Agent-based Modelling of Strategic behavior in PoW Protocols

1st Caspar Schwarz-Schilling
Department of Business Administration
University of Zurich
CH-8050 Zurich, Switzerland
caspar.schwarz-schilling@uzh.ch

2nd Sheng-Nan Li *
URPP Social Networks
University of Zurich
CH-8050 Zurich, Switzerland
shengnan.li@uzh.ch

3rd Claudio J. Tessone
UZH Blockchain Center, URPP Social Networks
University of Zurich
CH-8050 Zurich, Switzerland
claudio.tessone@uzh.ch

2021 Third International Conference on Blockchain Computing and Applications (BCCA) | 978-1-6654-2182-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/BCCA53669.2021.9657011

***Abstract*—In blockchain-based systems, such as Bitcoin's Proof-of-Work (PoW) protocol, it is expected that a miner's share of total block revenue is proportional to their share of the network's total hashing power. However, a deviation to this behavior is the *selfish mining* strategy, an attack vector discovered by Eyal and Sirer in 2014. This strategy may lead to a miner earning more than their "fair share". As a result, Bitcoin's security assumption of an honest majority may not be sufficient. In this paper, in order to verify whether selfish mining is indeed a profitable strategy in PoW systems, we introduce an agent-based model to simulate the dynamics of selfish mining behavior. The model is by design minimalistic allowing us to analyze the effect of network latency, hashing power distribution, and network topology on relative revenue of selfish miners. We find that for high levels of latency, selfish mining is always a relatively more profitable strategy, and the results turn out to be very robust to changes in the network topology. In addition, we find that the hashing power distribution following power laws, as found empirically, can make it harder for selfish miners to be profitable. Our analysis confirms the main observations that selfish mining is always relatively more profitable for hashing powers representing more than one third of the total computing power. Further, it also confirms that selfish mining behavior could cause a statistically significant high probability of contiguously mined blocks.**

***Index Terms*—Blockchain, Proof of Work, Selfish mining, Agent-based Model, Complex systems modelling**

I. Introduction

In 2014, Eyal and Sirer [1] 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*. In short, selfish mining aims to make *honest miners* waste their computational resources on blocks that are destined to not be part of the main chain, i.e. get no block reward. The *selfish miner* may achieve this by only releasing newly mined blocks strategically, as opposed to naively always broadcasting them immediately. 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.

* is corresponding author

Given the far reaching implications of this attack vector, it is worth investigating the dynamics of selfish mining within a network with block propagation delay. To achieve this, we introduce an agent-based model to study the selfish mining attack. Our main research questions are:

- RQ1: What is the effect of network latency on selfish mining?
- RQ2: What is the effect of network topology on selfish mining?
- RQ3: What is the effect of heterogeneous hashing power distributions on selfish mining?
- RQ4: Which observable characteristics does selfish mining produce in the sequence of blocks appended to the main chain?

The paper is organized as follows: Section II advances the concept of selfish mining and provides a literature review; Section III introduces the modelling approach and Section IV the results and limitations. Finally, Section V draws conclusions and poses venues for future research.

II. Related Literature

Eyal and Sirer [1] first introduced the attack vector *selfish mining*, also sometimes referred to as *block withholding* [2]. They show that under certain conditions a deviation from the standard Bitcoin protocol may be a more profitable strategy for a miner.

Many extensions to selfish mining strategies have been proposed, such as stubborn mining and publish-n strategy [3]–[5]. As a response, approaches to defend honest actors against selfish mining have been proposed. They can be categorized into two approaches: 1) making fundamental changes to the block validity rules, for example *ZeroBlock* [6]: 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* [7]: Here miners compare the weight of the chains instead of their length. In addition, it is debated whether the selfish mining strategy is actually profitable in practice: Some studies indicate that selfish mining may let

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.

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 [8].

Li *et al.* in 2020 [9], [10] proposed a method for detecting selfish mining behavior 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. They find evidence of selfish mining in Proof-of-Work protocols (mainly those with smaller number of nodes).

In 2021, Tessone *et al.* [11] introduced a minimalistic stochastic model of consensus in blockchain-based systems, with Proof-of-Work as a primary example. The authors show use as control parameter the ration between network delay and interblock time, showing a rapid decrease in the fraction of time the system is in consensus when the network delay decreases. The authors 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 [12] who deployed a synthetic Ethereum network with pre-defined, adjustable network delays.

