalid block is accepted. This cost is directly proportional to the amount of stake that the rational node has invested in the system. Note that this stake can be in the form of electricity for mining in a proof-of-work blockchain, amount locked in the blockchain for participation in a proof-of-stake blockchain, rewards earned by the miner, and likewise. For simplicity, we take this proportionality constant to be 1, making the cost equal to the stake invested. We denote this cost of a rational node i by κ_i. Let $\kappa_{\mathcal{R}}$ be κ_i of the rational player with minimum invested stake. Observe that the protocol designer can set a lower-bound on $\kappa_{\mathcal{R}}$ using a variety of well-known techniques.

Although Byzantine nodes behave arbitrarily by definition, we consider a specific behavior of Byzantine nodes. Specifically, we assume that the objective of Byzantine nodes is to minimize the utility of the rational nodes and prevent the protocol from achieving its goal, regardless of the cost they incur. Note that this assumption ensures the security of a Tiramisu protocol in the worst possible case, and hence it ensures security for all cases, i.e. regardless of the behavior of Byzantine nodes. With these assumptions, we model a game between honest, rational and Byzantine nodes' nodes in the committee, defined as: $\Gamma = \langle N, (\tau_i), (S_i), (u_i) \rangle$.

Definition 4.5 (Nash Incentive Compatible). *We say that a Tiramisu protocol $\Pi_{\mathcal{TM}}$ is Nash Incentive Compatible (NIC) if $\vec{s^*} = (s_1^2, \cdots, s_{n_{\mathcal{R}}}^2)$ is a PSNE for all rational nodes and at this PSNE all rational nodes obtain non-negative utility.*

For the analysis, let $P_{invalid}(\mathcal{N}, s_i^*)$ be node i's belief that an invalid block is accepted after it follows s_i^*. Honest nodes will always follow the protocol, i.e. s_i^2 strategy. Byzantine nodes will always follow s_i^3 because this is the strategy that best aligns with our assumption on the objective of Byzantine nodes. Hence we only consider rational nodes' behavior and therefore their equilibrium strategies.

The expected utility of a rational node, when following s_i^1 strategy for a single transaction block in CSL, is total rewards obtained for that block, minus the average cost of mining a block in ACL for that committee session, and minus the expected κ_i. Similarly, to calculate the utility of a rational node for strategy s_i^2, we need to subtract the cost of validating a block as well. The expected utilities of a rational node i for one round are as follows:

$$u_i(\cdot, \cdot, s_i^1) = TR_i - \frac{c_{mine}}{n_{TX}} - P_{invalid}(\mathcal{N}, s_i^1) \cdot \kappa_i \tag{4.2}$$

$$u_i(\cdot, \cdot, s_i^2) = TR_i - \phi \cdot c_{val} - \frac{c_{mine}}{n_{TX}} - P_{invalid}(\mathcal{N}, s_i^2) \cdot \kappa_i \tag{4.3}$$

For n_{CSL} nodes, let n_H, n_R, and n_B be number of honest, rational and Byzantine nodes respectively. Trivially, $n_{CSL} = n_H + n_R + n_B$. We use PSNE for analysis, as we have a static game with asymmetric information.

Claim 4.2. *For every rational node i, the probability of an invalid block being accepted will be more if it signs a block without validating, as compared to when it validates and then signs, i.e.,*

$$\delta = P_{invalid}(\mathcal{N}, s_i^1) - P_{invalid}(\mathcal{N}, s_i^2) > 0.$$

Intuitively, Claim 4.2 follows from the fact that the chance of an invalid block being accepted is more if a rational node decides to sign the block as valid without verifying it. Now we present the formal proof for the claim.

Proof. Without loss of generality, for the proof we consider a rational player i. Let

$$P_{invalid}(\mathcal{N}, s_i^1) = k_1$$

and

$$P_{invalid}(\mathcal{N}, s_i^2) = k_2$$

s.t.

$$0 < k_1, k_2 < 1.$$

In the event when node i plays the strategy s_i^1 i.e. signs the block as valid without validating, its belief regarding an invalid block being accepted can only increase. This follows by observing that the size of the committee, i.e., n_{CSL} is finite. Thus, node i not validating a block will directly imply that the chance of an invalid block being accepted will be more, i.e., $k_1 > k_2$. Now,

$$P_{invalid}(\mathcal{N}, s_i^1) - P_{invalid}(\mathcal{N}, s_i^2) = k_1 - k_2 = \delta > 0$$

$\square$

We denote the minimum possible value of δ for a protocol session running with specific security guarantees as δ_{min}. Along with the node's belief and some protocol parameters, δ_{min} is dependent on the CDF F which is appropriately defined in security analysis under proof of Proposition 4.1.

NIC Conditions. For all rational nodes, there exists a PSNE under certain conditions. We identify three such conditions to ensure that a Tiramisu protocol session $\Pi_{\mathcal{TM}}$ is NIC (Definition 4.5). We call these set of conditions as *NIC-Conditions* which are based on the following requirements. First, to ensure that Π_{CSL} is secure. Second, to ensure that rational nodes attain non-negative utility when they deviate from following the honest strategy. And third, to ensure that each rational node must obtain positive utility. The respective NIC-Conditions are as follows:

1. **Faithful Fault Tolerance Condition** $n_B < n_f$
2. **Maximum Payload Condition** $\phi \leq \frac{\kappa_{\mathcal{R}} \cdot \delta_{min}}{c_{val}}$
3. **Minimum Reward Condition** $TR \geq \phi \cdot c_{val} + \frac{c_{mine}}{n_{TX}}$

Note that NIC-Conditions provide constrains on protocol parameters which are dependent on protocol design, implementation, and live-network data[2]. Hence, satisfying these conditions rely on the careful implementation by protocol designer.

Theorem 4.1. *In Tiramisu, Faithful Fault Tolerance Condition, Maximum Payload Condition and Minimum Reward Condition are sufficient to ensure that the protocol is Nash Incentive Compatible (NIC). Formally, if Faithful Fault Tolerance Condition, Maximum Payload Condition and Minimum Reward Condition are true, then $\Pi_{\mathcal{TM}}$ is NIC according to Definition 4.5.*

Proof. To ensure that $\Pi_{\mathcal{TM}}$ is NIC, first requirement is that Π_{CSL} must work correctly. Hence the condition $n_B < n_f$ should be true, justifying the requirement of Faithful Fault Tolerance Condition.

Second, $s_i^* = s_i^2$ must be a PSNE for every rational node i. To ensure this, $u_i(R, s_i^1) \leq u_i(R, s_i^2)$ should satisfy according to Equation 4.1. On solving this condition using equations 4.2 and 4.3 we get,

$$\kappa_i \cdot (P_{invalid}(\mathcal{N}, s_i^1) - P_{invalid}(\mathcal{N}, s_i^2)) \geq \phi \cdot c_{val}.$$

Substituting $(P_{invalid}(\mathcal{N}, s_i^1) - P_{invalid}(\mathcal{N}, s_i^2))$ as δ, where $\delta \in (0, 1]$ following from Claim 4.2, we get our result for the value of ϕ as:

$$\phi \cdot c_{val} \leq \kappa_i \cdot \delta.$$

Now, since $\kappa_{\mathcal{R}} \leq \kappa_i$ and $\delta_{min} \leq \delta$, we get the following condition as:

$$\phi \cdot c_{val} \leq \kappa_{\mathcal{R}} \cdot \delta_{min},$$

justifying the requirement of Maximum Payload Condition.

Finally in third, for the above mentioned PSNE, utilities gained by all rational nodes must be non-negative, i.e.

$$TR_i - \phi \cdot c_{val} - \frac{c_{mine}}{n_{TX}} - P_{invalid}(\mathcal{N}, s_i^2) \cdot \kappa_i \geq 0.$$

Notice that in this PSNE, since all rational nodes will validate the blocks, $P_{invalid}(\mathcal{N}, s_i^2) = 0$. Also, $TR_i \leq TR$. From this we get the condition as:

$$TR \geq \phi \cdot c_{val} + \frac{c_{mine}}{n_{TX}},$$

justifying the Minimum Reward Condition. □

[2]Here, live-network data at least includes the stored data of a protocol session along with its consolidated statistics.

However, in a special case where $n_H > n_{CSL} - n_f$, $S^* = (s_i^1)$ is the PSNE for rational nodes where only valid blocks are accepted and is as special case of a NIC $\Pi_{\mathcal{TM}}$.

