Journal of Cybersecurity and Privacy

MDPI

Article

Stochastic Modelling of Selfish Mining in Proof-of-Work Protocols †

Caspar Schwarz-Schilling [1,‡], Sheng-Nan Li [2,‡] and Claudio J. Tessone [3,*]

[1] Department of Economics, University of Zurich, CH-8050 Zurich, Switzerland; caspar.schwarz-schilling@ethereum.org
[2] Blockchain & Distributed Ledger Technologies, University of Zurich, CH-8050 Zurich, Switzerland; shengnan.li@uzh.ch
[3] UZH Blockchain Center, University of Zurich, CH-8050 Zurich, Switzerland
* Correspondence: claudio.tessone@uzh.ch

† This paper is an extended version of our paper published in Schwarz-Schilling, C.; Li, S.N.; Tessone, C.J. Agent-based Modelling of Strategic behaviour in PoW Protocols. In Proceedings of the 2021 Third International Conference on Blockchain Computing and Applications (BCCA), Tartu, Estonia, 15–17 November 2021.

‡ These authors contributed equally to this work.

Abstract: In blockchain-based systems whose consensus mechanisms resort to Proof-of-Work (PoW), it is expected that a miner's share of total block revenue is proportional to their share of hashing power with respect to the rest of the network. The protocol relies on the immediate broadcast of blocks by miners, to earn precedence in peers' local blockchains. However, a deviation from this strategy named *selfish mining* (SM), may lead miners to earn more than their "fair share". In this paper, we introduce an agent-based model to simulate the dynamics of SM behaviour by a single miner as well as mining pools to understand the influence of (a) mining power distribution, (b) overlay network topology, (c) positioning of the selfish nodes within the peer to peer network. Our minimalistic model allows us to find that in high levels of latency, SM is always a more profitable strategy; our results are very robust to different network topologies and mining nodes' centrality in the network. Moreover, the power-law distribution of the miners' hashing power can make it harder for a selfish miner to be profitable. In addition, we analyze the effect of SM on system global efficiency and fairness. Our analysis confirms that SM is always more profitable for hashing powers representing more than one-third of the total computing power. Further, it also confirms that SM behaviour could cause a statistically significant high probability of continuously mined blocks opening the door for empirical verification of the phenomenon.

Keywords: blockchain; Proof-of-Work; selfish mining; agent-based model; hashing power; network latency

Citation: Schwarz-Schilling, C.; Li, S.-N.; Tessone, C.J. Stochastic Modelling of Selfish Mining in Proof-of-Work Protocols. *J. Cybersecur. Priv.* **2022**, 2, 292–310. https://doi.org/10.3390/jcp2020016

Academic Editor: Danda B. Rawat

Received: 11 April 2022
Accepted: 17 May 2022
Published: 20 May 2022

1. Introduction

In late 2008, an internet persona named Satoshi Nakamoto introduced Bitcoin to the world. Bitcoin is a "peer-to-peer electronic cash system" [1], and consists of a network of participants that maintain a shared view on a blockchain. A blockchain is a set of blocks, where each block in turn includes a set of transactions. To process new transactions, blocks are periodically formed by a group of network participants called miners. To produce these blocks is computationally very costly. Miners are incentivised to do this work honestly by being rewarded with a block reward in the form of cryptocurrency. When a miner mines a new block, she shares it with her peers in the network. If participants follow certain rules, consensus on the newly proposed block may form and is thus appended to the blockchain. The above protocol is also named as Proof-of-Work (PoW) consensus. The conventional wisdom asserts that the PoW protocol is incentive-compatible. However,

a crucial assumption is that a majority of network participants act honestly and are not controlled by a single colluding entity.

In 2014, Eyal and Sirer [2] introduced an attack vector on the Bitcoin protocol. They suggest that a miner that controls more than one-third of the network's computational resources may break Bitcoin's incentive compatibility. They find that a miner strategically deviating from the Bitcoin protocol may lead to an increase in their relative revenue. They call this strategy selfish mining. The main idea of selfish mining is to make honest miners waste computational resources on blocks that are not going to be part of the main chain [3], i.e., get no block reward. A selfish miner may achieve this by keeping newly mined blocks private, rather than broadcasting them immediately. This effectively causes a chain split, although honest miners do not know about it initially. Honest miners continue to mine on the *public chain*, whereas the selfish miner mines on their *private chain*. The selfish miner only broadcasts their private blocks as honest miners catch up to the private chain's height, with the intention to invalidate honest blocks. If a selfish miner is relatively more profitable they can invest more resources into computational hardware, further increasing their share of the network's computational resources. This cycle may eventually culminate in the selfish miner controlling the majority of the network and thus breaking Bitcoin's incentive compatibility [4].

To illustrate selfish mining in detail, consider block tree (a) of Figure 1, in which a selfish miner, say, Alice, controls 20% of the network's hashing power ($\alpha = 0.2$) and the honest miners collectively control 80% ($(1 - \alpha) = 0.8$). We use Eyal's [2] definition of α as the selfish miner's share of the network's total hashing power. In scenario (a) in Figure 1, Alice mines block 3 on top of block 2, but does not publish it in an attempt to further increase her advantage. Thus, Bob does not know about block 3 so he continues to mine on top of block 2. Bob mines the next block, which he immediately broadcasts to the network. Upon the receival of Bob's newly mined block 4, Alice will publish her privately kept block 3 in an attempt to avoid losing the block reward. At this point, it is a race between Bob's block and Alice's block. The next block that is mined decides, whether block 3 or block 4 will be part of the main chain. Another honest miner, say, Charlie, mines the next block on top of block 4. Consequently, Alice lost the race and wasted her computational resources on a block that without acting selfishly would have, with almost certainty, been part of the main chain. The potential advantage of selfish mining could also come at the risk of losing block revenue for not publishing the block immediately. However, consider scenario (b) in Figure 1, in which a selfish miner, still say Alice, controls 40% of the network's hashing power ($\alpha = 0.4$) and the honest miners collectively control 60% ($(1 - \alpha) = 0.6$). With blocks 9 and 10, Alice has managed to build up a lead of two blocks over the public chain. Now Bob, an honest miner, finds block 11 and immediately publishes it. Upon receival of block 11, Alice will publish her privately kept two blocks (9 and 10), making it the longest chain and thus the main chain. Once Bob receives block 10, he will acknowledge the longer chain as the main chain. Hence, Bob's block 11 ends up being an orphan. Alice made Bob waste resources on a block that was never going to end up in the main chain. This is the mechanism by which selfish mining may increase a selfish miner's fair share of block rewards.

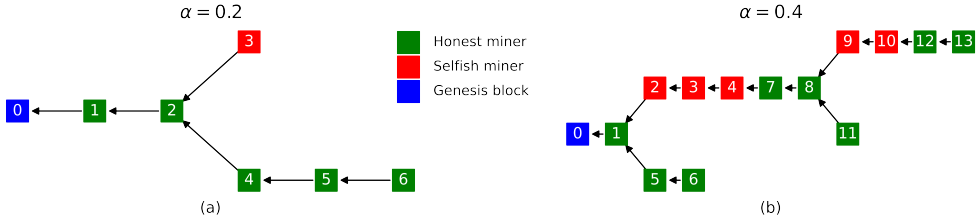


Figure 1. Visualisation of block trees for different selfish shares of the network's hashing power (α). (**a**) in which selfish miner controls 20% share of the hashing power; (**b**) in which selfish miner controls 40% share of the hashing power.

Note that if all miners behave *honestly*, then in expectation, a miner's share of total block revenue is equal to the miner's share of the network's hashing power. This is considered to be the *fair share* [5]. That is to say the fair share of the revenue for the selfish miner in scenario (a) would have been 0.2, but the realised selfish share of revenue was 0. This is due to the failed selfish mining attack. In scenario (b), in which two successful selfish mining attacks took place, the fair revenue share would have been 0.4, but the realised share of the revenue for the selfish miner was 0.5. These two scenarios illustrate that selfish mining may not always be a profitable strategy [6]. The main determinant for profitability is the selfish miner's share of the network's hashing power as well as the ability to win block races.

Given the far-reaching implications of this attack vector, it is worth investigating whether selfish mining is a profitable strategy. To achieve this, we introduce an agent-based model to study the dynamics and influences of selfish mining attacks, which is expanded from our previous conference paper [7]. Our main research questions in this paper are:

- RQ1: What is the effect of network latency, network topology, node's centrality and heterogeneous hashing power distributions on the profitability of selfish mining?
- RQ2: What is the effect of selfish mining on the network efficiency?
- RQ3: What is the effect of selfish mining on the fairness of reward distribution?
- RQ4: Which observable characteristics do selfish mining behaviours produce?
- RQ5: What is the difference in dynamics between a system with a single selfish miner, and multiple selfish miners cooperating in a mining pool?

The paper is organised as follows: Section 2 provides a literature review of selfish mining strategies and defensive approaches; Section 3 introduces the modelling approach and Section 4 the measurement after simulation; Section 5 shows the analysis results for single selfish miners; Section 6 analysis results for multiple selfish miners cooperating in a mining pool. Finally, Section 7 draws conclusions and poses some future research.