III. Selfish Mining

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* [13]. On the contrary, 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]. 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. More likely, however, is that a selfish miner is not operating alone, but is part of a selfish mining *pool*. Generally, a pool is a set of miners that cooperate, in order to smooth their returns over time. A mining pool pools computational resources of all members. The block rewards are shared between all mining pool members, proportional to their hashing power within the pool [14]. If a selfish miner operates within a selfish mining pool, private blocks are shared with all pool members, in order to extend the pool's private branch. For the following examples we will, without loss of generality, assume a single selfish miner.

We use [1]'s definition of α as the selfish miner's (or selfish pool) share of the network's total hashing power. To illustrate selfish mining, consider block tree (b) of Fig. 1, in which a selfish miner, say Alice, controls 40% of the network's hashing power ($\alpha = 0.2$) and the honest miners collectively control 80% ($(1 - \alpha) = 0.8$). 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 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.

This potential advantage of selfish mining comes at the risk of losing block revenue for not publishing the block immediately. Consider scenario (a) in Fig. 1, in which a selfish miner, say Alice, controls 20% of the network's hashing power ($\alpha = 0.4$) and the honest miners collectively control 60% ($(1 - \alpha) = 0.6$). 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.

Note that the fair share of revenue for the selfish miner in scenario (a) would have been 0.2, but the realized 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 realized share of revenue for the selfish miner was 0.5. These two scenarios illustrate that selfish mining may not always be a profitable strategy [8]. The main determinant for profitability is the selfish miner's share of the network's hashing power.

In order to investigate whether selfish mining is a profitable strategy when considering network latency, we introduce an agent-based model to simulate the miner's behavior and blockchain dynamics in the next section IV.

IV. Modelling approach

In the first part of section IV-A, we briefly discuss our agent-based model (ABM) in general. Afterwards, we define an algorithm that follows the logic of the selfish mining specification. Finally, we introduce a metric to capture the efficiency of the selfish mining strategy.

A. ABM of Selfish Mining

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

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.

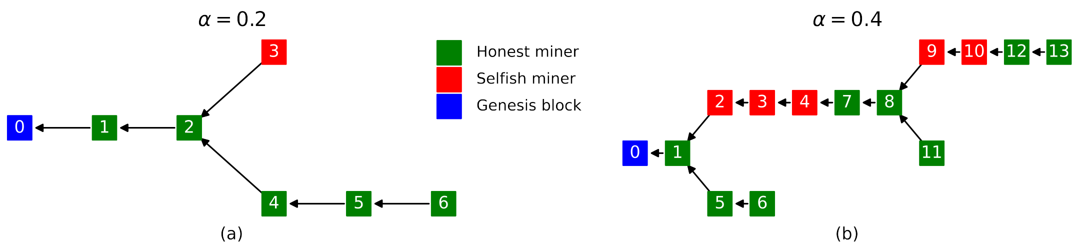


Fig. 1: Visualisation of block trees for different selfish shares of the network's hashing power (α)

network, which we will henceforth refer to as Bitcoin. Nodes connect are endowed with computational power which they devote to solve the Proof-of-Work problem. If this computational power is zero, the nodes only participate of the consensus but cannot create blocks; all others work to *mine* new blocks. Without loss of generality, we call all nodes *miners*. A miner can be of either honest or selfish type. The former will follow the standard Bitcoin protocol, the latter will perform the selfish mining strategy. For sake of simplicity, in the simulations, we assume a single selfish miner, which implies that $(N-1)$ miners are honest.

Miners are connected, so that they can communicate with one another. For the simulations we assume an average degree of 10. 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 [15]. To account for heterogeneity in the degree distribution, we consider different network topologies in the simulations (uniform random, Erdos-Rényi, Barabási-Albert).

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. 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 believe this is a reasonable assumption to make.

2) Mining: Each miner is assigned some computational power share, c_i, such that $\sum_{i=1}^{N} c_i = 1$. Further, we assume that the selfish mining pool 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 [16]. To reflect this heterogeneity of computational resources across miners, we consider various hashing power distributions in the simulations (uniform random, power-law, exponential).

Blocks are mined continuously and independently at a constant average rate. Thus, the proof-of-work algorithm that determines block creation is a Poisson process. Therefore, time between events – here events are new blocks – follows the exponential distribution. In Bitcoin, the average inter-block time is defined as 10 minutes, which we denote as $\tau_b = 10'$. Given that we assume a static topology, 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 orphaned blocks. However, we assume no such difficulty adjustment. Again, we justify this simplification with the fact that the defining time scale for selfish mining is much lower. But indeed, this may be accounted for in future research.