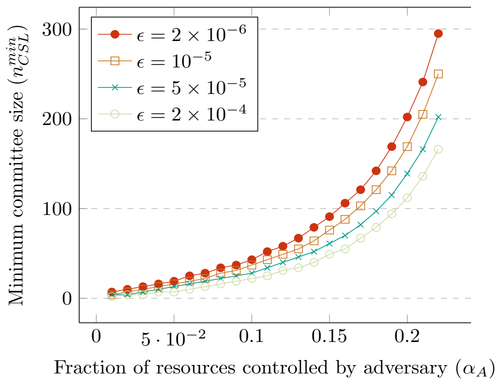

Figure 4.2: For a given α_A, n_{CSL}^{min} for different values of ϵ.

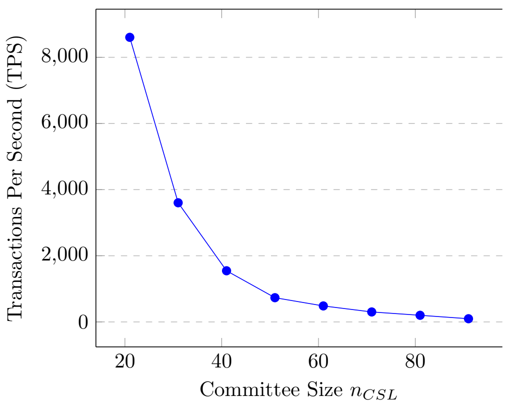

Figure 4.3: Throughput vs. Committee Size

4.3.2 Security Analysis

Setting. Our security analysis assumes that protocol Π_{ACL} is secure, i.e. ACL runs securely based on inherent security of Π_{ACL} and intelligent protocol design. We thus analyse the security of a $\Pi_{\mathcal{TM}}$ session by just ensuring the security of CSL. Notice that Tiramisu's security relies on securing Π_{CSL} and the inherent security guarantees of Π_{ACL}. We achieve this security just by quantifying the committee size in CSL based on an appropriately defined function in this analysis. Next, we consider this security analysis for a NIC Tiramisu protocol session. This is followed from Theorem 4.1 in our game-theoretic analysis. Notice that in this setting, since all rational players will follow the honest strategy, it is justified to consider all rational nodes as honest. Hence, we consider only honest and Byzantine nodes in our security analysis of a NIC Tiramisu protocol. Towards this, we first define our adversary model.

Adversary Model. We consider adversary A controlling α_A fraction of the democratized resources in the network. The remaining resources are controlled by a meta entity $\mathcal{H}$, denoting all other participants in the network. The adversary's goal is to control more than n_f nodes, i.e., the minimum number of nodes needed to compromise the protocol.

Note that the probability of a node joining the CSL is directly proportional to the amount of democratized resources owned by the respective participant in ACL. We denote the total participants in CSL to be n_{CSL} at any time. Similar to (Decker et al., 2016), we analyze Tiramisu in its *steady state*, i.e., the number of validators and computational resources are governed by their respective expected value. To capture this formally, we define the following *secure state*.

Definition 4.6 (Secure State). *A $\Pi_{\mathcal{TM}}$ session is said to be in a secure state ($\mathcal{SS}$) if the number of nodes in CSL controlled by an adversary is strictly less than minimum number of nodes needed to compromise Π_{CSL}. Formally, $n_A < n_f$.*

We highlight that Claim 4.1 relies on security and correct execution of $\Pi_{\mathcal{TM}}$. Since it relies on security guarantees of underlying protocols Π_{ACL} and Π_{CSL}, we say that Claim 4.1 is true if $\Pi_{\mathcal{TM}}$ session is in a secure state. Hence, we state that a $\Pi_{\mathcal{TM}}$ session achieves Safety and liveness only in a secure state given Π_{CSL} does so. Towards this, we introduce the secure state function $F_{\mathcal{SS}}$ which provides n_{CSL}^{min}, the minimum number of nodes required in CSL for a $\mathcal{SS}$ with $1-\epsilon$ probability.

$$F_{SS} : R \times R \rightarrow Z^{+},$$

and

$$n_{CSL}^{min} = F_{SS}(1-\epsilon, \alpha_A).$$

Here, $1-\epsilon$ denotes the required minimum probability for a $\mathcal{SS}$, and α_A denotes the fraction of network resources controlled by the adversary. Note that $F_{\mathcal{SS}}$ is a useful function to ensure Faithful Fault Tolerance Condition with desired probability guarantee.

Proposition 4.1. *A Tiramisu protocol is in a secure state for a given α_A with probability of at-least $1-\epsilon$ if the minimum number of nodes in CSL is*

$$n_{CSL}^{min} = F_{SS}(1-\epsilon, \alpha_A),$$

where

$$F_{SS} = argmin_{n_{CSL}}\{(1 - F(n_f; n_{CSL}, \alpha_A)) \leq \epsilon\}.$$

Here $F(n_f; n_{CSL}, \alpha_A)$ is an appropriately defined function in the proof and gives the probability of $n_A < n_f$.

Proof. The probability that a participant mines a block in ACL is directly proportional to its fraction of computational power in the system, i.e.,

$$\Pr(\text{Mine a block in ACL}) \propto \alpha_{participant}.$$

This follows by observing that the size of the committee, i.e., n_{CSL} is finite. Let X be the discrete random variable denoting the number of validators controlled by the adversary. Consider the following standard binomial distribution for deriving the probability of the number of validators controlled by the adversary A as,

$$F(k; n_{CSL}, \alpha_A) = \Pr(k; n_{CSL}, \alpha_A) = \Pr(X = k)$$

$$\text{s.t. } \Pr(X = k) = \binom{n_{CSL}}{k}(\alpha_A)^k(1-\alpha_A)^{(n_{CSL}-k)}$$

Now, let F be the corresponding CDF defined as,

$$F(x; n_{CSL}, \alpha_A) = \Pr(X \leq x).$$

$F(n_f; n_{CSL}, \alpha_A)$ gives the probability that the system is in a secure state, i.e. $n_A < n_f$, which depends on the value of n_{CSL} and α_A from the binary distribution.
With this we set

$$F_{SS}(1-\epsilon, \alpha_A) = argmin_{n_{CSL}}\{(1 - F(n_f; n_{CSL}, \alpha_A)) \leq \epsilon\}.$$

□

Selecting Safe Committee Size. ϵ denotes the probability of a system not in a secure state, i.e. $\epsilon = \Pr(\Pi_{\mathcal{TM}}$ not in $\mathcal{SS})$. Please observe that the probability of a node being added to CSL is directly proportional to $\alpha_{participant}$, i.e., the resources it holds in the system. Since each node in the CSL can either belong to the rational participants with probability $1 - \alpha_A$ or to a byzantine adversary with probability α_A, we can model this distribution as a binomial distribution. We know that, from Proposition 4.1, the protocol will not be in a secure state if the number of nodes controlled by the adversary in the committee is equal to or more than n_f, i.e. $\alpha_A \geq n_f$. Thus, ϵ becomes the binomial distribution of n_A successes in n_{CSL} independent experiments, with $\frac{\alpha_A}{n_{CSL}}$ as the probability of each success. Based on different values of α_A and probability guarantees of a secure state, we obtain the values of n_{CSL}^{min}[3]. We consider this analysis for probability guarantees for the system being in a secure state, ranging from 2×10^{-4} to 2×10^{-6}. Note that in Bitcoin, for 10% computational power with an adversary, the system is said to be secure with a probability of 2×10^{-4}, i.e., 1 in 5000. We consider the values of α_A, the resources controlled

[3]For additional details, we refer the reader to Proposition 4.1

by the adversary, in the range 1% to 18%. This range follows from the observation that the most computational resources owned by a single mining pool in Bitcoin are 17%.

Following from this, observe that this upper bound on α_A, becomes closer to $\frac{n_f}{n_{CSL}}$ as we increase n_{CSL}. In general, for any Byzantine agreement protocol to function correctly, it requires, at the least, that the number of participants controlled by the adversary n_A, be less than one-third of total nodes, n_{CSL}. For instance, $n_f = \frac{n_{CSL}+2}{3}$ for PBFT (Castro and Liskov, 1999). Therefore, our upper bound for a fraction of network resources controlled by an Adversary A is around $\frac{1}{3}$. In most blockchain protocols, using consensus mechanisms like Proof-of-Work, Proof-of-Stake, etc., this bound is $\frac{1}{2}$ (Bitcoin). Although, one should note that in practical scenarios, α_A does not reach over $\frac{1}{5}$ or 20%. For reference, the most computational resources owned by any miner in Bitcoin is 17%. Since our consideration is much higher than this percentage, we can say that the bounds in Tiramisu are reasonable for practical scenarios.

