# Analysis of Blockchain Selfish Mining: a Stochastic Game Approach

Boutaina Jebari[1,2], Khalil Ibrahimi[1], Mohammed Jouhari[1] , Mounir Ghogho[2,3]
[1]Ibn Tofail University, Faculty of Sciences, LaRI Laboratory, Kenitra, Morocco
[2]International University of Rabat, TICLab, Sala El Jadida, Morocco.
[3]University of Leeds, Faculty of Engineering, Leeds, UK
boutaina.jebari@uir.ac.ma,ibrahimi.khalil@uit.ac.ma,
mohammed.jouhari@uit.ac.ma, mounir.ghogho@uir.ac.ma.

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9839233

***Abstract*—Selfish mining is an attack on blockchain networks, where a minority mining pool deviates from the original mining protocol and keeps some blocks private. The goal of the attacking pool is to waste the computational power of the other miners and increase their revenue. In this paper, we use a new approach to analyze the profitability of such attacks. Using game theory, we model the interactions between pools to derive the utility of mining strategies. We simulate the game for a Bitcoin blockchain and analyze the profitability of an attack, in terms of the monetary award instead of the relative revenue. We express the utility to include the cost of a strategy and revisit existing selfish mining strategies to discuss possible outcomes of the game. Depending on the game parameterization, we highlight scenarios where the system could be compromised. To the best of our knowledge, this is the first work that models the selfish mining attack as a stochastic game.**

***Index Terms*—Selfish Mining, Game Theory, Bitcoin, Blockchain.**

## I. Introduction

Satoshi Nakamoto first proposed Bitcoin in 2008 as a decentralized digital currency that would allow secure transactions between peers without the need of a trusted third party. Now, more than a decade after its creation, Bitcoin is the most commonly used cryptocurrency and reached in November 2021 a record market price of $68,000[1]. Hence, even a minor attack on bitcoin would lead to major loss for all stakeholders, making the security of the system crucial for its existence.

Previous attempts to establish digital currencies failed as they could not make sure that spending a balance of the digital currency is carried out with one transaction only; this is known as the double spending problem. Bitcoin was constructed in a way that is computationally disadvantageous for malicious users to launch a double spending attack [1]. This is ensured by the Proof-of-Work (PoW) consensus, where miners compete to find the Hash of the new block, which is a computationally expensive problem. The more computational power or Hashpower a miner has, the more likely they are to win the race of finding the Hash of the new block. However, this inherited security was contested by several works. The authors of [2] present a comprehensive survey on security and privacy issues in bitcoin, including some major attacks such as the double spending, block withholding or eclipse attacks.

In particular, since the PoW tolerates a momentary fork of the public blockchain, some miners could choose to keep a number of blocks private before publishing them, thus creating deliberate forks on the public chain. This deviation from the bitcoin mining protocol is denoted as the selfish mining attack [3]. A successful attack occurs when a malicious pool's fork forfeits the public chain, allowing selfish miners to own more blocks and hence waste the hashrate of the honest miners.

In this paper, we define a new approach to analyze selfish mining attacks. Our goal is to verify whether a selfish mining attack can lead to the formation of a malicious pool with a majority hashrate. Although the attack is possible for most PoW-based blockchains, we focus our study on the bitcoin blockchain, without loss of generality. We use a game theory framework to define the players' best strategies depending on the system's parameterization. We summarize our main contributions as follows:

- We base our work on the model presented in [4] and extend it to a general sum stochastic game. To the best of our knowledge, this is the first work that applies a stochastic game approach to model selfish mining attacks.
- We study the profitability in terms of monetory rewards instead of the relative revenues and include new parameters such as the stale blocks, the cost of mining and the block size.
- We apply our model to the most famous mining strategies and analyze players' utilities to derive profitability thresholds.

## II. Related Work

The selfish mining attack was first introduced by Eyal and Sirer in [3]. The authors proposed a mining strategy that allows an attacking pool to waste the computational power of the other miners and increase their own relative revenue; see Algorithm 1 in [3] for details. They modeled the bitcoin system under the selfish mining attack as a Markov chain and analyzed the pools' revenues with respect to their relative hashrate and the propagation rate of the published blocks. Their results show that it was possible for a pool with a relative hashrate of $25\%$ to drive the honest miners to follow a selfish mining

This work was supported by the RSK-e-mobility project which is partly funded by the Rabat-Salé-Kenitra (RSK) region

[1]https://www.theguardian.com/technology/2021/nov/09/bitcoin-price-record-high-cryptocurrencies-ethereum

strategy until they reach an attacking pool with $51\%$ or more of the system's hashpower. In [5], the attack is extended to the case of two attacking pools. The authors construct a finite state machine of the system and derive the analytical expression of the attacker's relative revenue. Their findings show that it is possible to reach a profitability threshold of $21,48\%$ instead of the $25\%$ obtained in the case of one attacking pool.