The probability that miner i finds a new block in a given small time interval dt is

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

Note that since computational power is normalized, 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. Equation 1 implies that miner i's time to find a new block follows the exponential distribution with parameter η_i. Further, it follows that $\sum_{i=1}^{N} \eta_i = 1/\tau_b$. Given that the block creation process is a Poisson process, the number of blocks in a given interval follow a Poisson distribution with parameter η_i. Hence, the number of blocks mined by all miners also follows a Poisson distribution, but with parameters $\sum_i \eta_i$.

3) Blockchain dynamics: In our model, the atomic unit of information propagated through the network is a block b, which is mined by a miner m_b. Transactions are not considered specifically, as we are only interested in the process of block creation, namely mining. As introduced in [17], each block is of a certain height h_b (number of blocks directly chained together between block b and the genesis block). Further, each miner has at any point in time t a local copy of the blockchain $B_i(t)$. At time t, the height of miner i's local copy of a blockchain is defined as

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

Importantly, a miner deems the blockchain valid that has cumulatively spent most proof-of-work, which in the ABM

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.

always corresponds to the longest chain.

4) Network delay: For the sake of simplicity, it is assumed that block propagation process 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 $\sim \exp(-\lambda_{nd})$. Indeed, we assume that the communication between selfish and honest nodes and within the selfish pool follows the same distribution. In reality, it is more likely that selfish pool members use more efficient means of communication. For example, rather than propagating entire blocks they could send only so-called block headers to decrease latency. As a consequence, the results for the profitability threshold level of selfish miners will tend to represent an upper-bound. Future research could consider to differentiate block propagation speeds between pool members and non-members.

5) Evolution: The ABM evolves over time as events occur. There are two types of events that can occur: Either a new block is mined by some node or a block is propagated by some node to some other node. Event times are computed very efficiently with the *Gillespie algorithm* [18]. "It [...] numerically simulates the time evolution of the given [...] system. [...] This algorithm never approximates infinitesimal time increments dt by finite time steps Δt" [19].

Each node i independently mines a new block at a Poisson rate η_i. Thus, at the global level new blocks are created at a rate $\sum_i \eta_i$. Blocks, on the other hand, are propagated from node i to her peers at a rate λ_{nd}. Importantly, blocks are only propagated according to rules specified in subsubsection IV-A3. 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 to. The latter we define as the number of *active links* N_{al}, again following [11]'s notation. It follows that the rate of block propagation between two nodes connected by an active link occurs at an aggregate rate of $N_{al}\lambda_{nd}$.

Having defined the rates of all event types, it follows that the rate at which any of these events occurs is defined as $\xi = \sum_i \eta_i + N_{al}\lambda_{nd}$. Effectively, the Gillespie algorithm computes a waiting time such that the event occurrences in the system follow the distributions outlined in subsubsection IV-A2 and subsubsection IV-A3. To compute the waiting for the next event, the next event is selected with probability proportional to its rate of occurrence in the system. Thus,

$$\frac{N_{al}\lambda_{nd}}{\sum_i \eta_i + N_{al}\lambda_{nd}} \tag{3}$$

is the probability that the next event is a broadcasting event. If the next event is a broadcasting 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,

$$\frac{\sum_i \eta_i}{\sum_i \eta_i + N_{al}\lambda_{nd}} \tag{4}$$

is the probability that the next event is a mining event. If the next event is a mining event, then the probability with which a specific node is randomly 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.

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

B. Types of miners

a) Honest miner: Consider honest miner i and any other miner j (selfish or honest). If miner j at time t broadcasts block b to miner i, miner i will only accept block b and add it to her local copy of the blockchain if i's height of the local copy of the blockchain is less than the height of the block just received by miner j, namely if $H_i(t) < h_b$. If this block b is accepted, miner i also accepts all preceding blocks (only relevant if block b was mined on top of a different block than miner i was mining on before). At time t miner j and i effectively share the same blockchain copy. As a result $H_i(t)$ increases to match block b's height h_b. Similarly, if miner i mined a block at time t (instead of receiving a new block), this would also lead to an increase of $H_i(t)$.