4.3.3 Discussion

We highlight that a Tiramisu protocol can be designed using any Proof-of-X consensus protocol in the first layer, inheriting its security. Similarly, any BFT protocol in the second layer can be used. In our analysis of Tiramisu, considering three types of nodes, we proved that the protocol would be Nash-Incentive-Compatible (Definition 4.5) for three certain conditions according to Theorem 4.1. We have also proved that any Tiramisu protocol is in a secure state (Definition 4.6) with high probability under reasonable assumptions and pre-defined conditions on protocol parameters.

One must note that PeerCensus (Decker et al., 2016), Solida (Abraham et al., 2018), Hybrid Consensus (Pass and Shi, 2017a), and we believe, a few other protocols similar to Algorand (Chen and Micali, 2019) are a special instantiation of Tiramisu. In these special cases, we say that PeerCensus and Solida use a Proof-of-Work based protocol, whereas Hybrid Consensus and Algorand use a Proof-of-Stake protocol in the ACL. However, these protocols will need formal analysis to prove so, and this is left for future work. We believe our framework has further potential to be more universal and include major protocols as its instantiations.

4.4 ASHWAChain

ASHWAChain (Arora et al., 2020) is an example of a multi-layered blockchain protocol that fits in the Tiramisu framework. ASHWAchain is a committee-based blockchain protocol in which the committee is updated regularly. We present a formal protocol analysis in the full version of our paper (Arora et al.). The Access Control Layer of ASHWAchain use Proof-of-Work as a method of Sybil-resistance and PBFT (Castro and Liskov, 1999) towards achieving byzantine agreement in the Consensus Layer. As per analysis using the Tiramisu framework, we show that after selecting secure protocol parameters that ensure game-theoretic soundness, the protocol can achieve a throughput of more than 700 transactions per second, more than the claim in the original paper by selecting a smaller committee size but greater Proof-of-Work difficulty. We remark that analysis without a framework is also useful in finding optimal parameters for multi-layered blockchain protocols. Figures 4.2 and 4.3 show the effect of committee size on security and performance for ASHWAChain.

Chapter 5

Game-Theoretically Sound Reward Schemes for PoW Blockchain Protocols

Researchers have discovered several incentive-driven deviations *such as undercutting, selfish mining, etc., and economically motivated or Goldfinger attacks that threaten the security of a blockchain protocol. In this chapter, we identify the requirements that a game-theoretically sound reward scheme must meet. We show that existing reward schemes, such as the fixed block reward and transaction-fee-only schemes, fail to meet the proposed requirements for game-theoretic soundness for any PoW-based blockchain protocol. Our other significant contribution is identifying a class of general reward schemes that are game-theoretically sound for any blockchain protocol. With this, we propose a reward scheme, namely* Inflating Block Reward *scheme, that achieves game-theoretic soundness when paired with any PoW-based blockchain protocol.*

5.1 Introduction

Most literature in the field of blockchains is devoted to analysing the security and performance of blockchain protocols in models where most players are assumed to behave honestly. However, when deployed in the real world with *self-interested* players, the built-in reward model may not elicit honest behaviour at all times, as evident by the possibility of incentive-driven deviations and economic attacks. Hence in this work, we assume a setting in which self-interested or rational players participate in a PoW-based blockchain protocol and identify conditions on rewards to ensure game-theoretic soundness or incentive compatibility where players derive the maximal utility by acting as per the protocol. Based on our observations, we develop a game-theoretically sound reward scheme that does not suffer from the aforementioned issues, namely *Inflating Block Reward* scheme. Thus, we claim that game-theoretic issues should be attributed to underlying causes in the reward scheme and not the blockchain protocol.

5.1.1 Our Analysis

	Game Theoretic Property			
Reward Scheme	C1	C2	C3	IC = C1 ∧ C2
Transaction Fee-Only	✗	✗	✓	✗
Fixed Block Rewards	✓	✓	✗	✓
Inflating Block Rewards	✓	✓	✓	✓

Table 5.1: Summary of our results

[1] C1, C2 and C3 refers to the three constraints introduced in the paper and IC refers to the Incentive Compatibility Property.

In order to formally analyze the reward schemes, we propose that an ideal reward scheme must satisfy the following three criteria:

C1: *Strong Attack-Payoff Security* (Badertscher et al., 2018a) which guarantees security against incentive-driven deviations.

C2: *Ransom Reward Constraint* which ensures that a Goldfinger Attack is unprofitable. The threshold for this reward is derived in Eq. 5.3.

C3: *Transaction Fee* that incentivizes the player to include as many transactions in a block to maximize system efficiency.

We also show that Strong Attack-Payoff Security and Ransom Reward Guarantee ensure game-theoretic soundness. We analyze three classes of reward schemes, namely

1. Transaction-Fee-Only reward scheme,

2. Fixed Block Reward scheme,

3. our proposed reward scheme, Inflating Block Reward scheme,

for any general blockchain protocol. Since our results are applicable to all PoW-based blockchain protocols that fit our abstract model and independent of the design choices of the underlying blockchain protocol, we theorize that the issues that lead to possibilities of game-theoretic attacks on blockchain protocol lie in the reward schemes used rather than the protocol itself. Additionally, since our analysis is protocol-agnostic, our proposed scheme can be used in any such blockchain protocol in a plug-and-play fashion to ensure game-theoretic soundness.

5.1.2 Our Results

In this chapter, we present the following results:

1. We identify three conditions that ensure game-theoretic soundness, Strong Attack-Payoff Security, Ransom Reward Constraint and Transaction Fee.

2. The Transaction Fee-Only reward scheme fails to provide any security against game-theoretic attacks (Sec. 5.4.1).

3. The Fixed Block Reward scheme provides security against incentive-driven deviations but fails to guarantee safety against the Goldfinger Attack (Sec. 5.4.2). This reward scheme also fails to provide any incentive to fill a block efficiently with transactions.

4. Finally, we propose a novel reward scheme, *Inflating Block Reward* satisfying C1, C2 and C3. (Sec. 5.5)

Table 5.1 summarizes our results for the three reward schemes.

5.2 Stackelberg Game for a Blockchain Protocol

We model the game created by a blockchain protocol as a multi-player Stackelberg Game in which the leader is the protocol designer $\mathfrak{D}$ who selects the reward scheme $\mathcal{R}$ for the protocol followed by rational players $\mathcal{P} = \{P_1, \ldots, P_n\}$ who select the strategy s_i to follow which may be s_{Proto} (the strategy to follow the protocol proposed by $\mathfrak{D}$) or any other valid deviation. A *valid deviation* is any feasible strategy that does not remove the player P_i from the system. We call the players as players in the rest of the paper. The rational players also decide their level of participation (n) by investing the appropriate amount of computing power. Although, the blockchain mining continues for an infinite period of time, we only consider one round for game theoretic analysis. We assume $\mathcal{R}$ is fixed upfront and then the players in $\mathcal{P}$ continue to mine for infinite period. We only consider one steady state round for our game theoretic analysis. Thus it naturally fits in a single-leader-multiple-follower Stackelberg game.

Formally, for the Stackelberg Game $\Gamma = \langle \mathcal{N}, (S_i)_{i\in\mathcal{N}}, (U_i)_{i\in\mathcal{N}} \rangle$, we have the set of players $\mathcal{N} = \mathfrak{D} \cup \mathcal{P}$. The set of strategies $S_{\mathfrak{D}} = \mathcal{R}$ – Reward scheme chosen by $\mathfrak{D}$ and for $P_i \in \mathcal{P}$, $S_i = \{s_1, s_2, \ldots\}$ – set of valid deviations of the protocol by player P_i. We assume $s_{\text{proto}} \in S_i \forall P_i \in \mathcal{P}$. $U_{\mathfrak{D}}(\mathcal{R}, s)$ = Protocol designer's utility, and $U_{P_i}(\mathcal{R}, s)$ = mining players' utility where $s \in S = \prod_{i\in\mathcal{P}} S_i$. We use $I_b^{s_i,s_{-i}}$ to denote the event where the block b mined by a player with strategy s_i is inserted into the maximal chain when other players are following strategy s_{-i} .