2. Literature Review

selfish mining (SM) was first introduced as an attack vector by Eyal and Sirer [2] in 2014, and it is sometimes referred to as *block withholding* [8]. This strategy shows that under certain conditions a deviation from the standard Proof-of-Work protocols may be a more profitable strategy for a miner. The definition of SM attack has attracted widespread attention and many extensions to the strategy have been proposed, such as stubborn mining and publish-*n* strategy [3,9,10]. As a response, many approaches to defending honest actors against selfish mining have also been proposed. They can be categorised into two streams: (1) making fundamental changes to the block validity rules, for example, *ZeroBlock* [11]: A timestamp-free solution which requires that each block must be generated and received by the network within a maximum acceptable time interval, and (2) decreasing the probability of honest miners working on the selfish miner's chain during a fork, for example, *weighted FRP* [12]: Here miners compare the weight of the chains instead of their length. In 2020, Nicolas [13] summarised 20 primary selfish mining attack countermeasures using the proposed taxonomy of defensive strategies and analyzed the benefits and limitations. From his summary, one can find that most of the existing modelling methods of selfish mining are focused on optimizing the accuracy of detection, lacking the analysis of the importance of the underlying network dynamics in selfish mining.

In 2021, Tessone et al. [14] introduced a stochastic model of consensus in blockchain-based systems, with Proof-of-Work (PoW) as a primary example. The authors show a rapid decrease in the fraction of time the system is in consensus when the network delay decreases, and further show that heterogeneity in mining power leads to an amplified inequality in blocks contained in the main chain, while improving the overall consensus efficiency. The results are consistent with [15] who deployed a synthetic Ethereum network with pre-defined, adjustable network delays. Tessone's minimalistic model allows flexibly adjusting network delay and interblock time as control parameters, as well as the distribution of

mining power, which provides the basis for further research of the dynamics of the peer-to-peer network under PoW protocol.

In addition, it is debated whether the selfish mining strategy is profitable: Some studies indicate that selfish mining may let attackers gain extra revenue and break the balance between revenue and mining power [3], while others argue that selfish miners can never earn more revenue but only put themselves at risk for no gain [6]. Under which conditions selfish mining is a profitable strategy, and how the selfish mining influences the system's dynamic are questions that need more answers. In this paper, we try to answer these questions by extending Tessone's stochastic method to agent-based modelling of the selfish mining strategy, and after simulation measuring various global and local properties of the system.

3. Agent-Based Model and Simulation

Tessone [14] introduced a "minimalistic stochastic model of blockchain-based systems, with Bitcoin as a primary example". Our modelling of the selfish mining behaviour in PoW protocol is based on and inspired by their work. We adopt their notation and extend the logic of the model to include the possibility for miners to deviate from the PoW protocol to mine selfishly.

In this section, we will briefly describe each core component of our agent-based model(ABM) and define the entire algorithm in Section 3.1. Then the simulation of the ABM model will be introduced in Section 3.2.

3.1. ABM of Selfish Mining

3.1.1. Peer-to-Peer Network

The foundation of the agent-based model (ABM) is a set $\mathcal{N}$ of N nodes interconnected through a static graph $G(\mathcal{N}, \mathcal{E})$ representing the system's peer-to-peer network, which we will henceforth refer to as Bitcoin. The set $\mathcal{E}$ includes all connections among nodes, so that they can communicate with one another. The Bitcoin protocol specifies a default connection size of eight peers for a new node joining the network. But the average number of connections is effectively higher than the default value, because a node may be open to new incoming connections, meaning that over time the number of peers of a node tends to increase [16]. For the simulations, we assume an average degree of 10. To account for heterogeneity in the degree distribution, we consider different network topologies, such as uniform random, Erdos-Rényi, Barabási-Albert [17,18].

Further, we make the simplifying assumption of a static topology, meaning that miners do not enter or exit the network. The assumption also implies that miners do not drop or form any new connections with their peers in a certain time period. For selfish mining, the defining time scale is that of emerging consensus (block creation and propagation), which should be considerably lower than that of nodes altering their connections, or entering and exiting the network altogether. We, therefore, argue this to be a reasonable assumption.

In this peer-to-peer network, we call all nodes *miners*. A miner is either an honest or selfish type. The former will follow the standard Bitcoin protocol, the latter will perform the selfish mining strategy. For the sake of simplicity, we assume a single selfish miner in the simulations, which implies that $(N-1)$ miners are honest.

3.1.2. Block Mining

All miners have certain computational (hashing) power which they devote to solving the PoW problem. If mining honestly, the more computational power a miner invests, the greater its possibility to mine the next block first. As the agent of our model, each miner is assigned certain computational power share, c_i, such that $\sum_{i=1}^{N} c_i = 1$. Further, we assume that the selfish miner controls a share of α of the network's total computational power. Thus, honest miners control $(1-\alpha)$ of the network's hashing power. Bitcoin's hashing power distribution is very heavy-tailed. A single entity controlled as much as 21% of Bitcoin's total mining power [19]. To reflect this heterogeneity of computational resources

across miners, we consider various hashing power distributions in the simulations, such as uniform random, power-law, and exponential.

Blocks are mined continuously and independently at a constant average rate. Thus, the PoW algorithm that determines block creation is a Poisson process. Therefore, the time between events—here an event is the creation of a new block—follows the exponential distribution. In Bitcoin, the network adjusts the difficulty to mine blocks every 2016 blocks (or in expectation roughly every two weeks) such that the average inter-block time τ_b remains at 10 min, which is denoted as $\tau_b = 10'$. Given that we assume there is no such difficulty adjustment in our simulation, if there is selfish mining activity in the network, blocks are expected to be mined at a slower rate, because relatively more hashing power is wasted on outdated and orphaned blocks. Again, we justify this simplification with the fact that the defining time scale for selfish mining is much lower than the time it takes to find 2016 blocks. Thus, the probability that miner i finds a new block in a given small time interval dt is

$$p_i^m(t) \approx \eta_i dt = c_i \tag{1}$$

Equation (1) implies that the time it takes miner i to find a new block follows the exponential distribution with parameter η_i. Note that since computational power is normalised, c_i is equal to miner i's share of the network's total computational power, meaning that the probability of mining a new block is proportional to the miner's share of the network's computational power. Further, given that the block creation process is a Poisson process, the number of blocks mined by all miners in a given interval follows a Poisson distribution, but with parameters $\sum_i \eta_i$. Hence, it is immediate to derive that $\sum_{i=1}^{N} \eta_i = {\tau_b}^{-1}$.

3.1.3. Honest and Selfish Mining

Each block in the chain is of a certain height h_b (number of blocks directly chained together between block b and the genesis block). Further, at any point in time, t each miner has a local copy of the blockchain $B_i(t)$. Importantly, honest miners deem the chain valid that has cumulatively spent most Proof-of-Work, which in the ABM, for simplicity, always corresponds to the longest chain. Thus, at time t, the height of miner i's local copy of a blockchain is defined as

$$H_i(t) = \max_{b \in B_i(t)} (h_b) \tag{2}$$

According to the longest chain rule, honest miner i will accept block b and add it to her local copy of the blockchain if the height of her local copy of the blockchain is less than the height of the block just received, namely if $H_i(t) < h_b$. As a result $H_i(t)$ increases to match block b's height h_b. Similarly, if miner i mines a block at time t (instead of receiving a new block), this also increases $H_i(t)$. In addition, honest miners naively share blocks that they either just accepted or mined with all of their peers immediately. On the contrary, a selfish miner does not necessarily share a block with honest miners immediately. To decide when to broadcast which block, selfish miners keep track of the difference in height between the public and private chain, as well as the number of blocks they have attached to the private chain. We defined an algorithm that is executed by selfish miners following the logic of Eyal's selfish mining specification in [2]. The full *pseudo-code* was shown in our previous conference paper [7].

3.1.4. Block Propagation

As previously described, after accepting or mining a new block, the block is broadcasted through the peer-to-peer network. In reality, block propagation times depend on many parameters: actual distance, network traffic, block size, etc. Blocks vary in size depending on the number of transactions included in them. A larger block takes longer to send from node i to node j: "size matters" [16]. For the sake of simplicity, it is assumed that block propagation is also a Poisson process with parameter λ_{nd}^{-1}, the same average value for all edges in the peer-to-peer network. Thus, the average time for a block to be propagated from node i to one of her peers is distributed exponentially, namely $\sim \exp(-\lambda_{nd})$. In

numerical simulation, we use the parameter λ_{nd} to control the *network delay* (or *network latency*). A larger λ_{nd} reflects a lower network delay, meaning that blocks will diffuse more quickly through the network.