Importantly, honest miners naively share blocks that they either just accepted or mined themselves with all of her peers immediately.

b) Selfish miner: On the contrary, a selfish miner does not necessarily share a block with honest miners immediately. As a consequence, members of the selfish mining pool intentionally create a chain split to mine on their private chain. 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 define an algorithm that is executed by selfish miners with the aid of *pseudo-code*. Note that algorithm 1 follows the logic of Eyal's selfish mining specification in [1].

C. 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 $1, ..., S$ be selfish and miners $(S + 1), ..., N$ be honest. With w_i denoting miner i's revenue, relative selfish revenue is defined as

$$R_{selfish} = \frac{\sum_{i=1}^{S} w_i}{\sum_{i=1}^{S} w_i + \sum_{i=(S+1)}^{N} w_i} \tag{5}$$

D. Miner Sequence Bootstrapping Index

We now test the method of [9], whose central idea is to exploit the increased probability for selfish miners to publish contiguous blocks. To validate this methodology, we implement their *Miner Sequence Bootstrapping model* (MSB). First, we count the number of times that in period T miner i continuously discovers two blocks, C_i^T. Next, we shuffle the

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Selfish-Mining

```
On INIT
    current block ← genesis block    ▷ Block to mine on
    current height ← 0    ▷ Height of current block
    public height ← 0 ▷ Maximum height of public chain
    private chain length ← 0    ▷ Number of blocks attached to private chain
    mine on current block    ▷ Always mine on current block

On RECEIVE BLOCK(block)
    $\Delta \leftarrow$ (current height $-$ public height)
    if block mined by honest miner then
        if height of block $>$ public height then    ▷ New honest block, update local view
            public height ← height of block
            if height of block $>$ current height then    ▷ Honest chain ahead, adopt it
                current block ← block
                current height ← height of block
                broadcast block to all miners
                private chain length ← 0
            else if $\Delta = 1$ then    ▷ Lead was 1, publish private block to start race
                broadcast current block to honest miners
            else if $\Delta = 2$ then    ▷ Lead was 2, publish private chain
                broadcast current block to honest miners
            else if $\Delta > 2$ then    ▷ Lead was $> 2$, publish matching private block
                broadcast private block to match public height
    else
        if height of block $>$ current height then    ▷ New selfish block, update local view
            current block ← block
            current height ← height of block
            private chain length $+1$
            if $\Delta = 0$ and private chain length $= 2$ then    ▷ Was 1-1 race, publish new block
                broadcast block to all miners
                private chain length ← 0
            else    ▷ Extend private chain length
                broadcast block to selfish miners

On MINE BLOCK(block)
    $\Delta \leftarrow$ (current height $-$ public height)
    current block ← block
    current height ← height of block
    private chain length $+1$
    if $\Delta = 0$ and private chain length $= 2$ then    ▷ Was 1-1 race, publish newly found block
        broadcast block to all miners
        private chain length ← 0
    else    ▷ Extend private chain length
        broadcast block to selfish miners
```

chain repeatedly and again count the number of times that in period T miner i continuously discovers two blocks, which we call $S_i^T(t)$. Based on the shuffled simulation, we define $\langle S_i^T \rangle$ as as the expected number of times a miner i continuously discovers two blocks in period T. Then, the MSB value for miner i is defined as

$$MSB_i^T = \frac{C_i^T - \langle S_i^T \rangle}{\sigma[S_i^T]}, \tag{6}$$

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 $\forall MSB_i > 2$. We then compute an average MSB score for all selfish miners, MSB.

V. Results

This section discusses the results of the simulations of the ABM model introduced in section IV.

Note that we refer to high block latency, slow block propagation times or slow block diffusion times synonymously.

A. *Relative selfish revenue*

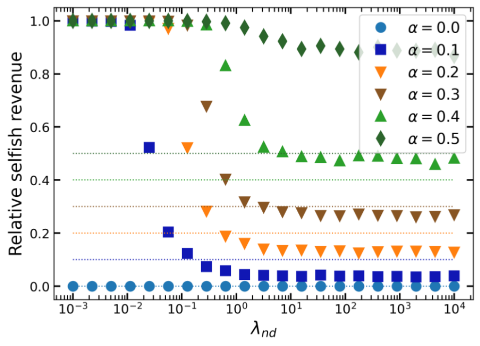
Figure 2 shows the impact of of introducing latency to [1]'s model of selfish mining. Note that λ_{nd} is the rate of block propagation. A high rate of block propagation implies short block diffusion times.