5.2.1 Protocol Designer's Utility

The protocol designer's job is to ensure the maximal adherence of players to the protocol. Typically, a blockchain requires atleast 50% of the computational power to follow the protocol in order to ensure a secure operation.

$$U_{\mathfrak{D}}(\mathcal{R}, s) = \begin{cases} 0 & \text{if } \#s_i = s_{\text{Proto}} \geq \frac{n}{2} \\ -\infty & \text{otherwise} \end{cases}$$

5.2.2 Player's Utility

A player that participates in a blockchain protocol must incur a cost f per unit time in anticipation of a reward $\mathcal{R}(b)$ when a block b is accepted. Each player has the computational power C. If a player is able to produce a block b after mining for a time t, the utility can be written as:

$$U_{P_i}(\mathcal{R}, s) = \left(\sum_{b\in\text{Mined}(i)\cap\text{Consensus\_Chain}} \mathcal{R}(b) \right) \cdot C_R - ft$$

where Mined(i) is the set of blocks mined by P_i, Consensus_Chain is the set of blocks that are accepted to be the part of consensus chain by all players, and C_R is the conversion rate of the cryptotoken to a fiat currency,.[1]

5.3 Game Theoretic Analysis

5.3.1 Rent Seeking Competition

A vital feature of a PoW-based blockchain is that a participating player always incurs a finite cost to mine a block due to the real-world cost of operating the equipment (electricity, cooling, hardware cost, etc.). Any self-interested player can freely enter or exit the system, leading to fluctuations in the net computational power available with the network of players. In most PoW-based blockchain systems, the difficulty of mining a block is dynamically updated in order to maintain a constant block creation rate despite fluctuations in net computational power. Thus, as more players join the network, the greater the expected cost to mine the block. Many researchers have shown that players' decision to participate or exit the system can be modelled by a rent-seeking tournament (Budish, 2018; Kroll et al., 2013).

In economic terms, rent refers to the cost borne by a player while competing for a prize. A *rent-seeking competition* arises when multiple players compete with each other for the prize by offering rent. In a perfect market with free entry, a rational player will choose to participate *iff* the expected prize exceeds the rent (Konrad, 2009).

The reward for mining a block in a blockchain protocol is fixed independently of the computational power invested by the network, which leads to players joining or exiting the network based on their profitability. In our case, the rent is collected from the players by the blockchain protocol in terms of computational power or Proof-of-Work. For each player that joins the network and invests some computational power, the rent for all players increases marginally since the mining difficulty increases. At the equilibrium of the rent-seeking competition, the cost incurred by a player would be equal to the expected reward offered to the player (Mankiw and Whinston, 1986). Therefore, the number of players n at the equilibrium will be,

$$nfT^* = \mathbb{E}[\mathcal{R}(b)]P(I_b^{s_{\text{Proto}} s_{-i}}) \tag{5.1}$$

[1]In this work, we assume that C_R stays within a narrow band once the cryptotoken becomes a mature currency. ((Miller and Weller, 1991), and (Svensson, 1992))

where T^* is the expected time to mine a block, which remains a fixed parameter of the blockchain protocol

5.3.2 Goldfinger Attack Analysis

We capture the value of a Goldfinger Attack for the attacker as V_{sabotage}. The protocol designer picks a reward R such that the honest players invest a computational power or hash rate $H = nC$ where C is the hash rate of an individual player. The adversary then matches the hash rate H by purchasing specialised equipment at rate f and also bears the same electricity cost as the honest majority. Adversary's utility for attacking will be $U_{\text{adversary}}(\text{attack}) = V_{\text{sabotage}} - (ft' + e)H$ where t' is the time for which the equipment is operated and e the cost of the mining equipment.

In order to ensure the adversary does not attack, $U_{\text{adversary}}(\text{attack}) < 0$. Thus, we obtain the Goldfinger Constraint in Eq. 5.2.

$$(ft' + e)H > V_{\text{sabotage}} \tag{5.2}$$

We term the potential loss of the attacker $(-U_{\text{adversary}}(\text{attack}))$ as the Goldfinger Premium. A positive Goldfinger Premium should be enough to deter attackers. However, a negative Goldfinger Premium would imply the entire cryptotoken system is a sitting duck, waiting to be attacked by an adversary.

In this work, we assume V_{sabotage} to be a fraction of the market capitalization of a cryptotoken, i.e., $V_{\text{sabotage}} \leq \xi M \cdot C_R$ for some constant ξ where M is the cryptotoken in circulation and C_R is the conversion rate of the cryptotoken to fiat currency. Now we show that in order to make the *Goldfinger Attack* un-profitable, a reward scheme must ensure a minimum reward guaranteed to the players. We term this condition as the *Ransom Reward Guarantee.*

5.3.3 Ransom Reward Guarantee

The cost of mining k blocks required for a double spending attack can be given by substituting $t' = kT^*$, hence, we get our combined constraint for safety Eq. 5.3,

$$\mathbb{E}[\mathcal{R}(b)]P(I_b^{s_{\text{Proto}}, s_{-i}}) > R_{\min} = \left(\frac{fT^*}{kfT^* + e}\right)V_{\text{sabotage}} \tag{5.3}$$

where k is the number of blocks that must be forked in order to double-spend a transaction. Eq. 5.3 establishes the minimum reward that must be offered to the player in order to ensure safety against Goldfinger Attacks. We term this reward as the "ransom" reward that must be guaranteed to the players in order to prevent a Goldfinger Attack.

Now, we analyze various classes of reward schemes for Game-Theoretic Soundness.

5.3.4 Criterion for Game-Theoretic Soundness

Based on our analysis, we formally define three requirements for Game-Theoretic Soundness.

5.3.4.1 C1: Strong Attack-Payoff Security

In a blockchain protocol, the protocol designer must select a reward scheme such that the best response of a player would be to mine on top of the maximal block. This may be modelled as a Stackelberg game between the protocol designer and the rational attacker, similar to (Badertscher et al., 2018a). Definition 5.1 captures this notion and guarantees that a protocol would be free of incentive-driven deviations if the protocol is Strongly Attack-Payoff Secure.

Definition 5.1 (Strong Attack Payoff Security (Badertscher et al., 2018a)). *A protocol Π is strongly attack-payoff secure if the attacker playing Π is ϵ-best response to the protocol designer playing Π for a constant ϵ, i.e.,*

$$U_{\mathcal{A}}(\Pi, \phi) \leq U_{\mathcal{A}}(\Pi, \Pi) + \epsilon \quad \forall \phi \tag{5.4}$$

5.3.4.2 C2: Ransom Reward Constraint

We had shown in Sec. 2.7.1.2 that the reward scheme must guarantee a minimum reward R_{min} to the player in order to guarantee safety against the Goldfinger Attack. This threshold remains a fraction of the total cryptotoken in circulation.

$$\mathcal{R}(b) \geq R_{\text{min}} \ \forall \ b \tag{5.5}$$

5.3.4.3 C3: Transaction Fees

Lastly, it is easy to see to that an ideal reward scheme must collect transaction fee in order to incentivize the player to include as many unique transactions as possible in a new block.

5.3.5 Incentive Compatibility

The notion of incentive compatibility for a reward scheme implies that the rational players will always choose to follow the protocol. Theorem 5.1 shows that Strong Attack-Payoff Security and Minimum Reward Guarantee are sufficient to ensure Incentive Compatibility. The proof for the theorem is provided in the supplementary material.

Definition 5.2 (Incentive Compatibility). *We call a reward scheme to be incentive compatible if it can ensure that the Stackelberg Game between the protocol designer and the attacker has an ϵ-Subgame Perfect Nash Equilibrium, i.e.,*

$$U_{\mathcal{A}}(O_t(\Pi, \phi)) \leq U_{\mathcal{A}}(O_t(\Pi, \Pi)) + \epsilon \ \forall \ \phi, t \tag{5.6}$$

where $O_t(\Pi, \phi)$ represents the final outcome at the end of the execution if the protocol designer selects the protocol Π and the attacker follows the protocol ϕ at time t.