3.2. Simulation

The ABM evolves over time as events occur. There are two types of events that can occur either (1) block creation: a new block is mined by some miner, or (2) block propagation: a block is propagated from one miner to another. The evolution of events can be modelled very efficiently with the *Gillespie algorithm* [20]: "It [...] numerically simulates the time evolution of the given [...] system. [...] This algorithm never approximates infinitesimal time increments dt by finite time steps Δt" [21].

Initially, at time $t = 0$, miners located in an underlying network are endowed with some computational power. Each miner i independently mines block at a Poisson rate η_i. Thus, at the global level new blocks are created at a rate $\sum_i \eta_i = {\tau_b}^{-1}$. Blocks, on the other hand, are propagated from node i to her peers at a rate λ_{nd}. Importantly, blocks are propagated by both the honest and selfish miners according to rules specified in Section 3.1.3. The ABM keeps track of all nodes that are broadcasting in any given state of the model, as well as all peers to which each node is broadcasting. The latter we define as the number of *active links* E_a, again following [14]'s notation. It implies that the rate of block propagation between two nodes connected by an active link occurs at an aggregate rate of $E_a\lambda_{nd}$.

All events independently occur following the rate at which *any* event occurs which is defined as $\xi = \sum_i \eta_i + E_a\lambda_{nd}$. Effectively, the Gillespie algorithm computes a waiting time such that the event occurrences in the system follow the distributions outlined in Section 3.1.2 for block mining and Section 3.1.4 for block propagation. To compute the waiting time, the next event is selected with probability proportional to its rate of occurrence in the system. Thus,

$$p^p = \frac{E_a\lambda_{nd}}{\sum_i \eta_i + E_a\lambda_{nd}} \tag{3}$$

is the probability that the next event is a propagation event. If the next event is a propagation event, then the probability with which a specific active link is executed is random, since we assume λ_{nd} to be equal for all active links. Similarly,

$$p^c = \frac{\sum_i \eta_i}{\sum_i \eta_i + E_a\lambda_{nd}} \tag{4}$$

is the probability that the next event is a block creation event. If the next event is a block creation event, then the probability with which a specific node is picked to be the miner is proportional to $\eta_i / \sum_j \eta_j$, which in turn is proportional to node i's share of the network's hashing power c_i.

Finally, after every event, time within the model is updated from t to $(t + t')$ with $t' \sim \exp(\xi)$.

To further consider the centrality of the selfish miner within the topology, one could assign the selfish miner to various positions in the network using various centrality measures, for example, the betweenness centrality which is given by the expression:

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\gamma_{st}(v)}{\gamma_{st}} \tag{5}$$

where γ_{st} is the total number of shortest paths from node s to node t and $\gamma_{st}(v)$ is the number of those paths that pass through v. Therefore, in the *betweenness centrality method*, we assign the miner v with the highest betweenness, $\max_{i \in \mathcal{N}}(C_B(v))$, as the selfish miner, and then randomly assign all honest miners.

In addition, to capture the effect of a correlation between node position and its hashing power (e.g., because nodes with more resources could place themselves better in the overlay network) we introduce the concept of *hashing betweenness*. In this setting, the ranking of hashing powers of honest nodes is the same as the ranking of their betweenness centrality. Different from the betweenness centrality method, after assigning the node with the highest betweenness centrality as the selfish miner, in the *hashing betweenness method*, we also assign the honest miners with relatively higher betweenness centrality more hashing power. As we know, nodes with higher betweenness centrality are more likely to have considerable influence within a network by their control over information propagation between others. Thus, more hashing power to miners with relatively high betweenness centrality, implies that the selfish miner could more quickly find out about rival blocks and can promptly respond accordingly. We expect this might be beneficial to a selfish miner.

4. Measurement

We want to measure both the global and local properties of the system after modelling the selfish mining strategy. First of all, we briefly outline the difference between global and local properties. Global properties are properties that emerge from the dynamics of the system as a whole. While local properties also emerge from the dynamics of the system, their logical meaning is independent of the global level. A local property may be averaged to make a statement about the global system, but effectively it is merely an average of local characteristics. An example of a global property is the Gini coefficient; it only exists on the global level, because one cannot make a statement regarding the fairness of block reward distribution when considering a single node. An example of a local property is the relative revenue of the selfish mining pool, which is node-centric, but can be averaged to obtain a global value.

For the global properties, we measure the Gini coefficient, orphan block rate, and consensus block rate. For the local properties, we measure the relative selfish revenue and MSB index (statistical anomaly of the selfish miner).

4.1. Gini Coefficient

To study the fairness of the system, we compute the well-established *Gini coefficient* for both the hashing power and block reward distribution. The Gini index is widely used to quantify the distribution of income and wealth [22,23]. It can range from zero, implying complete equality, to one, implying complete inequality. It is defined as

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} w_i - w_j}{2n \sum_{i=1}^{n} w_i} \tag{6}$$

To compute the block reward (or wealth distribution), we count the number of main chain blocks mined by each miner. Then the wealth of each miner i is defined as $w_i = \sum_{b \in M} \delta(i, m_b)$, where δ is the Kronecker delta that equals to 1 if $m_b = i$ (the miner of the block b is miner i), otherwise $\delta = 0$. A comparison of the Gini coefficients will reveal, whether the presence of selfish mining distorts the fairness of the system. Because in expectation, *ceteris paribus*, the *fair* block reward distribution should be proportional to the hashing power distribution.

4.2. Relative Selfish Revenue

The fundamentally important property to determine whether selfish mining is a more profitable strategy than honest mining is relative selfish revenue. If it exceeds the fair share (block rewards are proportional to hashing power), then selfish mining is relatively more

profitable. Let there be N miners, out of which S nodes are selfish; Let miners $1, \ldots, S$ be selfish and miners $(S+1), \ldots, N$ be honest. Then relative selfish revenue is defined as

$$R_{selfish} = \frac{\sum_{i=1}^{S} w_i}{\sum_{i=1}^{S} w_i + \sum_{i=(S+1)}^{N} w_i} \tag{7}$$

Notably, if we assume there is a single selfish miner, it implies in Equation (7) S simply equals 1.

4.3. Orphan Block Rate

A key characteristic of a consensus protocol, such as Bitcoin, is its efficiency. A measure of Bitcoin's efficiency is its orphan block rate because it indicates how much hashing power is wasted on blocks that are not part of the canonical chain (main chain). The orphan block rate is the probability of a block not being part of the main chain. The system produces a set of blocks of length $|B|$, with a subset of blocks as part of the main chain, with length $|M|$. Thus, the orphan block rate can be defined as,

$$O = \frac{|B| - |M|}{|B|}. \tag{8}$$

4.4. Statistical Anomalies in Selfish Mining

Li [24,25] proposes a method for detecting selfish mining behaviour in empirical networks. Their central idea is to exploit the increased probability for selfish miners to publish multiple blocks in a row. They assume that selfish miners always use the same address for the coinbase transaction (block reward) so that they can be identified as the same miner. To validate this methodology, we implement their *Miner Sequence Bootstrapping model* (*MSB*).

Following the *MSB* notation, we describe our statistical approach. First, we count the number of times that in period T miner i continuously discovers two blocks, C_i^T. Next, we shuffle the chain repeatedly and after each shuffle t we count the number of times that miner i continuously discovers two blocks in period T, which we call $S_i^T(t)$. Based on the repeated shuffle simulation, we define $\langle S_i^T \rangle$ as the expected number of times a miner i should continuously discover two blocks in period T. Then, the *MSB* index for miner i is defined as

$$MSB_i^T = \frac{C_i^T - \langle S_i^T \rangle}{\sigma[S_i^T]} \tag{9}$$

where $\sigma[S_i^T]$ is the standard deviation of all observations $S_i^T(t)$. This being a z-score, the *MSB* value is statistically significant at the 95% level for $MSB_i > 2$. The larger MSB_i index means that for consecutively discovering two blocks during period T, miner i succeeded more times than expected (i.e., based on her revenue share). Therefore, we compute an average *MSB* score for the selfish miner set in our ABM, in order to test whether the selfish mining behavior would divulge such identifiable statistical anomaly.

5. Results

For convenience, we first summarise all baseline parameters in this section. The peer-to-peer network is simulated with 100 miners, one of which is a solo selfish miner. Miners with a significant amount of hashing power are likely to be connected to other miners directly. That way they hear about newly mined blocks more quickly and thus waste less computational power on "old" blocks. This increases their profitability. Hence, we believe it to be justified to only model mining nodes as they predominantly determine the propagation dynamics relevant for mining blocks. While this is a relatively small network, it is justified by the fact that most mining activity is concentrated in very few nodes. Gencer [19] finds that "the top four Bitcoin miners have more than 53% of the average mining power" (over some period of time). Meanwhile, miners are on average

connected to 10 peers. All results simulate 10,000 min, which translates to an expected total of 1000 blocks. This in turn corresponds to roughly one week of mining. Further, all results are averaged over 50 repetitions.

While we also consider different hashing power distributions and topologies, when not otherwise stated, the results show uniform random distributions for both the degree distribution (topology) and hashing power distribution.

