Selfish Mining and Networking Effects

Thesis zur Erlangung des Grades
Master of Science (M. Sc.)
im Studiengang Computer Science

Alexander Wagner
`wagner.2@campus.tu-berlin.de`

Distributed Security Infrastructures
Institut für Softwaretechnik und Theoretische Informatik
Fakultät Elektrotechnik und Informatik
Technische Universität Berlin

Gutachter:
Prof. Dr. Florian Tschorsch
TBA: Second supervisor

eingereicht am: TBA

Hiermit erkläre ich an Eides statt, dass die vorliegende, dieser Erklärung angefügte Arbeit selbstständig und nur unter Zuhilfenahme der im Literaturverzeichnis genannten Quellen und Hilfsmittel angefertigt wurde. Alle Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind kenntlich gemacht. Ich reiche die Arbeit erstmals als Prüfungsleistung ein.

Berlin, TBA

. . . .

(Alexander Wagner)

Zusammenfassung

Abstract

Contents

Chapter 1

Introduction

Bitcoin is the most prominent example of a decentralized cryptocurrency [2]. Before the development of Bitcoin a decentralized cryptocurrency had been envisioned for many years. It is a system, where a ledger is kept consistent among multiple parties in a peer-to-peer network without the need of trust. It enables the deployment of electronic cash without a central authority figure like a bank. For this reason it is an enhancement to the currently established electronic banking system.

It is important to keep the ledger consistent and correct. Since there is no central authority, the ledger becomes a consensus problem, which has to be solved in a cooperative, distributed manner. It can also be seen as a Byzantine Agreement problem [5], because multiple indepedent parties have to agree on the state of the ledger. Solving this problem is the main contribution of the development of Bitcoin.

Bitcoin assumes an honest majority in a public system [14]. Therefore, the consistence and correctness of the ledger becomes a voting problem. If every participant in the system is able to cast one vote, then the ledger is correct and consistent if there exists an honest majority. However, voting in a public distributed system remains a hard problem, especially considering sybil attacks [3]. Sybil attacks enable an attacker to forge identities and obtain a dishonest majority. To solve this Bitcoin has to protect against sybil attacks, without a central authority figure. Bitcoin achieves this by binding voting right to computational power. Since computational power is a ressource harder to increase than the number of forged identities, obtaining a dishonest majority becomes much harder.

Bitcoin binds votes to computational ressources through cryptographic puzzles. In order for a peer to participate in the system, he has to solve a cryptographic puzzle. This process, also known as mining, consumes the computational ressources of the peer. Simply forcing participants to waste all their computing power, only to participate in the system would lead to no participants. Therefore, this mining process has to be incentivised. If a miner mines a block, he receives a mining reward. This helps spreading the overall computational power of the network among multiple different parties, since every party is competing for mining rewards. Without a mining reward there be no economical reason to spend computational ressources on bitcoin.

Mining is a process, which builds inherently on the idea of incentives. It is logical, that miners

will strive for the best strategy to maximize rewards. A mining protocol maximizing rewards is called incentive compatible. It can also be assumed that a miner will always execute the mining protocol, which maximizes rewards. The original bitcoin mining protocol is assumed to be incentive compatible [2]. However, Eyal and Sirer show the existence of deviant mining protocols with greater rewards [4]. Miners executing such protocols are called selfish miners. This imposes a threat, since it reduces the performance of the overall system. Additionally, selfish miners obtain a greater voting power than their computational ressources allow and as a result tilt the honest majority balance.
The central goal of this master thesis is to analyze the impact of selfish mining as an attack on blockchain systems. While it has been established that selfish mining imposes a threat on blockchain, it remains unassessed how big the impact is. Additionally, selfish mining is highly influenced by networking effects. Therefore, in order to assess the impact of selfish mining, analysis has to be performed in a model, which also captures the underlying network.

Chapter 2

Related Work

Selfish mining is a statistical attack. To analyze profitibality it is therefore beneficial to analytically model selfish mining. In order to study the impact of deviating mining strategies it is very important to represent the blockchain network as close to reality as possible in a mining model, to estimate realistic results. In the following recent selfish mining models as well as network models will be discussed.