Theorem 5.1. *Strong Attack-Payoff Security and Ransom Reward Constraint imply Incentive Compatibility.*

Proof. At every step of the protocol, the notion of Strong Attack-Payoff Security guarantees that following the protocol for both is the ϵ-best response and hence, the players have no incentive to collect more reward by strategically deviating from the protocol. Ransom Reward Constraint ensures that the players do not perform a Goldfinger Attack.

Hence, at any step of the protocol's execution no player has a strategy that yields more utility than following the protocol. Since, there are no one-shot deviations from following the protocol, this implies that following the protocol is an ϵ-Subgame Perfect Nash Equilibrium (from the One-Shot Deviation Principle (Watson, 2013)), i.e.,

$U_{\mathcal{A}}(\Pi, \phi) \leq U_{\mathcal{A}}(\Pi, \Pi) + \epsilon \ \ \forall \phi$ and $\mathcal{R}(b) \geq R_{\min} \ \forall \ b \Rightarrow U_{\mathcal{A}}(O_t(\phi, \Pi)) \leq U_{\mathcal{A}}(O_t(\Pi, \Pi)) + \epsilon \ \forall \ \phi, t$ □

Strong Attack-Payoff Security alone cannot guarantee Incentive Compatibility since it does not ensure mining to be profitable. Hence, the player might prefer not to participate in the system at all. However, the two conditions together imply a stronger condition for reward schemes. An ideal reward scheme must not only ensure that following the protocol is a ϵ-Subgame Perfect Nash Equilibrium but also guarantee that players collectively receive at least $R_{\min}$ reward.

Now we analyze two popular reward schemes for game-theoretic soundness.

5.4 Analysis of Existing Reward Schemes

5.4.1 Transaction-Fee-Only Schemes

In a Transaction-Fee-Only reward scheme $\mathfrak{T}$, all rewards offered to a player is contributed by the users of the cryptotoken via transaction fees. In this case, the no additional cryptotoken can be minted by the players. Hence, this is a deflationary reward scheme.

The authors of (Badertscher et al., 2018a) show that a Transaction-Fee Only reward scheme does not meet the requirements for Attack-Payoff Security. Hence, we do not repeat the same analysis but their result is sufficient to show that such a reward scheme cannot guarantee Strong Attack-Payoff Security for any general blockchain protocol.

Theorem 5.2 assumes a general blockchain protocol and reward scheme in which the transaction fees may not only come from the transactions included in the newly mined block but also from previously mined blocks.

Theorem 5.2 (Transaction Fees Impossibility). *Any pair of blockchain $\mathcal{B}$ and a corresponding protocol Π can not satisfy C2 under $\mathfrak{T}$.*

Proof. We define $\text{Ancestors}(b) = \{x \mid x \prec b \; \forall \; x \in B\}$ as the set of all blocks that precede b.
Let us assume there exists a reward function $R()$ such that $\mathcal{R}(b) = \sum_{x \in \text{Ancestors}(b)} f(b, x)$ where $f : B \times B \to \mathbb{R}$ is a function that takes in transaction fees in block x and returns the corresponding contribution to the reward for the block b.
Since no new currency can be minted via block rewards, the player's reward cannot exceed the total transaction fees collected in the blockchain till now less the reward already distributed to the players, i.e., $\sum_{b \in B} f(b, x) \leq \text{Fee}(x) \; \forall \; x$. Total reward available to the player mining the maximal block b_{max} would be $(\sum_{b \in B} \text{Fee}(b)) + \text{Fee}(b_{\text{max}}) - |B|C_{\text{mining}}$.

If there were negligible transaction fees available due to a dearth of available transactions.
Hence, $\text{Fee}(b_{\text{max}}) = 0$.
For the blockchain protocol to satisfy the Ransom Reward Constraint, $\mathcal{R}(b) \geq R_{\text{min}}$. Thus, total available reward for the next player would be
$R(b'_{\text{max}}) \geq \sum_{b \in B} \text{Fee}(b) - R_{\text{min}}$.

If mining is continued without new transaction fees, soon there will not be any available reward to be offered to the players. At this point, $R(b_{\text{max}})$ must be $< R_{\text{min}}$.

Thus, the reward scheme R would not be able to meet the Ransom Reward Constraint and hence, it would not be satisfy C2. $\square$

A formal proof is provided in the supplementary material. The intuition for the proof lies in the fact that no matter how the protocol rewards the players, it would have a fixed pool of rewards accumulated throughout the execution of the protocol most of which would have already been distributed to the players of existing blocks. If the total transaction fees accumulated by the players per block is less than R_{min}, this pool of reward would keep on reducing until it is exhausted. Once the pool of rewards is exhausted, the player cannot collect a reward more than R_{min}. Hence, the Ransom Reward Constraint (Eq. 5.3) would not be met, making the protocol vulnerable to a Goldfinger Attack.

5.4.2 Fixed Block Reward Schemes

In a Fixed Block Reward Scheme $\mathfrak{F}$, with every block there is a fixed constant reward r minted to the player of a block. We do not consider any contribution of transaction fees in this scheme in order to ensure Strong Attack-Payoff Security. As shown in Observation 5.1, in this case, mining the maximal block is the best response strategy.

Claim 5.1. *Any blockchain $\mathcal{B}$ and protocol Π is Strongly Attack-Payoff Secure under $\mathfrak{F}$.*

The proof of Observation 5.1 is same as the proof of Theorem 5.3 by setting $\gamma = 0$.

Claim 5.2. *Any blockchain protocol Π under $\mathfrak{F}$ guarantees C2 if $P(I_b^{\Pi,\Pi})r \geq R_{min}$.*

5.4.3 Analysis of Bitcoin's Reward Scheme

Bitcoin players are rewarded via block rewards which halve every 210,000 blocks. Based on our analysis in Sections 5.4.1 and 5.4.2, we argue that Bitcoin's reward scheme will not be game-theoretically sound in the future. In case block rewards continue to halve after every four years, there should come a time when the block reward drops below R_{min} (expected to happen in the next 12 years) when a Goldfinger Attack will become profitable on the Bitcoin network. Secondly, once the block rewards are exhausted, a Transaction-Fee-Only Reward Scheme would mean that Bitcoin won't be Strongly Attack-Payoff Secure. Thus, doubt looms over the security guarantees of Bitcoin in the future.

5.5 Inflating Block Reward Scheme

In this section, we propose a novel reward scheme, the Inflating Block Reward scheme, that combines a block reward expressed as a fraction of the total cryptotoken in circulation and transaction fee that is capped to a constant γ. We show that this scheme satisfies all three criterion for game-theoretic soundness for any PoW-based blockchain protocol. Therefore, it can be used in a plug-and-play manner to make the protocol game-theoretically sound.

Our reward scheme relies on the block reward to be a fraction of the circulating cryptotoken ($M = \sum_{b \in B} \text{BlockReward}(b)$). The block reward in turn increases the circulating cryptotoken by δM, therefore, M increases at the rate of δ compounded per block. One way to implement this could be to increase the block reward periodically after a certain number of blocks.

Let Fee(b) denote the sum of transaction fees of transactions in a block and M the total amount of cryptotoken in circulation. The reward given to a player for a mining for a block b under our reward scheme $\mathfrak{R}$ parameterized by δ is:

$$\mathfrak{R}(b) = \delta M + \max(\gamma, \text{Fee}(b)) \tag{5.7}$$

where $\text{BlockReward}(b) = \delta M$ is such that $P(I_b^{S,s_{\text{Proto}}})\delta M \cdot C_R \geq \left(\frac{fT^*}{kfT^*+e}\right)V_{\text{sabotage}}$ and γ is some constant much smaller than δM.