In Figure 2a, selfish mining is a more profitable strategy, if the graphs are above their respective "fair share" thresholds, as indicated by the respectively colored, dotted lines. For example, for $\alpha = 0.4$, we can see that relative selfish revenue never is below 0.4, meaning that selfish mining is more profitable $\forall \lambda_{nd}$.

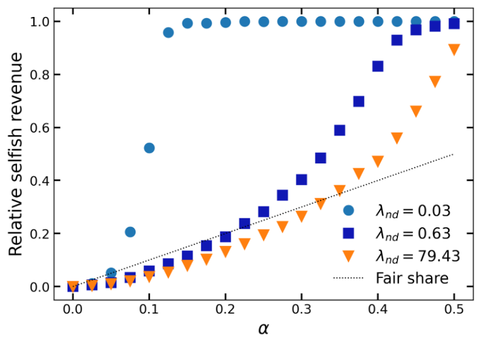
There are two important observations to be made. First, relative selfish revenue, as defined in subsection IV-C, is decreasing as block propagation time decreases. Second, for higher values of block latency (low λ_{nd}), selfish mining is always a more profitable strategy than honest mining. The selfish 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 (solo selfish miner). If it were a selfish mining pool and latency was equal for communication with pool members, the pool's profitability would suffer similarly. For faster in-pool communication, high latency again makes selfish mining relatively more profitable. Notably, for higher values of α selfish mining is more "resistant" 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

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.



(a)



(b)

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

diffusion is, the more mining power is required to surpass the fair share threshold for reasons stated above. Noteworthy is the fact that our model can confirm [1]'s main finding: Irrespective of how many block races selfish miners win, selfish mining is always relatively more profitable for mining power levels exceeding one third of the network' 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 $\forall\alpha \gtrapprox \frac{1}{3}$.

B. Hashing power distributions

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 for different hashing power distributions, namely a uniform random, power law, and exponential distribution. We show results for two latency regimes.

We observe that hashing power following an exponential distribution has negligible effect on the dynamics of the system, for either latency regime. The dynamics are indistinguishable to 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 (low λ_{nd}). First, relative selfish revenue crosses the fair-share-threshold only 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 is as follows: 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 node manages to outperform the selfish miner, thus creating the longest chain. For latency this high, the main chain is effectively built by one single miner, because block diffusion time exceeds the expected time between new blocks. However, [16] finds that Bitcoin mining power trends can be fit as exponential distributions.

C. Topologies

Next, we consider the effect of varying topologies on relative selfish revenue. Figure 3 shows relative selfish revenue as a function of selfish mining power for 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 dynamics of the system are very robust to changes in the underlying topology or hashing power distribution. The exception being powerful honest nodes in environments where block diffusion is very slow. This is more likely to exist in systems where hashing power is following a power law distribution. This is a relatively unfavorable environment for selfish mining.

D. MSB Index

To validate the Miner sequence bootstrapping model (MSB) as proposed by [9] consider Figure 4.

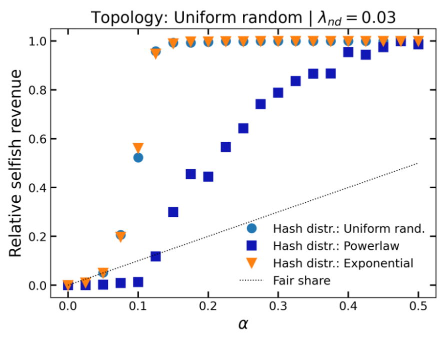
Figure 4a shows that for all latencies selfish mining is detectable with the MSB methodology. Notably, it does not depend on selfish mining being a relatively more profitable strategy. This is due to the fact that selfish miners will, regardless of operating below the fair-share threshold, have an increased probability of publishing two blocks in a row. Interestingly, the behavior is relatively robust to changes in selfish mining power. Trivially, for $\alpha = 0$ there are no selfish blocks and thus $MSB = 0$.

Figure 4b 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 minutes).