2.1 Selfish Mining Models

Blockchain Mining is most commonly modelled through markov decision processes. It is generally used to model decision making, where the outcomes are influenced by random processes and the decision of the decision maker [8]. For the case of selfish mining the selfish miner chooses his next action, so he controls the decision making process. The rest of the network, the block arrival and block propagation can be modelled by stochastic processes. Once implemented, a markovian model can be used to analytically estimate system properties. Selfish mining is concerned with maximizing obtained mining rewards, which is also called revenue.
Eyal and Sirer first described a selfish mining model based on markov decision processes [4]. They simulated their model to estimate revenue gain. Since blockchain uses a network to propagate mined blocks, it is important to analyze how the network was implemented in the model. Block propagation time is considered negligible compared to block generation time [4]. As a result Eyal and Sirer consider communication to be instantaneous [4]. Selfish increases profitiability for a relative mining power greater than 25% compared to the network.
Sapirshtein et al. further extended the model to consider all possible actions a selfish miner can perceive. Block propagation time remains unassessed, since it is again considered to be much smaller than block generation time. Sapirshtein et al. again utilize markov decision processes to model the system. They find a number of optimal policies and provide an analysis on the upper bound of revenue increase through selfish mining strategies. This markovian model was widely used and adopted in other research directions studying other aspects of selfish mining. For example the behavior of multiple selfish miners was simulated through Leelavimolsilp et al. [10]. One of the main findings is that the lower bound on the profitability

treshhold decreases with the number of selfish miners. Bai et al. [1] extended the model of Sapirshtein et al. [13] even further to analyze multiple selfish miners. This resulted in a more complex state space of the markov decision process. They show that in fact for the symmetric selfish miner, all selfish miners have the same hashrate, the profitability threshhold decreases, but for the asymmetric case the threshhold increases. The focus of this research was to deepen the understanding of different selfish mining strategies. However, networking factos remained unassessed.

Xiao et al. study the impact on the profitability threshhold and revenue gain of a networking advantage possessed by the selfish miner [15]. They model the network as a graph and find that networking advantage correlates to the betweenness centrality of the selfish miner. Additionally it highly affects the profitability threshhold and revenue gain of the attacker. This indicates that the structure of the network influences the selfish mining strategy. However, this model remains very abstract, since only the peer-to-peer layer and structure is modelled as a graph, disregarding any limitations imposed by physical infrastructure such as bandwidth. Nonetheless, it indicates that the underlying network influences the blockchain overlay, strengthening the assumption that there is a highly influencial dependency between networking effects and selfish mining.

It is not contested by any of the previous research, that network capabilities and communication delay impact selfish mining [10], although most research model block propagation as instantaneous. In addition, most research which is concerned with selfish mining, builts on top of the model presented by Sapirshtein et al.. This contributes to the negligence of networking effects, when analyzing selfish mining. Assuming that the underlying network does influence the system built on top, this master thesis aims to analyze the impact of networking effects on selfish mining. It is therefore important to represent the network in the model, that is used to analyze selfish mining.

2.2 Blockchain Network Models

Bitcoin and Proof-of-Work blockchains in general have been additionally modelled and analyzed from a networking perspective. In order to study selfish mining with the context of networking effects it is necessary to analyze the network. Most blockchain network models are concerned with the estimation of consistency. Consistency is the property of a blockchain that all honest parties output the same block sequence. Garay et al. study the core of the bitcoin protocol formally [5]. They analyze the protocol in a synchronous communication network and show persistence and liveness of committed transactions. They further proof that the adversarial computational power bound to reach Byzantine Agreement is $1/2$ of the network for a synchronized network. The adversarial bound decreases as the network drifts further away from synchronization [5]. The Analysis of Garay et al. indicate that the networks synchronicity highly influences the behavior of Proof-of-Work blockchains.

Pass et al. propose a new network model to analyze blockchains in terms of consistency and liveness in an asynchronous network [12]. They do not make any assumptions of synchronicity and proof consistency in a network with adversarial delays that are a-priori bounded. They show that the proof of work hardness needs to be set as a function of the maximum network de-

lay. New peers joining the network or peers getting corrupted are also modelled. They proove that Nakamotos protocol satisfies consistency even in a network with message delays. However, those network delays are modelled to be always adversarial. This leaves out networking factors impacting the system, which are caused by honest behavior.
Kiffer et al. built on top of the models of Garay et al. and Pass et al., but formulate a simple markov chain based method to analyze consistency. Additionally they provide lower bounds for consistency. They also analyze the GHOST protocol, where consensus is built over the heaviest observed subtree, in addition to the longest chain rule [9]. The model is based on rounds of communication. The modelled adversary controls a fraction of honest peers and can delay and reorder messages within a threshhold δ. The model therefore again captures only network attacks from an adversary, but disregards other networking effects.
Gervais et al. introduce a novel framework to analyse security and performance [6]. They model how network and consensus parameters influence stale block rate, block propagation times, throughput and security. Stale blocks are blocks which do not contribute to the consensus mechanism. Selfish mining is modelled as a markov decision process. The network layer is characterized by block size and the information propagation mechanism. Gervais et al. simulate the system over a network consisting of point-to-point connections between peers [6]. Those channels are defined by latency and bandwidth. Latency is set using global IP latency statistics. One major result is that an increasing block size increases block propagation time linearly and stale block rate exponentially.
Gopalan et al. utilize rumor-spreading to implement a new stochastic network model [7]. They study stability and scalability of their model. Each peer communicates at a given rate his oldest blocks to his neighbors. Communication channels are also bandwidth limited. This setup introduces network delays to blocks, which depends on the instantaneous network congestion. Unlike previous stochastic network models Gopalan et al. do not introduce delay based on sampling data, but rather on the communication behavior of peers. Since network congestion depends on the behavior of peers and selfish mining is a deviating behavior, the model introduced by Gopalan et al. will be used in the following to analyze selfish mining and networking effects.