5.1. Relative Selfish Revenue

In this Section 5.1, we discuss our the results considering the profitability of selfish mining with varying network delays (Section 5.1.1), hashing power distribution (Section 5.1.2) and network topologies (Section 5.1.3).

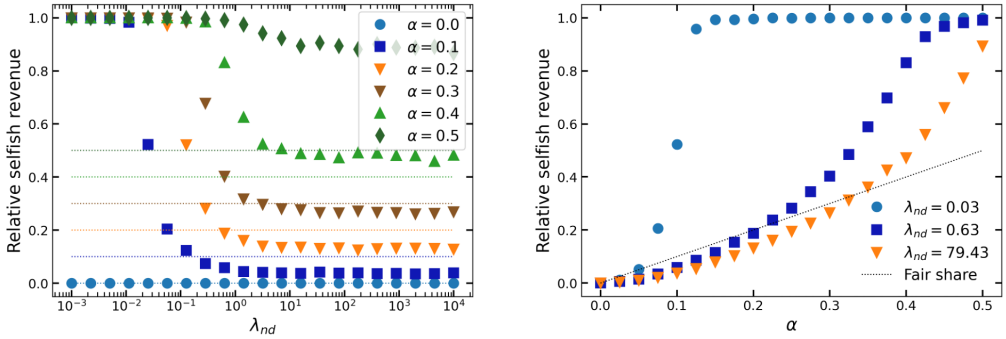
5.1.1. Network Delay

Note that we refer to high block latency, slow block propagation times or slow block diffusion times synonymously. When we refer to latency we mean block latency, i.e., the waiting time for a block to diffuse from one miner to a peer.

Figure 2 shows the impact of network latency, or block propagation time, on the relative reward of selfish mining with different hashing power levels. The control parameter λ_{nd} is the rate of block propagation. A high rate of block propagation implies short block diffusion times. In Figure 2a, selfish mining is a more profitable strategy, if the lines are above their respective "fair share" thresholds (i.e., their hashing power share), as indicated by the respectively coloured, dotted lines. For example, for $\alpha = 0.4$, we can see that relative selfish revenue never falls below 0.4, meaning that selfish mining is more profitable no matter how fast block propagation is.

There are two important observations to be made. First, relative selfish revenue, as defined in Section 4.2, is decreasing as the block propagation rate increases. Second, for a lower rate of block propagation (higher network delay), selfish mining is always a more profitable strategy than honest mining. The selfish miner benefits relatively more from slow block diffusion because they are more likely to extend their chain than honest miners. The reason is that the selfish miner does not suffer from slow block diffusion time, since they do not need to broadcast it in the first place. Notably, for higher values of α selfish mining is more robust against fast block diffusion. This is intuitive, as selfish mining becomes more profitable with increasing mining power.

Figure 2b shows for some values of λ_{nd} the relationship between the selfish miner's share of the network's hashing power, α, and relative selfish revenue, $R_{selfish}$. The values of λ_{nd} have been chosen to expose the dynamics of different latency regimes. We use these values for most following results. Importantly, the figure confirms some above findings, namely that $R_{selfish}$ is increasing in α. The faster block diffusion is, the more mining power is required to surpass the fair share threshold for the reasons stated above. Noteworthy is the fact that our model can confirm [2]'s main finding: selfish mining is always relatively more profitable for mining power levels exceeding one-third of the network's total mining power. In terms of latency, this means that no matter how high the block propagation rate is—and therefore unfavorable to selfish miners (see graph of $\lambda_{nd} = 79.43$)—selfish mining is always relatively more profitable for $\forall \alpha \gtrapprox \frac{1}{3}$.



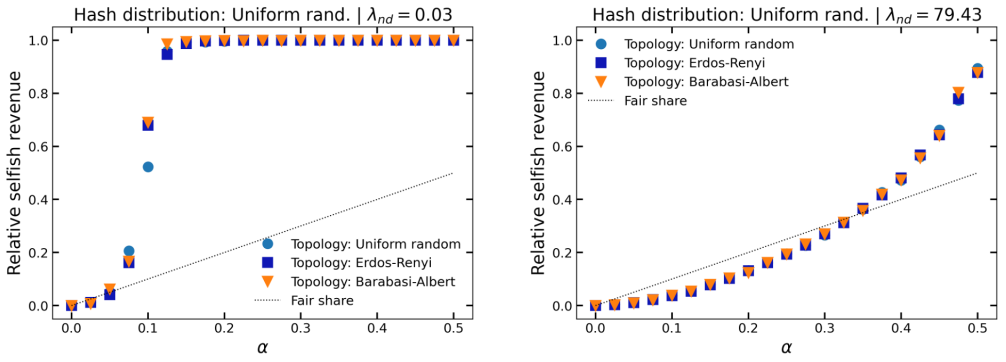
(**a**) (**b**)

Figure 2. Relative selfish revenue as a function of: (**a**) network latency (λ_{nd}) and (**b**) selfish hashing power (α).

5.1.2. Hashing Power Distribution

We are interested in learning the effect of varying hashing power distributions on relative selfish revenue, in order to understand the robustness of the selfish mining strategy. Figure 3 shows relative selfish revenue as a function of selfish mining power α under different hashing power distributions, namely a uniform random, power law, and exponential distribution. We show results for two different latency regimes.

We observe that hashing power following an exponential distribution has a negligible effect on the dynamics of the system, for either latency regime. The dynamics are indistinguishable from the previous results with a uniform random distribution. However, the left column of Figure 3 shows remarkable differences for hashing power following a power-law distribution with very high block latency (see Figure 3c where $\lambda_{nd} = 0.03$). First, relative selfish revenue crosses the fair share threshold for higher values of α. Second, the slope of the regression of relative selfish revenue on selfish mining power is notably less steep. The reason for this might be that hashing power following a power-law distribution implies that it is relatively more likely that an honest miner has a very large share of the network's hashing power. This in turn implies that when block diffusion is very slow, it is more likely that an honest miner manages to outperform the selfish miner, thus creating the longest chain. For this high latency, the main chain is effectively built by one single miner, because block diffusion time exceeds the expected time between new blocks. However, Gencer [19] finds that Bitcoin mining power trends can be fit as exponential distributions.



(**a**) (**b**)

Figure 3. *Cont.*

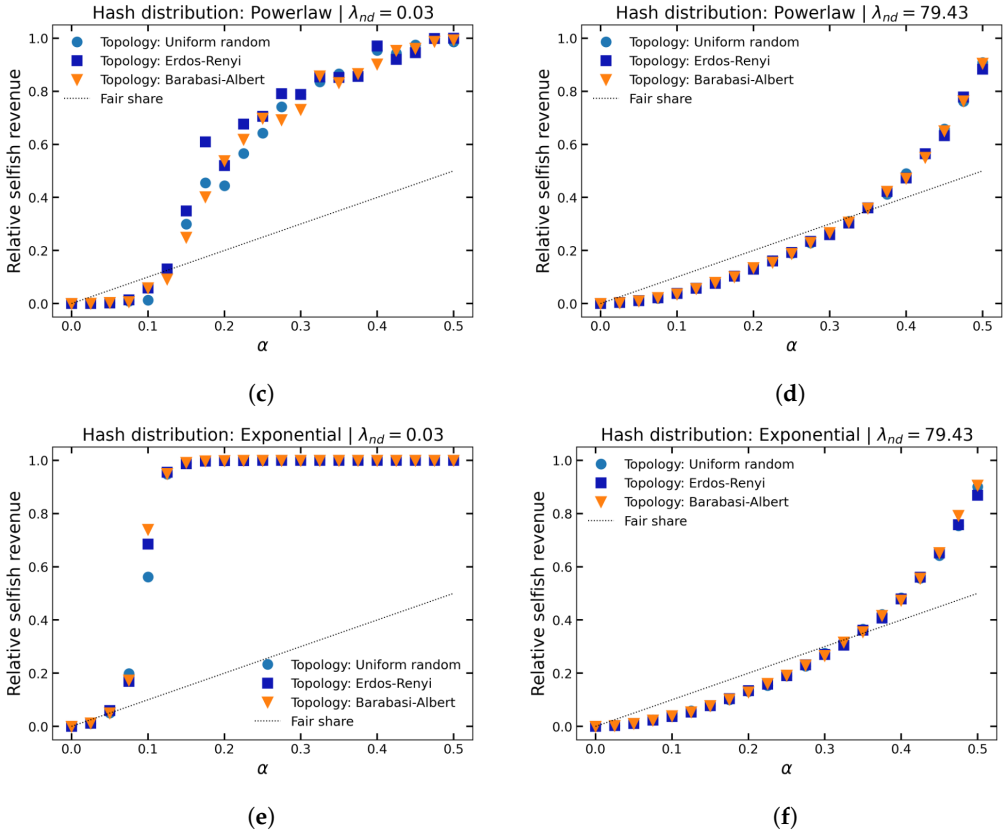


Figure 3. Relative selfish revenue as a function of selfish hashing power (α) under different distribution: (**a**,**b**) in which the hashing powers of all miners follow the uniform distribution; in (**c**,**d**) miners' hashing powers follow the power-law distribution; and in (**e**,**f**) miners' hashing powers follow the exponential distribution. In addition, the subfigures in left column show the results when network has very high latency $\lambda_{nd} = 0.03$, and those in right column show the results with low network latency $\lambda_{nd} = 79.43$.