Theorem 5.3 ($\mathfrak{R}$ is Strongly Attack-Payoff Secure). *Any blockchain $\mathcal{B}$ and protocol Π is Strongly Attack-Payoff Secure under the reward scheme $\mathfrak{R}$.*

Proof. Let $U_{\mathcal{A}}^{(\Pi,\Pi)}(b)$ denote the utility of an attacker $\mathcal{A}$ for mining a block b according to the protocol Π if all players choose to follow the protocol (Π).

$$U_{\mathcal{A}}^{(\Pi,\Pi)}(b) = P(I_b^{\Pi,\Pi})(\delta M + \max(\gamma, \text{Fee}(b)) - C_{\text{eqb}}$$
$$U_{\mathcal{A}}^{(\Pi,\Pi)}(b) \geq P(I_b^{\Pi,\Pi})\delta M - C_{\text{eqb}}$$

Let $U_{\mathcal{A}}^{(\phi,\Pi)}(b)$ denote the utility of the attacker for mining a block b according to any strategy ϕ if all other players choose to follow the protocol.

$$U_{\mathcal{A}}^{(\phi,\Pi)}(b) = P(I_b^{\phi,\Pi})(\delta M + \max(\gamma, \text{Fee}(b)) - C_{\text{eqb}}$$
$$U_{\mathcal{A}}^{(\phi,\Pi)}(b) \leq P(I_b^{\phi,\Pi})(\delta M + \gamma) - C_{\text{eqb}}$$

The utility of attacker playing Π when the protocol designer selects the protocol Π, $U_{\mathcal{A}}(\Pi, \Pi)$ is given by $U_{\mathcal{A}}^{(\Pi,\Pi)}(b)$. The utility when the attacker plays ϕ and the protocol designer selects the protocol Π, $U_{\mathcal{A}}(\Pi, \phi)$is given by $U_{\mathcal{A}}^{(\phi,\Pi)}(b)$.

$$U_{\mathcal{A}}^{(\phi,\Pi)}(b) \leq U_{\mathcal{A}}^{(\Pi,\Pi)}(b) + \gamma$$

Since $P(I_b^{\Pi,\Pi}) \geq P(I_b^{\phi,\Pi}) \ \forall \ \phi$. Therefore, any protocol Π is strongly attack-payoff secure under the reward scheme $\mathcal{R}$. $\square$

The proof for Theorem 5.3 is provided in the supplementary material. The intuition for the proof is that the block reward can vary only from δM to $\delta M + \gamma$ leaving little room for extra profit deviation. Thus, we can argue that following the protocol is the γ-best response for a player, making it Strongly Attack-Payoff Secure.

Claim 5.3. *A blockchain $\mathcal{B}$ and protocol Π satisfy C2 under $\mathfrak{R}$.*

Proof. The minimum reward under the reward scheme $\mathfrak{R}$ is $P(I_b^{S,s_{\text{Proto}}})\delta M$ which is greater than the $R_{\min}$ satisfying the Ransom Reward Guarantee (Eq. 5.3). $\square$

Hypothetical Analysis in Bitcoin

In this section, we hypothetically discuss our reward scheme if it was deployed in Bitcoin today. In order to do this, we first approximate V_{sabotage} and $R_{\min}$ and then propose adequate value of ξ for our scheme $\mathfrak{R}$.

As of January 2022, the open interest on Bitcoin Futures across all exchanges is 11.3 Billion USD, we use this value as our upper bound on V_{sabotage}. For calculating $R_{\min}$; we need to calculate the cost of obtaining mining equipment to match the Bitcoin network's computing power. The most cost-efficient computing equipment, Antminer S7 costs around 250 USD, assuming electricity cost to be around 0.06 USD per kWh pegs the $R_{\min}$ around 95k USD or 2.05 bitcoins, which is well below the current block reward of 300k USD or 6.25 bitcoins. The transaction fees contribute a fraction of the player's reward averaging around 0.1 bitcoin per block. Thus, curerently majority of the mining reward is contributed by block rewards, and the safety is inconclusive without block rewards.

From these statistics of Bitcoin, we estimate a minimum of ξ required to be 8×10^{-7} which implies an inflation of 3.8% in the cryptotoken in circulation per annum.

5.6 Conclusion

In this chapter, we discussed game-theoretic attacks that plague blockchain protocols. We show that game-theoretic issues can be attributed to existing reward schemes rather than the design of the blockchain protocol. We achieve this by formally defining three requirements to guarantee game-theoretic soundness and analyzing existing reward schemes independent of the underlying blockchain protocol to show that they do not meet our requirements. Based on our observations, we proposed a game-theoretically sound reward scheme, Inflating Block Reward scheme that satisfies all three criteria for game-theoretic soundness.

Future Work

1. We leave it to future work to explore other possible reward schemes that offer similar guarantees of game-theoretic soundness.

2. Our ideas can possibly be extended to non-PoW-based blockchains such as Proof-of-Stake blockchains. We leave for future work to examine such possibilities.

3. In this work, we assumed the players that maintain the blockchain protocol and users that perform transactions are separate entities and assume the the users merely provide transactions as input to the system. We leave it to future work to consider the possibility of users colluding with the players.

Chapter 6

Conclusion and Future Work

Blockchain Technology has transformative potential for many industries ranging from finance to healthcare. At its core lies a blockchain protocol responsible for ensuring the correct functioning of the distributed ledger. One of the critical challenges in designing blockchain protocols today is the lack of scalability to ensure greater throughputs. In this thesis, we focus on the scalability challenge and study blockchain protocols through the lens of game-theory. We present a new issue of network fairness towards scaling a Proof-of-Work-based protocol such as Bitcoin (Chapter 3). Based on a game-theoretic analysis, we identified the security threats posed due to a lack of network fairness in PoW-based blockchain protocols (Section 3.3). We also demonstrate via examples that such threats are also present in modern blockchain protocols like OHIE proposed to be more scalable (Section 3.4). Therefore, we establish that network fairness is a crucial issue to be solved in order to achieve scalability. We also study the paradigm of layering blockchain protocols for scalability and present a formal game-theoretic analysis in the BAR model (Chapter 4). Based on our analysis, we predict the parameters for optimal security (Section 4.3.1). Lastly, we provide a game-theoretic analysis of reward schemes used in Proof-of-Work-based blockchain protocols (Chapter 5) that demonstrates shortcomings of the existing reward schemes. We also propose the Inflating Block Reward scheme and prove that it ensures game-theoretic soundness (Chapter 5.5).

Thus, in this thesis, we analyze blockchain scalability using game-theoretic models and present novel issues that need to be adequately addressed by blockchain protocol designers in order to achieve scalability while ensuring security. We further contribute to blockchain protocol design by developing game-theoretic models and analyses for PoW-based and layered-blockchain protocols.

6.1 Future Work

In our work, we have identified new issues and developed new models for studying blockchain protocols. In the future, blockchain protocol designers can design protocols that guarantee network fairness in order to deter incentive-driven deviations. Layered blockchain protocols are an exciting line of work that has recently gained traction as scalability concerns have gained priority among blockchain developers. We expect our work to be useful for developing and analyzing layered blockchain protocols by providing a formal analysis.

Bibliography

Economic survey: Upi is single largest retail payment system in the country. https://www.business-standard.com/budget/article/economic-survey-upi-is-single-largest-retail-payment-system-in-the-country-122013100968_1.html. Accessed: 2021-03-30.

BitcoinStats, 2017. http://bitcoinstats.com/network/propagation/.

Documentation/TechnicalWhitepaper.md, 2020. https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md.

Shard chains, Aug 2021. URL https://ethereum.org/en/eth2/shard-chains/.

Ittai Abraham, Dahlia Malkhi, Kartik Nayak, Ling Ren, and Alexander Spiegelman. Solida: A Blockchain Protocol Based on Reconfigurable Byzantine Consensus. In James Aspnes, Alysson Bessani, Pascal Felber, and João Leitão, editors, *21st International Conference on Principles of Distributed Systems (OPODIS 2017)*, volume 95 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:19, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-061-3. doi: 10.4230/LIPIcs.OPODIS.2017.25. URL http://drops.dagstuhl.de/opus/volltexte/2018/8640.

Amitanand S. Aiyer, Lorenzo Alvisi, Allen Clement, Jean Philippe Martin Michael Dahlin, and Carl Porth. Bar fault tolerance for cooperative services. In *Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005, SOSP 2005, Brighton, UK, October 23-26, 2005*, page 45–58, 2005.

Colleen Alkalay-Houlihan and Nisarg Shah. The pure price of anarchy of pool block withholding attacks in bitcoin mining. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 1724–1731. AAAI Press, 2019. doi: 10.1609/aaai.v33i01.33011724. URL https://doi.org/10.1609/aaai.v33i01.33011724.

Yackolley Amoussou-Guenou, Bruno Biais, Maria Potop-Butucaru, and Sara Tucci-Piergiovanni. Rational vs byzantine players in consensus-based blockchains. In *In Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020, IFAAMAS, 9 pages*. IFAAMAS, 2020. URL `http://ifaamas.org/Proceedings/aamas2020/pdfs/p43.pdf`.