Chapter 3

Model

In order to model networking effects and selfish mining, it is essential to capture network properties in an analytic model. The model can then be used to estimate selfish mining profitability. Gopalan et al. have introduced a new blockchain model, which captures network properties.

3.1 Bitcoin Mining Fundamentals

To understand selfish mining and its implications on network behavior it is essential to understand bitcoin mining in general. Bitcoin utilizes proof-of-work blockchain as a distributed ledger technology. It includes transactions into so called blocks. Blocks possess a unique ID and reference a previous block [14]. This construct builds a directed acyclic graph. The root of this tree is also called genesis block. Thus, every block directly or indirectly references the genesis block.
A correct block includes a nonce, which solves a cryptographic puzzle. The challenge is to alter the nonce until the hash of the set of transactions, the hash of the previous block and the nonce produce a partial hash collision. Essentially, the hash has to be smaller than some threshold value, which is also referred to as difficulty [14]. Thus, Bitcoin binds block creation to the computational ressources a peer possesses, since the partial hash collision can only be solved through trial and error. The correctness of the block is easily verifiable through third parties. Thus, Bitcoin ensures a fair leader election through this process.
Bitcoin uses a peer-to-peer network to propagate the mined blocks in the system. The network is unstructured as every peer tries to maintain a minimum of eight connections and performs neighbor discovery over DNS, IRC and asking neighbors [14]. Blocks are propagated over the peer-to-peer layer through flooding. Once a miner mines a block through solving a cryptographic puzzle, he can publish the block and receives rewards through transaction fees and mining rewards. This provides an incentive to the miner to generate as many correct blocks as possible [2].
Consensus is established over the longest chain rule [2]. This means that the block ending the longest chain determines the state of the blockchain. This also implies that a miner only receives rewards, if his mined blocks are included in the main chain. Thus, a miner wants

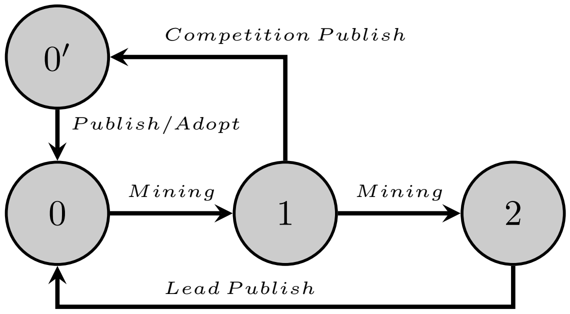


Figure 3.1: Abstract representation of state transtitions of eyal and sirer model for one selfish miner

to produce as many correct blocks, that are part of the main chain, as possible. A protocol maximizing reward gain is thus incentive compatible. A miner produces a relative share of blocks proportionally to his relative share of computational power of the whole network. Thus, a miner should produce a relative share of the main chain proportional to his relative share of computational power.
The original protocol, also called honest mining, assumes publishing blocks immediately after mining. Honest mining is assumed to be incentive compatible [2]. It follows that no miner can earn disproportionate rewards by deviating from the protocol. Consequently, earning disproportionate rewards through deviation from the honest mining protocol, would disprove Bitcoin's incentive compatability claim.

3.2 Original Selfish Mining Model

One protocol deviation is selfish mining, which was first introduced by Eyal and Sirer [4]. Selfish Mining is a vulnerability, which aims at increasing revenue through block withholding. The selfish miner aims at producing a greater relative share of blocks of the main chain, than the relative share of computational power of the network. Therefore, selfish mining violates Bitcoins incentive compatibility claim, as it offers a more profitable mining protocol than honest mining. This is problematic, since it not only breaks fair leader election, but also results in potentially longer confirmation times for transactions of users. Eyal and Sirer model the network over a set of miners. A miner finds a subsequent block after a time interval that is exponentially distributed [4]. They further define the revenue of a miner as his fraction of total blocks on the longest chain. The selfish miner withholds mined blocks [4]. This selfish miner now possesses a private chain, which differs from the publicly known chain. Based on the difference between those two chains, the selfish miner performs actions. For clarification the state space and actions are modelled in Figure 3.1. The numbers in the states indicate the lead of the private to the public chain. s denotes the lead of the private chain compared to the public chain. We can identify a total of five different actions.