5.1.3. Network Topologies

Next, we consider the effect of varying network topologies on relative selfish revenue. Figure 4 shows relative selfish revenue as a function of selfish mining power under uniform random, Erdos-Renyi and Barabasi-Albert topologies. Again, we show results for two latency regimes. It is apparent that the dynamics of the system are very robust to changes in topology for both latency regimes.

Overall, we can consolidate that the profitability of a selfish mining strategy is very robust to changes in the underlying network topology or hashing power distribution. The exception being powerful honest nodes in environments where block diffusion is very slow. In addition, the system in which hashing power follows a power-law distribution is a relatively unfavourable environment for a selfish miner.

Furthermore, to investigate the importance of centrality within the topology, the selfish miner is assigned to various positions in the network using various measures. In particular, we compare the performance difference between a random assignment, an assignment based on the standard betweenness centrality, and an assignment based on the new concept of *hashing betweenness* (introduced in Section 3.2).

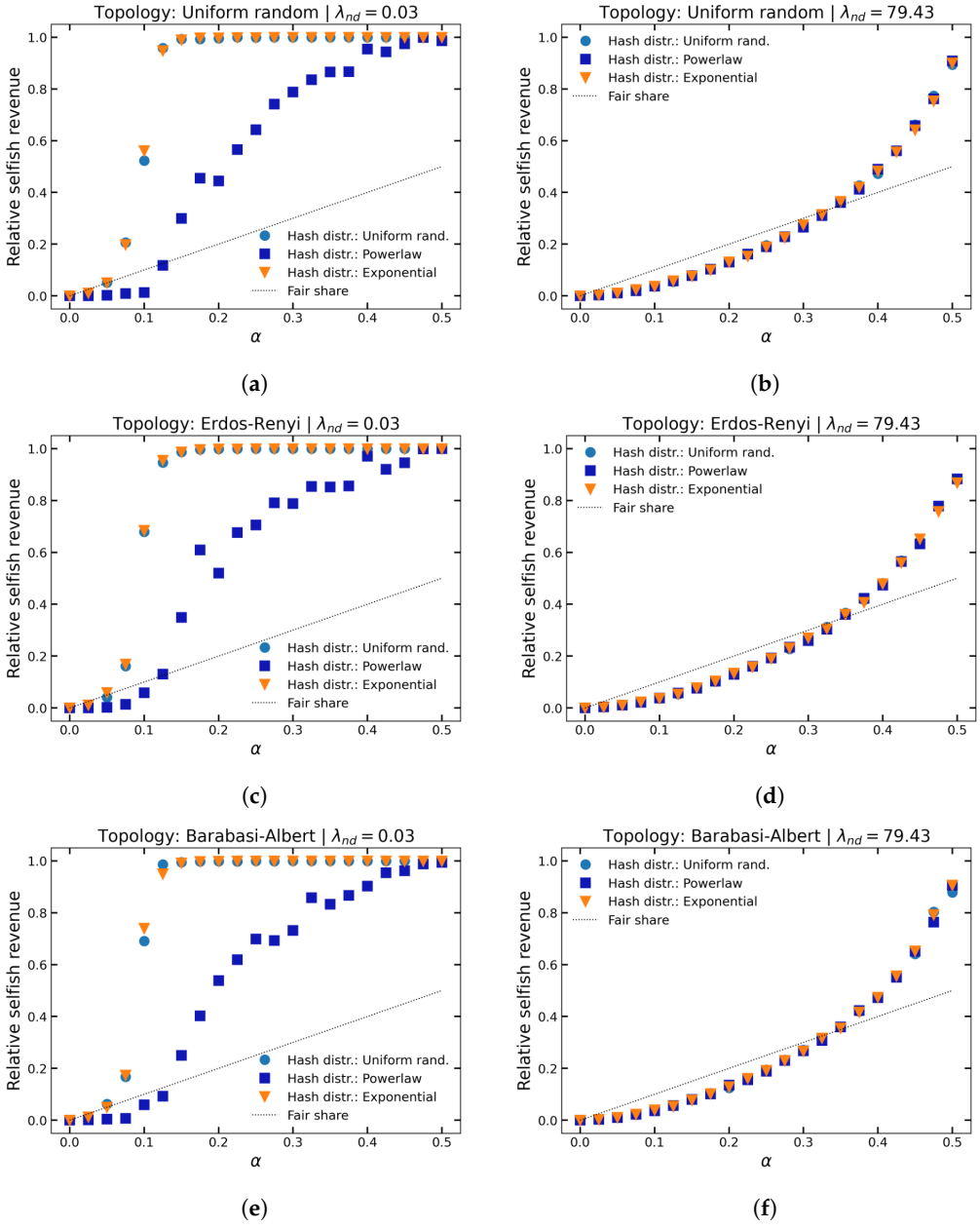
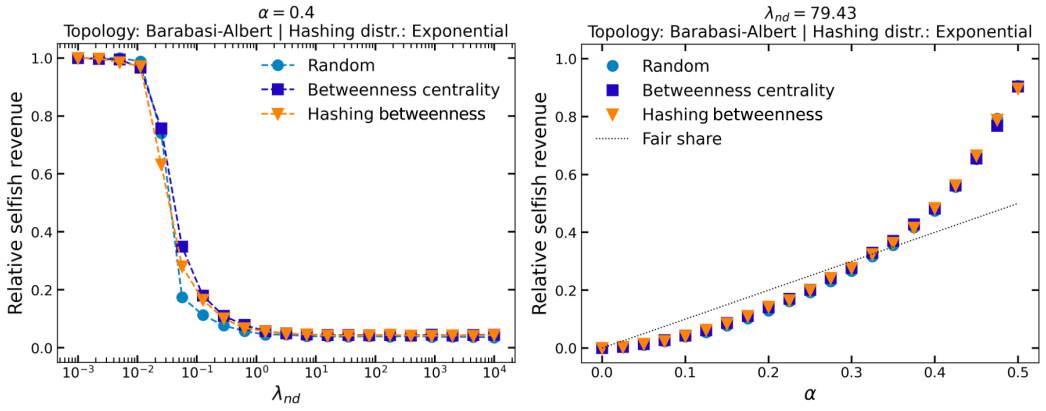


Figure 4. Relative selfish revenue as a function of selfish hashing power (α) for different network topologies: in (**a**,**b**) the network is in the uniform random topologies; in (**c**,**d**) the network is in the Erdos-Renyi topologies; and in (**e**,**f**) the network is in the Barabasi-Alber topologies. In addition, the subfigures in left column show the results when network has very high latency $\lambda_{nd} = 0.03$, and those in right column show the results with low network latency $\lambda_{nd} = 79.43$.

In Figure 5, for various centrality measures, we show the relative selfish revenue as a function of both latency (for $\alpha = 0.4$) and selfish hashing power (for $\lambda_{nd} = 79.43$) under the Barabasi-Albert topology and exponential hashing power distribution. Finally, we do not observe any notable effect of any centrality measure, despite choosing a heterogeneous network topology. The reason for this is likely to be the relatively small network size ($N = 100$), as well as the fact that the network is very well connected (nodes have an average degree of 10). Although it seems that the location of the selfish miner in the network has almost no effect on the profitability, the final conclusion still needs more experiments using more realistic network parameters. Therefore, future research could re-investigate the effect of centrality measures on a larger network.



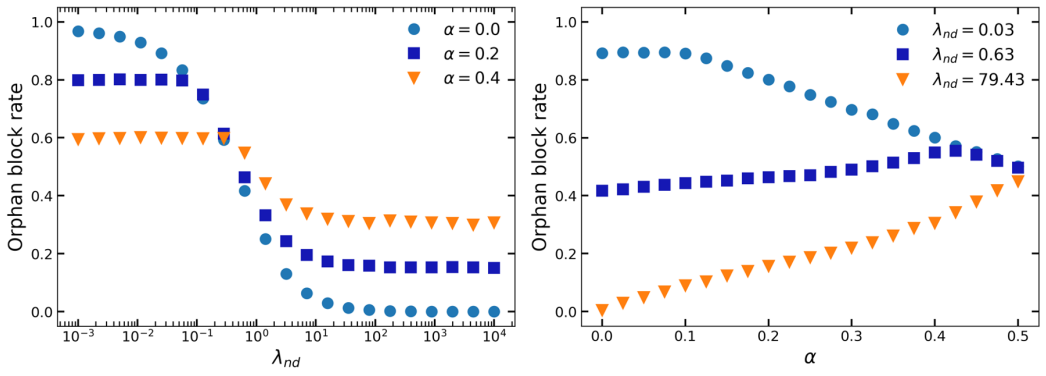
(**a**) (**b**)

Figure 5. Relative selfish revenue as a function of: (**a**) latency (λ_{nd}) and (**b**) selfish hashing power (α). Selfish miners are located according to various centrality measures.