Sanidhay Arora, Anurag Jain, Sankarshan Damle, and Sujit Gujar. Tiramisu: Layering consensus protocols for scalable and secure blockchains. `https://www.dropbox.com/s/c1am9xbvyd51ndf/Tiramisu.pdf?dl=0`.

Sanidhay Arora, Anurag Jain, Sankarshan Damle, and Sujit Gujar. Ashwachain: A fast, scalable and strategy-proof committee-based blockchain protocol. In *Workshop on Game Theory in Blockchain at WINE 2020 (GTiB@WINE 2020)*, 2020. URL `https://econcs.pku.edu.cn/wine2020/wine2020/Workshop/GTiB20_paper_9.pdf`.

Raphael Auer and Rainer Böhme. The technology of retail central bank digital currency. *BIS Quarterly Review, March*, 2020.

Moshe Babaioff, Shahar Dobzinski, Sigal Oren, and Aviv Zohar. On bitcoin and red balloons. In *Proceedings of the 13th ACM conference on electronic commerce (ACM EC'12)*, pages 56–73, 2012.

Christian Badertscher, Juan A. Garay, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. But why does it work? a rational protocol design treatment of bitcoin. In *EUROCRYPT (2)*, pages 34–65. Springer, 2018a. doi: 10.1007/978-3-319-78375-8_2.

Christian Badertscher, Peter Gaži, Aggelos Kiayias, Alexander Russell, and Vassilis Zikas. Ouroboros genesis: Composable proof-of-stake blockchains with dynamic availability. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 913–930, New York, NY, USA, 2018b. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243848. URL `https://doi.org/10.1145/3243734.3243848`.

Shehar Bano, Alberto Sonnino, Mustafa Al-Bassam, Sarah Azouvi, Patrick McCorry, Sarah Meiklejohn, and George Danezis. Sok: Consensus in the age of blockchains. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, AFT '19, page 183–198, New York, NY, USA, 2019. Association for Computing Machinery.

ISBN 9781450367325. doi: 10.1145/3318041.3355458. URL https://doi.org/10.1145/3318041.3355458.

Soumya Basu, Ittay Eyal, and Emin Gün Sirer. Falcon - a fast bitcoin backbone. URL https://www.falcon-net.org/.

Eric Budish. The economic limits of bitcoin and the blockchain. Working Paper 24717, National Bureau of Economic Research, June 2018. URL http://www.nber.org/papers/w24717.

Vitalik Buterin. Hard problems in cryptocurrency: Five years later. https://vitalik.ca/general/2019/11/22/progress.html. Accessed: 2022-08-31.

Miles Carlsten, Harry Kalodner, S. Matthew Weinberg, and Arvind Narayanan. On the instability of bitcoin without the block reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 154–167, New York, NY, USA, 2016a. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978408. URL https://doi.org/10.1145/2976749.2978408.

Miles Carlsten, Harry Kalodner, S Matthew Weinberg, and Arvind Narayanan. On the instability of bitcoin without the block reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 154–167, 2016b.

Miles Carlsten, Harry Kalodner, S Matthew Weinberg, and Arvind Narayanan. On the instability of bitcoin without the block reward. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 154–167, 2016c.

Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance. In *Third Symposium on Operating Systems Design and Implementation, New Orleans, USA, February 1999*, pages 173–186. Laboratory for Computer Science, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139, 1999.

Dimitris Chatzopoulos, Anurag Jain, Sujit Gujar, Boi Faltings, and Pan Hui. Towards mobile distributed ledgers. *IEEE Internet of Things Journal*, 2021.

Yash Chaurasia, Visvesh Subramanian, and Sujit Gujar. Pupow: A framework for designing blockchains with practically-useful-proof-of-work & vanitycoin. In *2021 IEEE International Conference on Blockchain (Blockchain)*, pages 122–129, 2021. doi: 10.1109/Blockchain53845.2021.00026.

Jing Chen and Silvio Micali. Algorand: A secure and efficient distributed ledger. *Theoretical Computer Science*, 777:155 – 183, 2019. ISSN 0304-3975. doi: https://doi.org/10.1016/j.tcs.2019.02.001. In memory of Maurice Nivat, a founding father of Theoretical Computer Science - Part I.

Lin Chen, Lei Xu, Zhimin Gao, Keshav Kasichainula, and Weidong Shi. Nonlinear blockchain scalability: a game-theoretic perspective. *arXiv preprint arXiv:2001.08231*, 2020.

Lin Chen, Lei Xu, Zhimin Gao, Ahmed Sunny, Keshav Kasichainula, and Weidong Shi. Nonlinear blockchain scalability: a game-theoretic perspective, 2021.

Matt Corallo. Fibre - fast internet bitcoin relay engine. URL `https://bitcoinfibre.org/`.

P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels. Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 585–602, Los Alamitos, CA, USA, may 2020. IEEE Computer Society. doi: 10.1109/SP40000.2020.00040. URL `https://doi.ieeecomputersociety.org/10.1109/SP40000.2020.00040`.

Sankarshan Damle, Sujit Gujar, and Moin Hussain Moti. Fasten: Fair and secure distributed voting using smart contracts. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–3, 2021. doi: 10.1109/ICBC51069.2021.9461060.

Bernardo David, Peter Gaži, Aggelos Kiayias, and Alexander Russell. Ouroboros praos: An adaptively-secure, semi-synchronous proof-of-stake blockchain. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018*, pages 66–98, Cham, 2018. Springer International Publishing. ISBN 978-3-319-78375-8.

C. Decker, J. Seidel, and R. Wattenhofer. Bitcoin meets strong consistency. In *In Proceedings of the 17th International Conference on Distributed Computing and Networking Conference (ICDCN)*. ICDCN, 2016. URL `https://tik-db.ee.ethz.ch/file/ed3e5da74fbca5584920e434d9976a12/peercensus.pdf`.

Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *IEEE P2P 2013 Proceedings*, pages 1–10. IEEE, 2013.

Ethereum Community. Ethereum whitepaper. `https://ethereum.org/en/whitepaper/#a-next-generation-smart-contract-and-decentralized-application-platform`.

Ittay Eyal. The miner's dilemma. In *Proceedings of the 2015 IEEE Symposium on Security and Privacy*, SP '15, page 89–103, USA, 2015. IEEE Computer Society. ISBN 9781467369497. doi: 10.1109/SP.2015.13. URL `https://doi.org/10.1109/SP.2015.13`.

Ittay Eyal and Emin Gün Sirer. Majority is not enough: Bitcoin mining is vulnerable. In Nicolas Christin and Reihaneh Safavi-Naini, editors, *Financial Cryptography and Data Security*, pages 436–454. Springer Berlin Heidelberg, 2014. ISBN 978-3-662-45472-5.

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.

Chen Feng and Jianyu Niu. Selfish mining in ethereum. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 1306–1316, 2019. doi: 10.1109/ICDCS.2019.00131.

Ben Fisch, Rafael Pass, and Abhi Shelat. Socially optimal mining pools. In Nikhil R. Devanur and Pinyan Lu, editors, *Web and Internet Economics*, pages 205–218, Cham, 2017. Springer International Publishing. ISBN 978-3-319-71924-5.

Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, apr 1985. ISSN 0004-5411. doi: 10.1145/3149.214121. URL `https://doi.org/10.1145/3149.214121`.

Juan Garay, Aggelos Kiayias, and Nikos Leonardos. The bitcoin backbone protocol: Analysis and applications. Cryptology ePrint Archive, Report 2014/765, 2014. `https://eprint.iacr.org/2014/765`.

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*, SOSP '17, page 51–68, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350853. doi: 10.1145/3132747.3132757. URL `https://doi.org/10.1145/3132747.3132757`.

Cyril Grunspan and Ricardo Perez-Marco. Selfish mining in ethereum. In Panos Pardalos, Ilias Kotsireas, Yike Guo, and William Knottenbelt, editors, *Mathematical Research for Blockchain Economy*, pages 65–90, Cham, 2020. Springer International Publishing. ISBN 978-3-030-53356-4.