- $Mining$: This action means that the peer has mined block. Mining adds the block to the private chain. It therefore causes s to increase.

- $Lead\ Publish$: When s increases to 2, the selfish miner will publish his private chain. It therefore causes s to change from 2 to 0.
- $Competition\ Publish$: When s is 1 and the selfish miner receives a block from another peer, he will publish his block of the same height from the private chain instead of the received one, to compete against the other miner. This causes a state transition to $0'$.
- $Publish$: If the selfish miner is in state $0'$, he is in a competition situation. The selfish miner will immediately publish his next mined block. This will cause the selfish miner to transition to state 0.
- $Adopt$: The selfish miner will adopt the main chain once he receives a new block in a competition situation.

Executing this protocol leads to a strict revenue increase, if the mining power is greater than 33% according to Eyal and Sirer [4].

3.3 Blockchain Gossip Model

The Blockchain Gossip Model of Gopalan et al. consists of a set of peers P connected through a peer-to-peer network. Peers add blocks to the blockchain through a process called mining. The peer-to-peer network is modelled as an undirected Graph $H = (V, E)$. An edge $(i, j) \in E$ represents communication possibilities between $v_i \in V$ and $v_j \in V$. The set of vertices is finite, such that $|V| = N \in \mathbb{N}$. Vertices are associated with peers, such that v_i represents peer $p_i \in P$. Additionally, a directed acyclic graph $G_{p_i}(t) = (B_{G_{p_i}}(t), E_{G_{p_i}}(t))$ is associated with each peer p_i, at each point in time $t \in \mathbb{R}+$. The vertex set $B_{G_{p_i}}(t) \subset \mathbb{N}$ represents the blocks known of peer p_i at time t. The associated edge set of $E_{G_{p_i}}(t)$ represents references between blocks. The following holds true for shorter notations:

$$B_G(t) = \cup_{i=1}^{N} B_{G_{p_i}}(t) \text{ and } E_G(t) = \cup_{i=1}^{N} E_{G_{p_i}}(t) \tag{3.1}$$

Furthermore, the following equations hold for the principle of blockchains:

$$\forall p \in P : G_{p_i}(0) = (\{0\}, \emptyset) \tag{3.2}$$

$$t_1 < t_2 \rightarrow B_{G_{p_i}}(t_1) \subseteq B_{G_{p_i}}(t_2) \tag{3.3}$$

$$t_1 < t_2 \rightarrow E_{G_{p_i}}(t_1) \subseteq E_{G_{p_i}}(t_2) \tag{3.4}$$

Note that in this representation 0 denotes the genesis block described in Equation 3.2. $G_{p_i}(t)$ evolves over time. Blocks arrive over continuous time according to a stationary point process A with intensity λ. Each block $b \in \mathbb{N}$ arrives at a random peer p_i. This models peer p_i mining block b at time t and that at this time the block is also added to $B_{G_{p_i}}(t)$. References are added to $E_{G_{p_i}}(t)$ according to policy and depending on $G_{p_i}(t^-)$, where t^- is a moment in time infinitesimally before t. O_i denotes the set of outgoing neighbors of block i.

The communication is modelled as a marked point process T_{p_i}. Each mark corresponds to another peer $p_j \in P \backslash \{p_i\}$. In an epoch peer p_i contacts p_j and thus, adds the lowest numbered block of $B_{p_i}(t) \backslash B_{p_j}(t)$ to the set of Vertices B_{p_j}. If $B_{p_i}(t) \backslash B_{p_j}(t)$ is not empty, E_{p_j} is also updated accordingly.
The peer-to-peer network dynamics are modelled as a continuous time rumor-spreading process with exogenous arrivals [7]. Since communication is bound to the process T_{p_i}, the block dissemination is bandwidth limited. Reference selection and thus O_{p_i} is chosen accordant to longest chain policies [7]. Let $L_{p_i}(t)$ denote the set of nodes farthest away from the genesis block 0, known to peer p_i at time t.

$$L_{p_i}(t) := \{j \in B_{p_i}(t) : d(j, 0) \geq d(j', 0), \forall j' \in B_{p_i}(t)\} \tag{3.5}$$

Let $max\_dist(G_{p_i}(t))$ denote that distance. Note that the set $O_{p_i} \cap L_{p_i}(t)$ is non empty. This constructs a simple directed acyclic graph. The Tree Policy [7] can then be determined as $|O_{p_i}| = 1$ and establishes the following relationship:

$$|E_{G_{p_i}}(t)| = |B_{G_{p_i}}(t)| - 1 \tag{3.6}$$

Every block will have exactly one outgoing reference, according to some deterministic rule [7]. Gopalan et al. assume that the block with the lower index number will be chosen.