5.2. Orphan Block Rate

Figure 6 shows the relationship between the orphan block rate and latency, for different levels of selfish mining power. Figure 6a shows that for slow block diffusion times there is a high rate of orphan blocks. This is in line with the intuition that slow block propagation causes many *accidental* chain forks. The reason behind this is the increased probability of a new block being found before it has fully spread through the network. Further, we can observe that for higher values of α there is less variation in the orphan block rate. For small λ_{nd}, the logic behind this observation is that selfish miners always build on their own blocks and with more mining power they are more likely to build on the main chain, effectively decreasing the number of orphan blocks.

In Figure 6b we can note two things. First, with no selfish mining present ($\alpha = 0$) we observe that for fast block diffusion times the orphan block rate is low. This is intuitive as with low latency accidental chain forks are less likely to occur. Second, as the selfish miner's share of the network's hashing power increases, the orphan block rate converges towards 0.5. This is due to the fact that for $\alpha = 0.5$ relative selfish revenue is converging towards 1. Once the selfish miner controls half of the network's hashing power, she can comfortably build the longest chain and therefore all honest blocks are rejected. Since honest miners control half of the network's mining power, this implies that half of all blocks are orphan blocks.



(**a**) (**b**)

Figure 6. Orphan block rate as a function of: (**a**) latency (λ_{nd}) and (**b**) selfish hashing power (α).

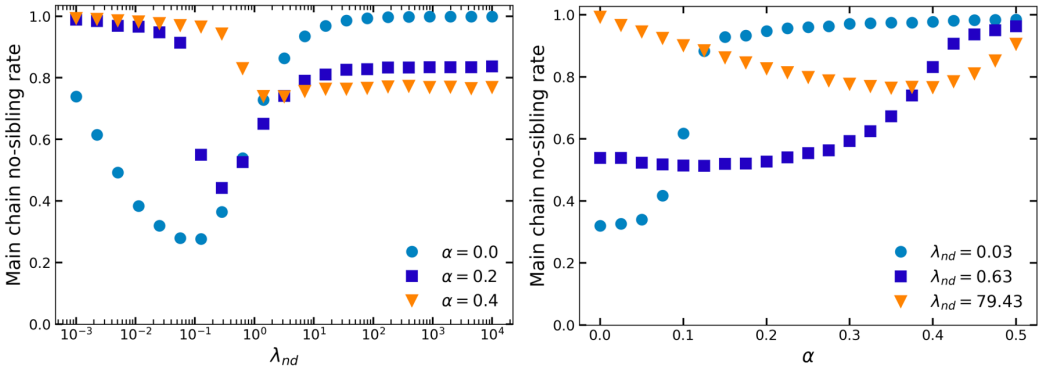
5.3. Consensus Block Rate

Selfish mining could cause a lot of orphan blocks, by intentionally splitting the chain. Most orphan blocks are honest blocks that are part of stale chains. However, most orphan

blocks do not split the main chain themselves, but are merely attached to it directly or indirectly via a chain of more orphan blocks. We are interested to reduce this "noise" of orphan blocks and only consider blocks that split the main chain. This will allow us to better understand the relationship between selfish mining and its impact on the blockchain forks. We call a block a *main chain no-sibling block* or a *consensus block*, if the main chain block does not share the same parent block with any other block. To compute the *main chain no-sibling rate*, we divide the number of main chain no-sibling blocks by the total number of main chain blocks.

The dynamics of the consensus block rate (as shown in Figure 7) are relatively complex. Considering Figure 7b: First, without selfish mining ($\alpha = 0$), the rate of consensus blocks converges to one as the block diffusion rate become very fast. Conversely, the sibling rate of main chain blocks is high for high latency. Second, when $\alpha = 0.5$ the selfish miner effectively controls the blockchain. Consequently, honest miners do not manage to build on top of the main chain as they continue to mine on stale chains. Therefore, main chain splits can barely take place and thus the sibling rate for main chain blocks is close to zero. Third, consider the graph for fast block diffusion, $\lambda_{nd} = 79.43$. Initially, the share of siblings grows as α increases, but beyond some inflexion point, the rate of siblings decreases again. This point of inflexion corresponds roughly to the threshold for which selfish mining becomes relatively more profitable than honest mining (for $\lambda_{nd} = 79.43$ this corresponds to $\alpha \approx \frac{1}{3}$ as per Figure 2b). This makes intuitive sense, because while selfish mining is not relatively more profitable, selfish miners continuously and intentionally split the main chain, without managing to build up an advantage of multiple blocks. Hence, there are many main chain splits, or few main chain no-sibling blocks. Once selfish mining becomes relatively more profitable, the intentional chain splits are outweighed by the relatively frequent advantages the selfish miners can now build up, thus leading to fewer main chain sibling blocks.

The results in Figure 7a are very similar. First, when there is no selfish mining, high latency implies more main chain splits. Before being equivalent to the block creation rate, the faster block diffusion will cause more sibling blocks. After exceeding the block creation rate by far, the faster block diffusion brings the no-sibling rate to one. On the other hand, when there is selfish mining, for very high latency, the selfish miner more effectively controls the main chain and so the rate of main chain siblings converges to zero. As latency decreases, selfish mining is less efficient in splitting the main chain, meaning that they tend to lose more block races against honest miners. Thus, the main chain no-sibling rate decreases until a point of inflexion. This point of inflexion is driven by the selfish mining strategy becoming relatively less profitable for higher λ_{nd}. For high λ_{nd} the rate of siblings decreases again as the dynamics are now dominated by honest miners, who observe fewer accidental chain splits. Intuitively, the points of inflexion show that the efficiency of selfish mining is more resistant to higher selfish mining power levels.



(**a**) (**b**)

Figure 7. Share of main chain blocks that have no siblings as a function of: (**a**) latency (λ_{nd}) and (**b**) selfish hashing power (α).

Overall Figure 7 illustrates the efficiency of the selfish mining behaviour. Here, efficiency refers to the effectiveness of intentional chain splits by the selfish miner. We learn that the dynamics are non-trivial and depend on the latency and mining power regime the selfish miner finds herself in. In short, the rate of main chain no-sibling blocks is driven by selfish mining when it is relatively profitable and is dominated by honest miners when selfish mining is relatively unprofitable.

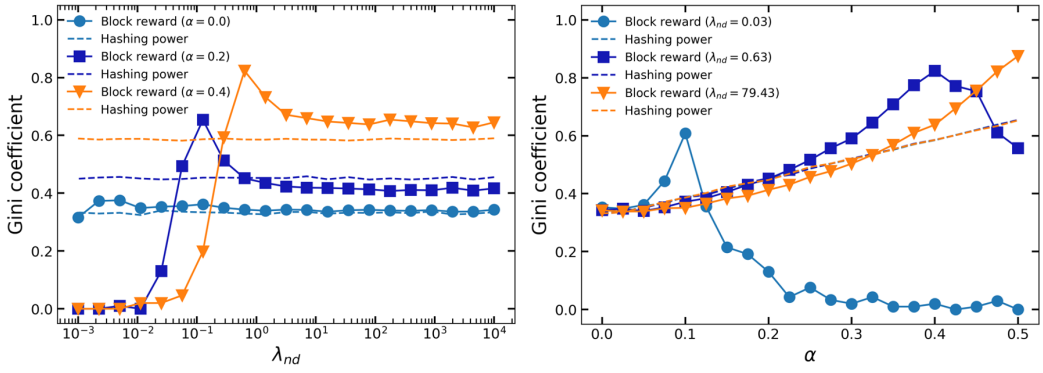
5.4. Gini Coefficients

As introduced in Section 4.1, we compute the Gini coefficient for both the distribution of block rewards and the hashing power distribution. If a selfish mining attack is successful, the block reward concentration should exceed that of the hashing power distribution. In a scenario in which selfish mining is more profitable, the selfish miner will earn block rewards in excess of her fair share, implying that block rewards get relatively more concentrated in the selfish miner. Thus, with profitable selfish mining, the Gini coefficient of block rewards should exceed the Gini coefficient of hashing power.

From Figure 2a we know that for $\alpha = 0.4$ selfish mining is always profitable, which is reflected in Figure 8a by the fact that block rewards are more unfairly distributed than the hashing power, and vice versa for $\alpha = 0.2$. As expected, the wealth inequality level remains relatively constant when there is no selfish mining activity ($\alpha = 0$). The reason why the Gini coefficients (for non-zero α values) are not "stabilised" yet for small λ_{nd} is due to a lack of variation in w_i as defined in Equation (6). For very slow block propagation rates (small λ_{nd}) the blockchain consists of a high share of orphan blocks, as we have seen in Section 5.2. Thus, the share of miners that have found a block finalised in the main chain is very low.