Charlie Hou, Mingxun Zhou, Yan Ji, Phil Daian, Florian Tramèr, Giulia Fanti, and Ari Juels. Squirrl: Automating attack analysis on blockchain incentive mechanisms with deep reinforcement learning. In *NDSS 2021, virtually, February 21-25, 2021*. The Internet Society, 2021. URL `https://www.ndss-symposium.org/ndss-paper/squirrl-automating-attack-analysis-on-blockchain-incentive-mechanisms-with-deep-reinf`

Aggelos Kiayias and Giorgos Panagiotakos. Speed-security tradeoffs in blockchain protocols. Cryptology ePrint Archive, Report 2015/1019, 2015. `https://eprint.iacr.org/2015/1019`.

Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. Ouroboros: A provably secure proof-of-stake blockchain protocol. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017*, pages 357–388, Cham, 2017. Springer International Publishing. ISBN 978-3-319-63688-7.

Uri Klarman, Soumya Basu, Aleksandar Kuzmanovic, and Emin Gün Sirer. bloxroute: A scalable trustless blockchain distribution network whitepaper. *IEEE Internet Things J.*, 2018.

Eleftherios Kokoris-Kogias, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ewa Syta, and Bryan Ford. Omniledger: A secure, scale-out, decentralized ledger via sharding. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 583–598, 2018. doi: 10.1109/SP.2018.000-5.

Kai Andreas Konrad. *Strategy and dynamics in contests*. Oxford University Press, 2009.

Joshua A Kroll, Ian C Davey, and Edward W Felten. The economics of bitcoin mining, or bitcoin in the presence of adversaries. In *Proceedings of WEIS*, volume 2013, page 11, 2013.

Yixiao Lan, Yuan Liu, Boyang Li, and Chunyan Miao. Proof of learning (pole): Empowering machine learning with consensus building on blockchains (demo). *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(18):16063–16066, May 2021. URL `https://ojs.aaai.org/index.php/AAAI/article/view/18013`.

Chenxin Li, Peilun Li, Dong Zhou, Zhe Yang, Ming Wu, Guang Yang, Wei Xu, Fan Long, and Andrew Chi-Chih Yao. A decentralized blockchain with high throughput and fast confirmation. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, pages 515–528, 2020.

Songze Li, Mingchao Yu, Chien-Sheng Yang, Amir Salman Avestimehr, Sreeram Kannan, and Pramod Viswanath. Polyshard: Coded sharding achieves linearly scaling efficiency and security simultaneously. *IEEE Transactions on Information Forensics and Security*, 16:249–261, 2021. doi: 10.1109/TIFS.2020.3009610.

Kevin Liao and Jonathan Katz. Incentivizing blockchain forks via whale transactions. In *International Conference on Financial Cryptography and Data Security*, pages 264–279. Springer, 2017.

Ziyao Liu, Nguyen Cong Luong, Wenbo Wang, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. A survey on applications of game theory in blockchain, 2019.

N. Gregory Mankiw and Michael D. Whinston. Free entry and social inefficiency. *The RAND Journal of Economics*, 17(1):48–58, 1986. ISSN 07416261. URL `http://www.jstor.org/stable/2555627`.

Will Martino, Monica Quaintance, and Stuart Popejoy. Chainweb : A proof-of-work parallel-chain architecture for massive throughput. 2018.

Andrew Miller, Ari Juels, Elaine Shi, Bryan Parno, and Jonathan Katz. Permacoin: Repurposing bitcoin work for data preservation. In *2014 IEEE Symposium on Security and Privacy*, pages 475–490, 2014. doi: 10.1109/SP.2014.37.

Marcus Miller and Paul Weller. Exchange Rate Bands with Price Inertia. *The Economic Journal*, 101(409):1380–1399, 11 1991. ISSN 0013-0133. doi: 10.2307/2234891. URL `https://doi.org/10.2307/2234891`.

Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. URL `http://www.bitcoin.org/bitcoin.pdf`.

Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 305–320, 2016. doi: 10.1109/EuroSP.2016.32.

Michael Neuder, Daniel J. Moroz, Rithvik Rao, and David C. Parkes. Selfish behavior in the tezos proof-of-stake protocol. *CoRR*, abs/1912.02954, 2019. URL `http://arxiv.org/abs/1912.02954`.

Martin J Osborne and Ariel Rubinstein. *A course in game theory*. MIT press, 1994.

Rafael Pass and Elaine Shi. Hybrid Consensus: Efficient Consensus in the Permissionless Model. In Andréa W. Richa, editor, *31st International Symposium on Distributed Computing (DISC 2017)*, volume 91 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 39:1–39:16, Dagstuhl, Germany, 2017a. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-053-8. doi: 10.4230/LIPIcs.DISC.2017.39. URL `http://drops.dagstuhl.de/opus/volltexte/2017/8004`.

Rafael Pass and Elaine Shi. Fruitchains: A fair blockchain. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 315–324, 2017b.

Serguei Popov. The tangle.

F. Ritz and A. Zugenmaier. The impact of uncle rewards on selfish mining in ethereum. In *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS PW)*, pages 50–57, 2018. doi: 10.1109/EuroSPW.2018.00013.

Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 515–532. Springer, 2016a.

Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. Optimal selfish mining strategies in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 515–532. Springer, 2016b.

Fabian Schär. Decentralized finance: On blockchain- and smart contract-based financial markets, 03 2020.

Shoeb Siddiqui, Ganesh Vanahalli, and Sujit Gujar. Bitcoinf: Achieving fairness for bitcoin in transaction fee only model. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, page 2008–2010, Richland, SC, 2020a. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Shoeb Siddiqui, Ganesh Vanahalli, and Sujit Gujar. Bitcoinf: Achieving fairness for bitcoin in transaction fee only model. In Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith, editors, *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 2008–2010. International Foundation for Autonomous Agents and Multiagent Systems, 2020b.

Yonatan Sompolinsky and Aviv Zohar. Accelerating bitcoin's transaction processing. fast money grows on trees, not chains. Cryptology ePrint Archive, Report 2013/881, 2013. `https://eprint.iacr.org/2013/881`.

Yonatan Sompolinsky, Yoad Lewenberg, and Aviv Zohar. Spectre: A fast and scalable cryptocurrency protocol, 2016. `https://eprint.iacr.org/2016/1159`.

Yonatan Sompolinsky, Shai Wyborski, and Aviv Zohar. Phantom and ghostdag: A scalable generalization of nakamoto consensus. Cryptology ePrint Archive, Report 2018/104, 2018. `https://eprint.iacr.org/2018/104`.

Chrysoula Stathakopoulou. A faster bitcoin network. 2015.

Lars E. O. Svensson. An interpretation of recent research on exchange rate target zones. *Journal of Economic Perspectives*, 6(4):119–144, December 1992. doi: 10.1257/jep.6.4.119. URL `https://www.aeaweb.org/articles?id=10.1257/jep.6.4.119`.

Mahnush Movahedi Timo Hanke and Dominic Williams. Dfinity technology overview series consensus system. CoRR abs/1805.04548 (2018), 2018. URL `https://arxiv.org/pdf/1805.04548.pdf`.

Visa Inc. Visanet booklet. `https://usa.visa.com/dam/VCOM/download/corporate/media/visanet-technology/visa-net-booklet.pdf`.

Qin Wang, Jiangshan Yu, Shiping Chen, and Yang Xiang. Sok: Diving into dag-based blockchain systems. *CoRR*, abs/2012.06128, 2020. URL `https://arxiv.org/abs/2012.06128`.

Qin Wang, Rujia Li, Qi Wang, and Shiping Chen. Non-fungible token (NFT): overview, evaluation, opportunities and challenges. *CoRR*, abs/2105.07447, 2021. URL `https://arxiv.org/abs/2105.07447`.

Joel Watson. *Strategy: An Introduction to Game Theory.* W W Norton & Co Inc, 2013.

Wikipedia contributors. Front running — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Front_running&oldid=944160433`, 2020. [Online; accessed 12-June-2020].

H. Yu, I. Nikolic, R. Hou, and P. Saxena. Ohie: Blockchain scaling made simple. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 112–127, Los Alamitos, CA, USA, May 2020. IEEE Computer Society. doi: 10.1109/SP.2020.00008. URL `https://doi.ieeecomputersociety.org/10.1109/SP.2020.00008`.

Mahdi Zamani, Mahnush Movahedi, and Mariana Raykova. Rapidchain: Scaling blockchain via full sharding. In *Proceedings of the 2018 ACM SIGSAC CCS*, CCS '18, page 931–948, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243853. URL `https://doi.org/10.1145/3243734.3243853`.

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 (S&P)*, pages 175–192, 2019. doi: 10.1109/SP.2019.00086.