3.4 Extended Blockchain Gossip Model

The selfish mining attack is described as a peer executing a protocol deviant from honest mining [4]. Therefore a selfish miner can be modelled according to the model described in Subsection 3.3 through altering the reference selection and communication process. The reference selection process is policy driven, and can thus be modified by providing a new selfish policy.
The first aspect to be modified is the communication process. Key idea of selfish mining is block withholding. The selfish miner possesses three blockchain representations.

- $G_{SM_{public}}(t)$: which is updated by other peers.
- $G_{SM_{comm}}(t)$: which is used to update other peers.
- $G_{SM_{private}}(t)$: with the following relations:
  - $G_{SM_{public}}(t) \subseteq G_{SM_{private}}(t)$.
  - $G_{SM_{private}}(t) \backslash G_{SM_{public}}(t)$ represents blocks mined but unpublished by the selfish miner.

The concept has been visualized in Figure 3.2. A total number of five processes is used to let all entities interact with each other.

- $Arrival\ Process\ A$: Blocks arrive to the selfish miner over the external arrival process A.
- T_{p_i}: Ensures blocks from other peers are communicated to $G_{SM_{public}}(t)$.

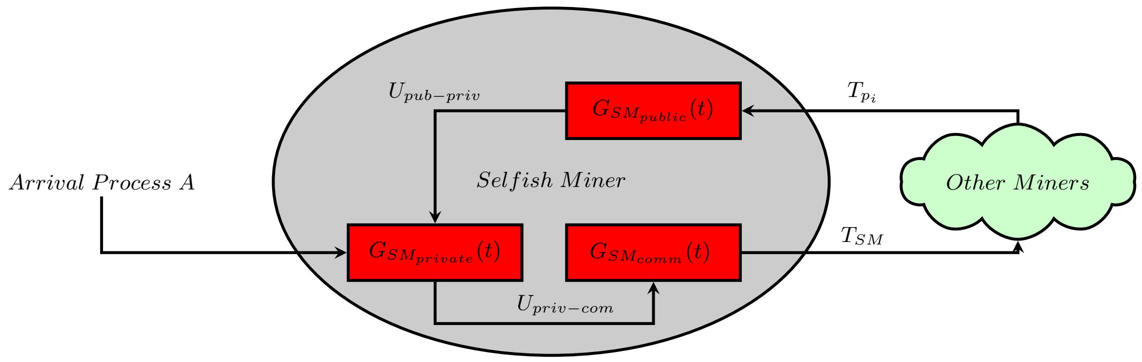


Figure 3.2: Abstract representation of model entities and communication processes

- $U_{pub-priv}$: Ensures that $G_{SM_{public}}(t) \subseteq G_{SM_{private}}(t)$ holds true, meaning $U_{pub-priv}$ updates $G_{SM_{private}}(t)$, when new blocks arrive to $G_{SM_{public}}(t)$ from other peers.

- $U_{priv-com}$: Updates $G_{SM_{comm}}(t)$ according to $G_{SM_{private}}(t)$ and the selfish mining rules S.

- T_{SM}: Ensures other peers are updated with blocks from $G_{SM_{comm}}(t)$.

Peer $SM \in P$ has an associated policy slightly different to the policy described in the gopalan model 3.5. Note that to follow the Tree Policy [7], a deterministic rule has to be established for the case that $|O_{SM} \cap L_{SM}(t)| > 1$. Assume that SM has the knowledge of the set of blocks mined through him, $M_{SM}(t) \subset B_{G_{SM}}(t)$. SM will set

$$(L_{SM}(t) \cap M_{SM}(t)) \neq \emptyset \rightarrow L'_{SM}(t) \subset (L_{SM}(t) \cap M_{SM}(t)) \tag{3.7}$$

It then follows that $|L'_{SM}(t)| = 1$. This modified tree policy sets references according to the original selfish mining protocol described by Eyal and Sirer.
S is a set of rules which describes how $G_{SM_{private}}(t)$ updates $G_{SM_{comm}}(t)$. The rules have to follow the state description of Eyal and Sirer 3.2. Therefore we need a state variable describing the difference between private and public chain. Let s be the state variable determining selfish mining actions [4]. Then s can be described as a difference between $G_{SM_{private}}(t)$ and $G_{SM_{public}}(t)$.