Similarly, Figure 8b shows that for reasonable block propagation rates (i.e., block diffusion time is less than the expected block creation time), the concentration of block rewards is growing more quickly than that of hashing power. The thresholds for when wealth concentration surpasses mining power concentration mimic the fair share threshold levels found in Figure 2b.

In summary, when selfish mining is relatively more profitable, the block reward concentration increases, and vice versa.



(**a**) (**b**)

Figure 8. Gini coefficients as a function of: (**a**) latency (λ_{nd}) and (**b**) selfish hashing power (α).

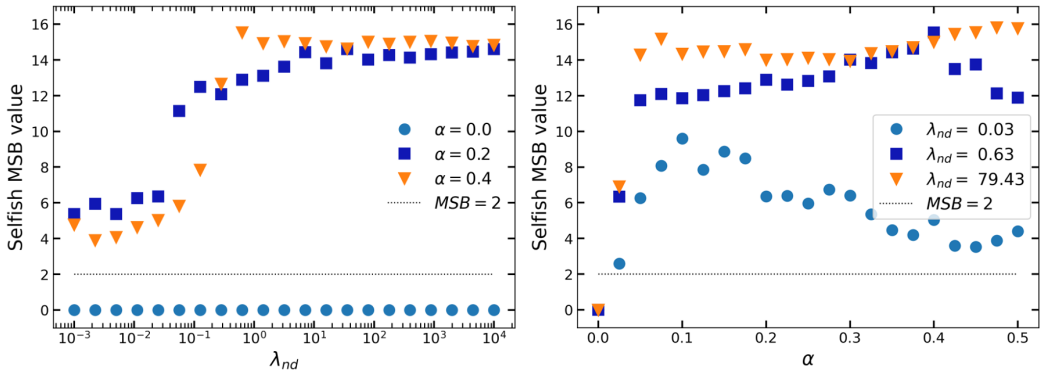
5.5. MSB Index

To validate the *Miner sequence bootstrapping model* (*MSB*) as proposed by [24] consider Figure 9.

Figure 9a shows that for all latencies selfish mining is detectable with the *MSB* methodology. Notably, it does not depend on selfish mining being a relatively more profitable strategy. This is due to the fact that selfish miners will, regardless of operating below the fair-share threshold, have an increased probability of publishing two blocks in a row.

Interestingly, the behaviour is relatively robust to changes in selfish mining power. Trivially, for $\alpha = 0$ there are no selfish blocks and thus $MSB = 0$.

Figure 9b confirms the findings above. However, for $\lambda_{nd} = 0.03$, after some inflection point, the MSB value is decreasing in α. This is surprising. But note that these dynamics occur in a latency regime that can be considered extreme (implied block delay time of more than 30 min).



(**a**) (**b**)

Figure 9. MSB index as a function of (**a**) latency (λ_{nd}) and (**b**) selfish hashing power (α).

In our ABM, we can assign each block to a specific miner, such that [25]'s assumption of miners having only one identity in the network is fulfilled. We are not modelling the fact that miners may choose to hide selfish mining behaviour by frequently changing their addresses. Further, our results validate that the detection approach using MSB index [24,25], can produce results as expected, namely that selfish mining causes statistically detectable levels of consecutive blocks by the same miner.

6. Selfish Mining Pool

All the previous simulations and results are based on the assumption that there is only a single selfish miner in the network. However, in reality, miners often cooperate in mining pools to share the larger computational power, and to ensure a more stable reward payout. Therefore, we want to further consider the situation in which there is a selfish mining pool.

We assume there is a selfish mining pool in which members share the information with their pool peers (i.e., each selfish node is connected to all the other selfish nodes in a peer-to-peer (p2p) network). That means, if one selfish mining pool peer finds a block, it is shared within the selfish pool, but not necessarily with honest nodes. As a result, selfish nodes will mine on top of the latest selfish block, but honest nodes will not unless it is shared with them (when selfish miners need to win a block race against honest miners). They do not act as individual selfish nodes, but they operate like a mining pool (in practice they would share the profits proportionally too, which is not relevant for the model as we only care about the aggregate performance). For the simulation, we randomly assign each selfish node a hashing power and normalise the values so that the sum of all selfish miners' hash rates equals α.

Figure 10 shows mainly two results for multiple selfish miners operating within a pool. First, the relationship between the share of main chain blocks and the share of hashing power is very linear within types of miners. This implies that within honest miners, rewards are distributed fairly, as well as within the selfish mining pool. Here fairness refers to fairness *within* groups of miners that are of the same type (selfish or honest), not across types. Trivially, selfish mining causes a bias in block reward distribution across types of miners. Second, latency can cause different dynamics, if the selfish share of the network's total hashing power (α) is shared among multiple miners. Figure 10c shows that even for $\alpha = 0.4$ selfish mining is relatively less profitable for high latency ($\lambda_{nd} = 0.63$). This is due to the fact that we assume block latency within the selfish mining pool to be the same as

that outside of the pool. If the selfish mining power is shared across many miners, the pool suffers from high block latency, because by the time selfish nodes find out about blocks from selfish peers, the honest nodes are likely to have found a block themselves already. Finally, for $\alpha = 0.4$ and a relatively fast block propagation rate ($\lambda_{nd} = 7.14$) selfish mining is relatively more profitable than honest mining.

We have shown that there is no bias in the distribution of block rewards within miner types. Further, we have shown a glimpse into the dynamics of the system with a selfish mining pool. However, it is worth noting that we currently assume that the communication among selfish and honest miners follows the same distribution. In reality, it would be sensible for selfish nodes to set up very fast communication channels with their selfish mining pool peers which are not necessarily using the default p2p layer. Therefore, a variation in block propagation speeds for communication between pool members and non-members remains an important dynamic to analyze.

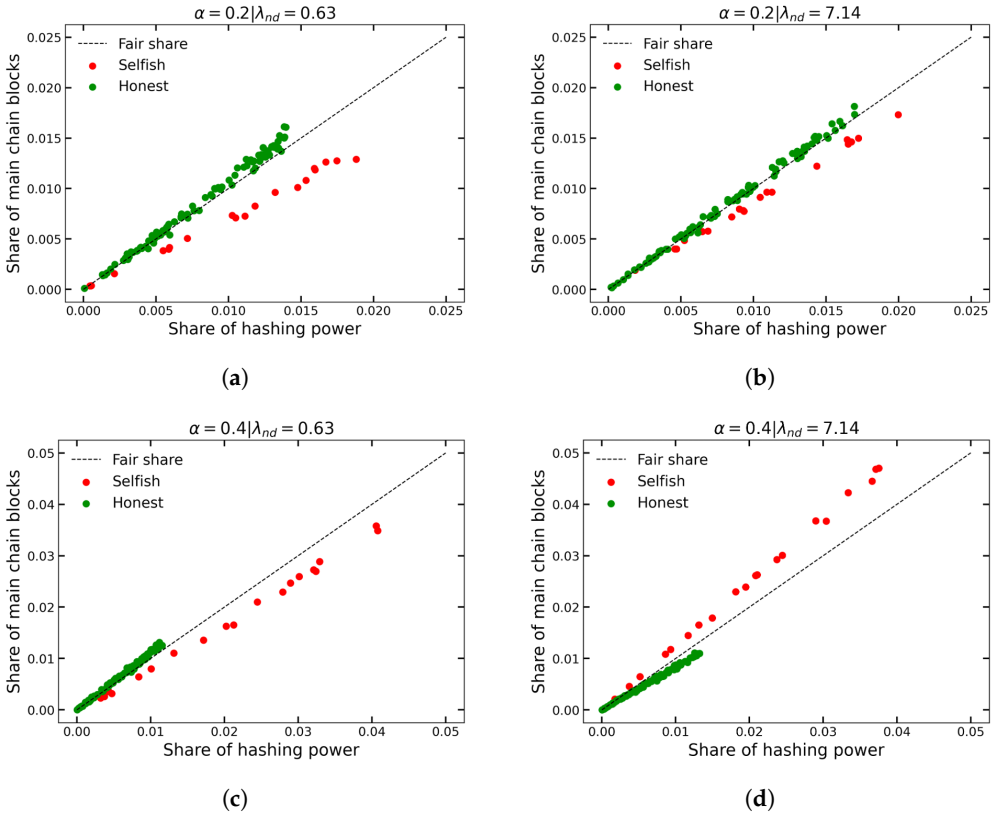


Figure 10. Single realisations for a selfish mining pool: Share of main chain blocks as a function of the share of hashing power for all nodes. In subfigures (**a**,**b**), all selfish miners in the mining pool control 20% hashing power, and in (**c**,**d**) those selfish miners control 40% hashing power. In addition, the subfigures in left column show the results when network has high latency $\lambda_{nd} = 0.63$, and subfigures in right column show the results with low network latency $\lambda_{nd} = 7.14$.

7. Discussion and Conclusions

In this paper, we introduced an agent-based model that simulates the selfish mining strategy in a Proof-of-Work protocol setting. The modelling of a peer-to-peer network structure allows us to extensively study the effects of block propagation delays, network topology and mining power distribution on relative selfish revenue. After simulating the selfish miner's behaviour, we further analyze the influence of selfish mining on network efficiency and reward fairness using orphan block rate and Gini index respectively.Our research suggests that selfish mining is a viable and profitable strategy.