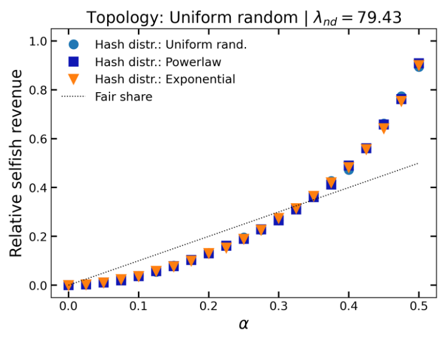
E. Limitations

In our ABM, we can assign each block to a specific miner, such that [9]'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 behavior through a Sybil "attack". Further, our results show that their approach

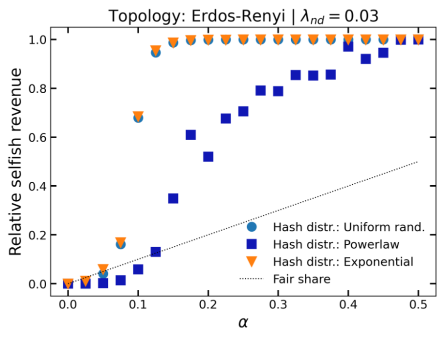
Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.



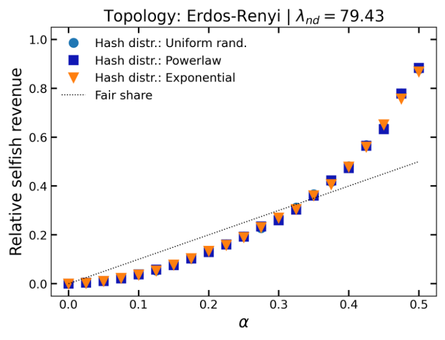
(a)



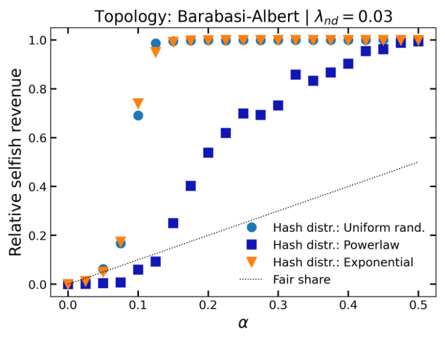
(b)



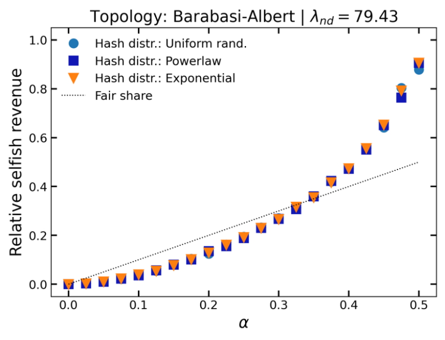
(c)



(d)

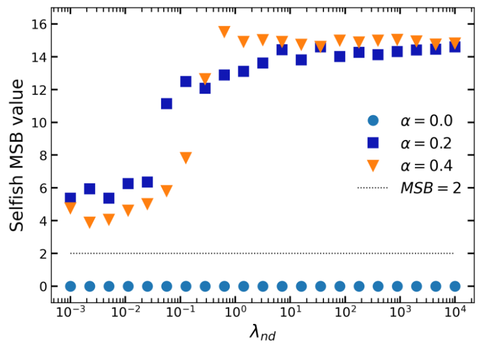


(e)

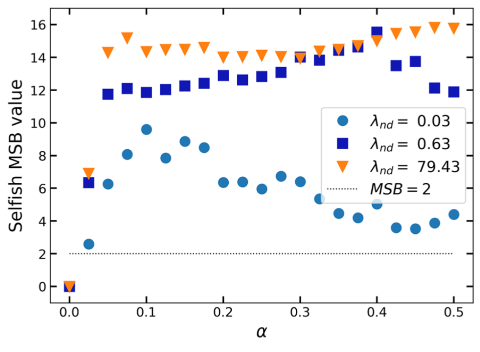


(f)

Fig. 3: Relative selfish revenue as a function of selfish hashing power (α) for different topologies and different hash power distribution



(a)



(b)

Fig. 4: MSB value as a function of latency (λ_{nd}) and selfish hashing power (α)

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.

produces results as expected, namely that selfish mining causes statistically significant levels of consecutive mined blocks. As much as a model can validate a methodology, our results confirm the work in [9].

Some of our simulations were limited by the relatively small network size, which may cause a lack of heterogeneity in the network [12]. We encourage to re-investigate some emerging properties for larger underlying networks.