$$max\_dist\_mined(G_{SM_{private}}(t)) := d(j,0), j \in M_{p_i}(t) \tag{3.8}$$

$$s(t) := max\_dist\_mined(G_{SM_{private}}(t)) - max\_dist(G_{SM_{public}}(t)) \tag{3.9}$$

Let t_{inc} refer to the set of times, where s increased and analogous t_{dec} refer to the set of times, where s decreased. Selfish mining is protocol, which needs a formulation of states in order to be characterized. Gopalan et al. introduced t^- as a point in time infinitesimally before t. In addition to describe selfish mining a function is needed to access the point in time where s changed last. Let $f_{-1}(t)$ be a function that outputs the point in time, where s changed the

latest before t. Now all tools are available to characterize the selfish mining protocol on top of the stochastic network model introduced by Gopalan et al..
$U_{priv-com}$ can be characterized through four kind of update actions. Analogous to Subsection 3.2, those actions are $Lead\ Publish$, $Competition\ Publish$, $Publish$ and $Adopt$. $Mining$, the fifth action described in Subsection 3.2, is modelled through the arrival process. This can be used to model the selfish mining protocol desribed by Eyal and Sirer.

1. $Lead\ Publish$: Assume $t \in t_{inc}$ and $s(t) \geq 2$, then $U_{priv-com}$ updates $G_{SM_{comm}}(t)$, such that $G_{SM_{comm}}(t) = G_{SM_{private}}(t)$. Once the selfish miner has established a lead of two blocks against the public chain, he will update the blockchain representation used for communication towards other peers. In other words, he publishes the private chain.

2. $Competition\ Publish$: Assume $t \in t_{dec}$, $s(t) = 0$, $s(f_{-1}(t)) = 1$, $s(f_{-1}(t)^{-}) = 0$. This means that the selfish miner mined a block, did not publish it and now received a block from another of the same height. This leads to the competition scenario. Accordingly, $U_{priv-com}$ updates $G_{SM_{comm}}(t)$, such that it includes the subgraph induced by the nodes on the paths between $L'_{SM}(t)$ and 0. The selfish miner will publish the block, which caused the private chain to lead by one against the public chain, before he received a new block. This transitions to

$$0'(t) \rightarrow \big(t \in t_{dec} \wedge s(t) = 0 \wedge s(f_{-1}(t)) = 1 \wedge s(f_{-1}(t)^{-}) = 0\big) \tag{3.10}$$

   This situation $0'$ is also shown and visualized in Subsection 3.2 and causes the selfish miner to execute honest mining for only the next step.

3. $Publish$: Assume $0'(t^{-}) = \top$ and $t \in t_{inc}$, $U_{priv-com}$ updates $G_{SM_{comm}}(t)$, such that it includes the subgraph induced by the nodes on the paths between $L'_{SM}(t)$ and 0. The selfish miner will publish his newly mined block, because he was previously in $0'$.

4. $Adopt$: Assume $0'(t^{-}) = \top$ and $s(t) = -1$, then $U_{priv-com}$ updates $G_{SM_{comm}}(t)$, such that $G_{SM_{comm}}(t) = G_{SM_{private}}(t)$. The selfish miner will adopt the public chain, because he was previously in $0'$.

This concludes the selfish mining extension of the Gopalan Model. In the next chapters simulative analysis of this model will be focused.

Chapter 4

Contribution

The goal of this thesis is to analyze the relationship between selfish mining and networking effects. Therefore, the model proposed by Gopalan et al. [7] has been enhanced to model selfish mining behaviour in 3.4. The main contributions are based on simulative evaluations and can be categorized as the following:

1. Implementation and Verification of Blockchain Gossip Model [7]
2. Integration of selfish mining as described in 3.4 in Blockchain Gossip Model
3. Analysis of networking characteristics
4. Analysis of selfish mining characteristics
5. Model verification

Contributions are building blocks for each other. Integration of selfish mining is only possible when the Blockchain Gossip Model has been implemented. As such the following sections will discuss contributions in a developing and chronological order.

4.1 Implementation and Verification of Simpy Blockchain Simulator

The core implementation of the Blockchain Gossip Model Simulator is based on simpy [11]. The implementation will be referred to as Simpy Blockchain Simulator. Simpy is a process based discrete event simulator. The Blockchain Gossip Model described by Gopalan et al. [7] consists of four parts.