Assuming a single selfish miner, we show that relative selfish revenue is a decreasing function of block propagation speed. Further, we find that for high levels of latency selfish

mining is always a relatively more profitable strategy. In particular, we confirm [2]'s main observation that selfish mining is always relatively more profitable for hashing power share being more than 1/3, even when considering network dynamics such as latency. Furthermore, we show that the dynamics of selfish mining are very robust to changes in the underlying network topology and the centrality of the selfish miner's location. However, we find that a hashing power distribution following a power law greatly affects relative selfish revenue. In other words, powerful honest nodes can make it harder for selfish miners to be relatively more profitable. Regarding the global effect, we find that when selfish mining is relatively more profitable, the block reward concentration also increases. Besides the intuition that slower block diffusion would cause a high rate of orphan blocks, we also reveal that with the selfish miner's share of the network's hashing power increasing to 50%, the orphan block rate also converges towards 50% and the main chain no-sibling rate converges to 100%. Finally, using Miner Sequence Bootstrapping (MSB) index [24,25], we confirm selfish mining behaviour also causes statistically significant levels of consecutive blocks by the same miner, which is beneficial in detecting selfish miners in real-world systems such as Bitcoin.

The composability of our ABM has the advantage of allowing modifications to the behaviour of agents easily. In addition, the model is flexible to simulate more selfish miners too. We extend the model to analyse the dynamic when multiple selfish miners cooperate within a pool. For the simplification, we currently assume that the communication within the mining pool and outside the pool follows the same rate. However, in reality, it is more likely that pool members use more efficient means of communication. For example, rather than propagating entire blocks only sending block headers might decrease latency. Thus, an improvement to the model would consider faster in-pool communication than the average information propagation time among the network. Another limitation is that some of our simulations were limited by the relatively small network size, which may cause a lack of heterogeneity in the network [15]. An extension of this work will include larger networks as well as non-mining full nodes as part of the peer-to-peer network. Further, the effect of the selfish miner's network location, as well as their connectivity, will be considered. For example, we are planning to propose a new concept of *hashing centrality* in the peer-to-peer mining network, to study the effect of selfish miners being connected to "powerful" honest nodes.

Last but not least, our paper shows that a combination of game-theoretic analysis paired with agent-based modelling can help to advance the field by giving insights into what effects strategic behaviour has on the global properties of the system. Agent-based modelling proves to be a valuable methodology, where analytical derivations or direct implementation in large-scale distributed networks fail to be viable. ABMs allow us to bridge the gap between theory and empirical complex systems.

Author Contributions: Conceptualisation, C.J.T., C.S.-S. and S.-N.L.; methodology, C.J.T. and C.S.-S.; formal analysis, C.S.-S.; writing—original draft preparation, S.-N.L.; writing—review and editing, C.J.T., C.S.-S. and S.-N.L.; supervision, C.J.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partly funded by the China Scholarship Council (CSC) (No. 201808310212).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Decentralized Business Review. 2008; p. 21260. Available online: https://www.debr.io/article/21260-bitcoin-a-peer-to-peer-electronic-cash-system (accessed on 1 May 2022).
2. Eyal, I.; Sirer, E.G. Majority is not enough: Bitcoin mining is vulnerable. In Proceedings of the International Conference on Financial Cryptography and Data Security, Christ Church, Barbados, 3–7 March 2014 ; Springer: Berlin/Heidelberg, Germany, 2014; pp. 436–454.
3. Sapirshtein, A.; Sompolinsky, Y.; Zohar, A. Optimal selfish mining strategies in bitcoin. In Proceedings of the International Conference on Financial Cryptography and Data Security, Christ Church, Barbados, 22–26 February 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 515–532.
4. Lewenberg, Y.; Bachrach, Y.; Sompolinsky, Y.; Zohar, A.; Rosenschein, J.S. Bitcoin mining pools: A cooperative game theoretic analysis. In Proceedings of the Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, Istanbul, Turkey, 4–8 May 2015; pp. 919–927.
5. Tasca, P.; Tessone, C.J. A Taxonomy of Blockchain Technologies: Principles of Identification and Classification. *Ledger* **2019**, *4*. [CrossRef]
6. Wright, C.S.; Savanah, S. The Fallacy of the Selfish Miner in Bitcoin: An Economic Critique. 2017. Available online: https://ssrn.com/abstract=3009466 (accessed on 1 May 2022).
7. Schwarz-Schilling, C.; Li, S.N.; Tessone, C.J. Agent-based Modelling of Strategic behavior in PoW Protocols. In Proceedings of the 2021 Third International Conference on Blockchain Computing and Applications (BCCA), Tartu, Estonia, 15–17 November 2021; pp. 111–118.
8. Bag, S.; Ruj, S.; Sakurai, K. Bitcoin block withholding attack: Analysis and mitigation. *IEEE Trans. Inf. Forensics Secur.* **2016**, *12*, 1967–1978. [CrossRef]
9. Nayak, K.; Kumar, S.; Miller, A.; Shi, E. Stubborn mining: Generalizing selfish mining and combining with an eclipse attack. In Proceedings of the 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany, 21–24 March 2016; pp. 305–320.
10. Liu, H.; Ruan, N.; Du, R.; Jia, W. On the Strategy and Behavior of Bitcoin Mining with N-attackers. In Proceedings of the 2018 on Asia Conference on Computer and Communications Security, Incheon, Korea, 4–8 June 2018; ACM: New York, NY, USA, 2018; pp. 357–368.
11. Solat, S.; Potop-Butucaru, M. Brief announcement: Zeroblock: Timestamp-free prevention of block-withholding attack in bitcoin. In Proceedings of the International Symposium on Stabilization, Safety, and Security of Distributed Systems, Boston, MA, USA, 5–8 November 2017; Springer: Cham, Switzerland, 2017; pp. 356–360.
12. Zhang, R.; Preneel, B. Publish or perish: A backward-compatible defense against selfish mining in bitcoin. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 14–17 February 2017; Springer: Cham, Switzerland, 2017; pp. 277–292.
13. Nicolas, K.; Wang, Y.; Giakos, G.C.; Wei, B.; Shen, H. Blockchain system defensive overview for double-spend and selfish mining attacks: A systematic approach. *IEEE Access* **2020**, *9*, 3838–3857. [CrossRef]
14. Tessone, C.; Tasca, P.; Iannelli, F. Stochastic modelling of blockchain consensus. *arXiv* **2021**, arXiv:2106.06465.
15. Geier, M.; Tessone, C.J.; Vanotti, M.; Vileriño, S.; Márquez, D.G.; Mocskos, E. Using network emulation to study blockchain distributed systems: The ethereum case. In Proceedings of the 2019 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), Pavia, Italy, 13–15 February 2019; pp. 51–58.
16. Decker, C.; Wattenhofer, R. Information propagation in the bitcoin network. In Proceedings of the IEEE P2P 2013 Proceedings, Trento, Italy, 9–11 September 2013; pp. 1–10.
17. Chen, H.; Jin, H.; Sun, J.; Deng, D.; Liao, X. Analysis of large-scale topological properties for peer-to-peer networks. In Proceedings of the IEEE International Symposium on Cluster Computing and the Grid, CCGrid 2004, Chicago, IL, USA, 19–22 April 2004; pp. 27–34.
18. Shahsavari, Y.; Zhang, K.; Talhi, C. Toward Quantifying Decentralization of Blockchain Networks With Relay Nodes. *Front. Blockchain* **2022**, *5*, 812957. [CrossRef]
19. Gencer, A.E.; Basu, S.; Eyal, I.; Van Renesse, R.; Sirer, E.G. Decentralization in bitcoin and ethereum networks. In Proceedings of the International Conference on Financial Cryptography and Data Security, Nieuwpoort, Curaçao, 26 February–2 March 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 439–457.
20. Gillespie, D.T. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.* **1976**, *22*, 403–434. [CrossRef]
21. Gillespie, D.T. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* **1977**, *81*, 2340–2361. [CrossRef]
22. Dalton, H. The measurement of the inequality of incomes. *Econ. J.* **1920**, *30*, 348–361. [CrossRef]
23. Morgan, J. The anatomy of income distribution. *Rev. Econ. Stat.* **1962**, *44*, 270–283. [CrossRef]
24. Li, S.N.; Yang, Z.; Tessone, C.J. Mining blocks in a row: A statistical study of fairness in Bitcoin mining. In Proceedings of the 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Toronto, ON, Canada, 2–6 May 2020; pp. 1–4.
25. Li, S.N.; Yang, Z.; Tessone, C.J. Proof-of-Work cryptocurrency mining: A statistical approach to fairness. In Proceedings of the 2020 IEEE/CIC International Conference on Communications in China (ICCC Workshops), Chongqing, China, 9–11 August 2020; pp. 156–161.