VI. Conclusion

This paper mainly analyzed the effect of block propagation delays, network topology and mining power distribution on relative selfish revenue.We introduced an agent-based model that simulates the selfish mining strategy in a Bitcoin setting. The modeling of a peer-to-peer network structure allows us to extensively study the effects of latency. Our research suggests that selfish mining is a viable and profitable strategy. Future research may focus on tuning underlying model parameters based on empiric network size, topology and hashing power distribution.

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 [1]'s main observation that selfish mining is always relatively more profitable for $\alpha > 1/3$. Furthermore, we show that the dynamics of the selfish mining are very robust to changes in the underlying network topology. 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. Finally, using Miner Sequence Bootstrapping (MSB) index [9], we confirm selfish mining behavior also causes statistically significant levels of consecutive mined blocks.

The composability of our ABM has the advantage of allowing modifications of the behavior of agents easily. Future research should particularly analyze the effects of selfish mining pools. Further, a variation in block propagation times for communication within the selfish mining pool is an important dynamic to analyze.

Our paper shows that a combination of game theoretic analysis paired with agent-based modelling can help to advance the field giving insights into what effects strategic behavior has on the global properties of the system. Agent-based modeling 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 to bridge the gap between theory and empirical complex systems.

References

[1] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *International conference on financial cryptography and data security*, pp. 436–454, Springer, 2014.

[2] S. Bag, S. Ruj, and K. Sakurai, "Bitcoin block withholding attack: Analysis and mitigation," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1967–1978, 2016.

[3] A. Sapirshtein, Y. Sompolinsky, and A. Zohar, "Optimal selfish mining strategies in bitcoin," in *International Conference on Financial Cryptography and Data Security*, pp. 515–532, Springer, 2016.

[4] K. Nayak, S. Kumar, A. Miller, and E. Shi, "Stubborn mining: Generalizing selfish mining and combining with an eclipse attack," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 305–320, IEEE, 2016.

[5] H. Liu, N. Ruan, R. Du, and W. Jia, "On the strategy and behavior of bitcoin mining with n-attackers," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pp. 357–368, ACM, 2018.

[6] S. Solat and M. Potop-Butucaru, "Brief announcement: Zeroblock: Timestamp-free prevention of block-withholding attack in bitcoin," in *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pp. 356–360, Springer, 2017.

[7] R. Zhang and B. Preneel, "Publish or perish: A backward-compatible defense against selfish mining in bitcoin," in *Cryptographers' Track at the RSA Conference*, pp. 277–292, Springer, 2017.

[8] C. S. Wright and S. Savanah, "The fallacy of the selfish miner in bitcoin: An economic critique," 2017.

[9] S.-N. Li, Z. Yang, and C. J. Tessone, "Proof-of-work cryptocurrency mining: a statistical approach to fairness," in *2020 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, pp. 156–161, IEEE, 2020.

[10] S.-N. Li, Z. Yang, and C. J. Tessone, "Mining blocks in a row: A statistical study of fairness in bitcoin mining," in *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pp. 1–4, IEEE, 2020.

[11] C. Tessone, P. Tasca, and F. Iannelli, "Stochastic modelling of blockchain consensus." arXiv:2106.06465, 2021.

[12] M. Geier, C. J. Tessone, M. Vanotti, S. Vileriño, D. G. Márquez, and E. Mocskos, "Using network emulation to study blockchain distributed systems: The ethereum case," in *2019 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, pp. 51–58, IEEE, 2019.

[13] P. Tasca and C. J. Tessone, "A taxonomy of blockchain technologies: Principles of identification and classification," *Ledger*, vol. 4, no. 0, 2019.

[14] Y. Lewenberg, Y. Bachrach, Y. Sompolinsky, A. Zohar, and J. S. Rosenschein, "Bitcoin mining pools: A cooperative game theoretic analysis," in *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 919–927, Citeseer, 2015.

[15] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE P2P 2013 Proceedings*, pp. 1–10, IEEE, 2013.

[16] A. E. Gencer, S. Basu, I. Eyal, R. Van Renesse, and E. G. Sirer, "Decentralization in bitcoin and ethereum networks," in *International Conference on Financial Cryptography and Data Security*, pp. 439–457, Springer, 2018.

[17] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[18] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," *Journal of computational physics*, vol. 22, no. 4, pp. 403–434, 1976.

[19] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The journal of physical chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977.

Authorized licensed use limited to: University of Fribourg - Bibliothèque cantonale et universitaire. Downloaded on December 12,2022 at 15:07:40 UTC from IEEE Xplore. Restrictions apply.