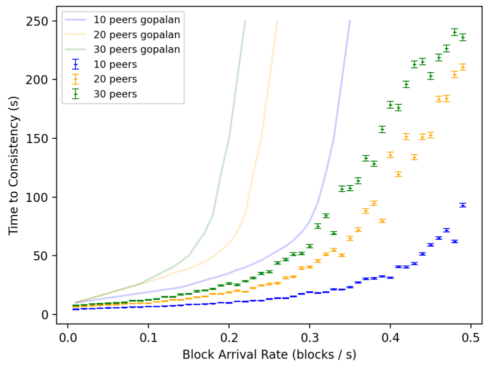
- Networkgraph representation
- Blockchain representation
- Block Arrival Process representation
- Communication Process representation

Simpy Implementation of Blockchain Gossip Model Simulator The network graph is represented by an adjacency matrix. In the synthetic data experiments of Gopalan et al. [7] they analyze the complete graph for 10, 20 and 30 peers. Thus, the matrix representing the network graph is an unit matrix. The blockchain representation is a set of blocks and a set of edges for each peer, which are developing over time. The block arrival process and the communication process are modelled as a poisson process. Specifically to a discrete event simulator, this implies that the interarrival time between scheduled events follows an exponential distribution. On each event of the block arrival process a block arrives uniformly at a random peer. At each event of the communication process T_i a peer p_i tries to update a certain peer p_j according to the epoch associated with the event. The block to be updated is deterministically chosen. According to Gopalan et al. [7] the block with the lowest index number is chosen. Additionally, the average rate of the communication processes is set to one.

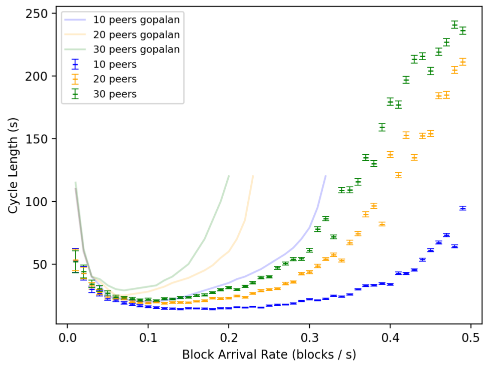
Evaluation of Simpy Blockchain Simulator on synthetic data experiments Gopalan et al. introduce four key metrics to analyze the system. Those are

- Time to Consistency — The average time an inconsistent system needs to reach a state of consistency
- Cycle Length — The sum of the average time to consistency and the average of the time the system stays consistent
- Consistency Fraction — The average fraction of peers that are consistent at each point in time
- Age of Information — The average number of blocks an average peer is away from the consistency state

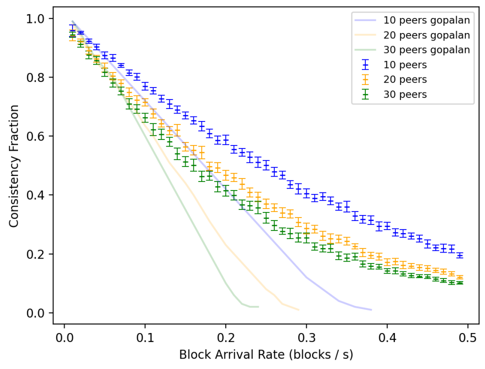
All metrics mentioned above refer to the term consistency. Gopalan et al. [7] define consistency as $B(t)$ 3.1, the unison of all blocks produced by the block arrival process A. These metrics can be used to verify whether the Simpy Blockchain Simulator is achieving similar numbers to the implementation of Gopalan et al. [7].
The metrics of time to consistency and cycle length are very closely related, because both rely on the time the system needs to reach consistency. Figure 4.1a and Figure 4.1b show this close relationship. Additionally the comparison between the Simpy BLockchain Simulator shows a very similar tendency in both metrics. Especially in Figure 4.1a it is observable that the curve has the same tendency, but has a flatter increase when the Block Arrival Rate increases. Figure 4.1a shows that an increase in number of peers or an increasing Block Arrival Rate leads to an increase in the average Time to Consistency. Since Cycle Length is the sum of the average time to consistency and the average of the time the system stays consistent the same behavior can be observed in Figure 4.1b. Additionally Figure 4.1b shows that for very small numbers for the Block Arrival Rate the Cycle Length increases again. When the system has a low Block Arrival Rate the system tends to stay longer in a state of consistency, which is due to the fact that the idle time increases.
Consistency Fraction and Age of Information are both metrics measuring the consistency of an average peer. The Consistency Fraction is the fraction of peers, which have a blockset equal



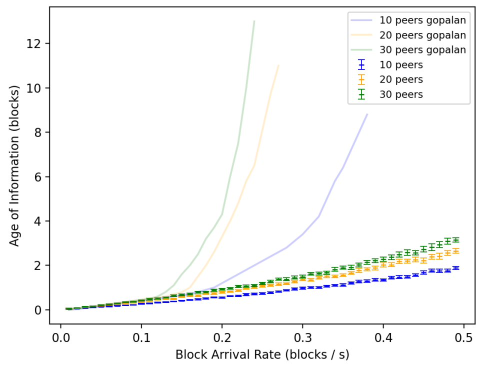
(a) Time to Consistency, Comparison between Blockchain Gossip Model Simulator based on simpy and values produced by Gopalan et al. [7]



(b) Cycle Length, Comparison between Blockchain Gossip Model Simulator based on simpy and values produced by Gopalan et al. [7]



(c) Consistency Fraction, Comparison between Blockchain Gossip Model Simulator based on simpy and values produced by Gopalan et al. [7]



(d) Age of Information, Comparison between Blockchain Gossip Model Simulator based on simpy and values produced by Gopalan et al. [7]

to $B(t)$ 3.1. For both the simulation results by Gopalan et al. [7] and the Simpy Blockchain Simulator we can observe, that the Consistency Fraction decreases with an increasing block-rate and peer number. While the exact numbers do differ, similar tendencies can again be observed. The Age of Information metric analyzes how much an average peer differs from $B(t)$ 3.1. It showcases an increase for an increasing blockrate and peer number.

The differences between the simulation results by Gopalan et al. [7] and the Simpy Blockchain Simulator indicate that information spreads faster in the Simpy Blockchain Simulator. This is most probably due to the fact that the implemented communication processes are truly concurrent.

4.2 gopalan modell extension

In the previous section the Simpy Blockchain Simulator was evaluated on the synthetic data experiments presented by Gopalan et al. [7] and achieved similar results. As a foundation for implementing selfish mining in the Simpy Blockchain Simulator extensions and modifications have to be realised. Those include

- Modification of the underlying statistical distribution of the Block Arrival Process
- Specifications regarding reference selection process
- Policy compliant consensus definition

Modification of the underlying statistical distribution of the Block Arrival Process In the original Blockchain Gossip Model proposed by Gopalan et al. [7], the Block Arrival Process selects the peer, where the next block arrives uniformly at random. Analysis of the Bitcoin network suggests that this is not an appropriate representation. Since computing power is not spread evenly throughout the network [**?**], uniform random arrivals are not an appropriate representation.

Specifications regarding reference selection process

Policy compliant consensus definition - pre selfish mining
- gopalan blockchain metriken umdefinieren sodass sie auf bäumen funktionieren
- addit. network metrics, block propagation etc.

4.3 selfish mining extension

Chapter 5

Evaluation

Chapter 6

Conclusion

Bibliography

[1] Q. Bai, X. Zhou, X. Wang, Y. Xu, X. Wang, and Q. Kong. A deep dive into blockchain selfish mining. Cryptology ePrint Archive, Report 2018/1084, 2018. `https://eprint.iacr.org/2018/1084`.

[2] S. Barber, X. Boyen, E. Shi, and E. Uzun. Bitter to better — how to make bitcoin a better currency. In A. D. Keromytis, editor, *Financial Cryptography and Data Security*, pages 399–414, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-32946-3.

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

[4] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. *CoRR*, abs/1311.0243, 2013. URL `http://arxiv.org/abs/1311.0243`.

[5] J. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 281–310. Springer, 2015.

[6] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun. On the security and performance of proof of work blockchains. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 3–16, 2016.

[7] A. Gopalan, A. Sankararaman, A. Walid, and S. Vishwanath. Stability and scalability of blockchain systems, 2020.

[8] O. Ibe. *Markov processes for stochastic modeling*. Newnes, 2013.

[9] L. Kiffer, R. Rajaraman, and A. Shelat. A better method to analyze blockchain consistency. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 729–744, 2018.

[10] T. Leelavimolsilp, L. Tran-Thanh, and S. Stein. On the preliminary investigation of selfish mining strategy with multiple selfish miners. *CoRR*, abs/1802.02218, 2018. URL `http://arxiv.org/abs/1802.02218`.

[11] N. Matloff. Introduction to discrete-event simulation and the simpy language. *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August*, 2 (2009):1–33, 2008.

[12] R. Pass, L. Seeman, and A. Shelat. Analysis of the blockchain protocol in asynchronous networks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 643–673. Springer, 2017.

[13] A. Sapirshtein, Y. Sompolinsky, and A. Zohar. Optimal selfish mining strategies in bitcoin. *CoRR*, abs/1507.06183, 2015. URL `http://arxiv.org/abs/1507.06183`.

[14] F. Tschorsch and B. Scheuermann. Bitcoin and beyond: A technical survey on decentralized digital currencies. *IEEE Communications Surveys and Tutorials*, 18:1–1, 03 2016. doi: 10.1109/COMST.2016.2535718.

[15] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou. Modeling the impact of network connectivity on consensus security of proof-of-work blockchain, 2020.