Several other works question whether the selfish mining strategy introduced in [3] was the optimal one for the attacking pool, and propose to use a Markov Decision process (MDP) to define the best selfish mining strategies. The authors in [4] formalize the selfish mining strategy as a single-player decision problem. They introduce a set of actions and states and analyze possible strategies that will allow the selfish pool to maximize its own relative revenue. Their work shows that, indeed, there exists another deviation from the bitcoin mining policy with better profitability thresholds. Similarly, the authors of [6] analyze selfish mining in the case of multiple attackers and model the system as a multi-player decision process. They link the profitability of an attack not only to the attacker's relative revenue but also to its stale block rate, which quantifies the number of mined blocks that were not accepted in the public chain. In [7], the authors combine selfish mining attacks to eclipse attack and retrieve best strategies using a MDP. They show that the success of a selfish mining attack does not only depend on the hashrate of the selfish pool but also on other parameters such as the cost of mining, propagation ability of blocks or even the block size.

On the other hand, in [8], the authors review how previous works define the profitability of selfish mining. Since the relative revenue is chosen as a metric for profitability, it is assumed that a strategy is more profitable if it allows to have a better share of published blocks. However, selfish mining allows an increase of the selfish pool's relative revenue at the expense of decreasing both players' revenues. In this sense, the authors of [8] criticize the choice of a pool's relative revenue as a profitability metric and define instead the profit of a strategy as the difference between the rewards and cost, averaged by time. They then demonstrate that the selfish mining strategy defined in [3] is less profitable than honest mining, regardless of the hashrate of the selfish pool.

In this work, we study these repeated interactions between the two mining pools, and see if, depending on the parameters mentionned above and the chosen strategies, the honest pool can force the selfish pool to give up the selfish strategy and follow an honest mining strategy. To do so, we model the aformentioned repeated interactions as a stochastic game, and analyze the players' best responses to different mining strategies.

## III. Selfish Mining Game Model

We consider a PoW blockchain with two mining pools: a pool that mines honestly, called honest pool, and a selfish or attacking pool that aims to increase its profit by deviating from the original protocol. The two pools compete to find the PoW solution first. When a block is found, a pool decides on the best action to take to maximize its profit. We model the repeated interactions between the two pools as a stochastic game between two players: the honest and the selfish pools. As both players can gain profit from a round of the game, we define the game to be a general-sum stochastic game. To do so, we make the following assumptions:

- To rule out the $51\%$ attack scenario, we suppose that the relative hash power of the selfish pool, denoted $\alpha$, is less than 51%. We also assume that the probability of a pool to find a block is proportional to its hashrate. The selfish pool (respectively honest pool) has therefore a probability $\alpha$ (rsp. $1-\alpha$) to find a block first.
- When a fork is created in the public blockchain, it is possible for the honest pool to mine after the selfish pool's fork, as shown in Fig.1. Each branch in the figure represents a possible scenario for the fork resolution. For example, the branch $B$-$S_1$-$H_2$ is the scenario where the honest pool publishes the block $H_2$ after the selfish pool's block $S_1$. The block $B$ represents the last block of the public blockchain before the fork. This was linked to the propagation ability of the blocks and modeled as a redistribution of the hash power [5][6]. In this work, we assume that the honest player can decide on the fork to mine after.

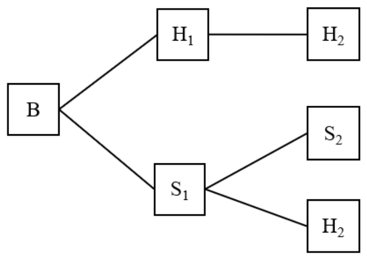


Fig. 1. Possible fork resolution scenarios

- Several works linked the security of a blockchain network to the block size and transaction arrival rate [9, 10]. To include this parameters in our work, we suppose that players do not start mining until enough transactions are gathered to mine a block. Let $\lambda$ be the transaction arrival rate and $K$ the number of transactions required to form a block. We make the simplified assumtion that the probability of a block formation $p_b$ is equivalent to the probability that the number of arrived transactions is bigger than the number required to a form a block. Let $Tr$ be the random variable describing the number of transactions that arrived during a time interval $T$; $Tr$ follows a Poisson distribution with parameter $\lambda$; so, $p_b$ can be expressed as follows:

$$p_b = p(Tr \geq K) = 1 - \sum_{k=1}^{K-1} \frac{\lambda^k}{k!} \mathrm{e}^{-\lambda} \qquad (1)$$

We formally define our stochastic game as a tuple $\mathcal{G} = \langle \mathcal{P}, \mathcal{S}, (\mathcal{A}_i)_{i\in\mathcal{P}}, (u_i)_{i\in\mathcal{P}} \rangle$. Next, we will detail our game's players, actions and utility functions.

*1) The players $\mathcal{P}$:* We assume our game to be played by two players: the honest pool and selfish pool called respectively $H$ and $S$. Let $\mathcal{P} = \{H, S\}$ be the set of players.

*2) The actions $(\mathcal{A}_i)_{i\in\mathcal{P}}$:* Each player can choose, throughout the game, between a set of available actions. For the selfish pool, we adopt the actions proposed in the literature, defined as follows:

- **Adopt,** or action A: the attacker accepts the honest pool's chain and decides to give up on his private chain.
- **Override,** or action O: the attacker publishes more blocks than the longest fork on the chain, overriding the honest pool's chain.
- **Match,** or action Ma: the attacker publishes as many blocks as those on the longest fork on the public chain, creating a competition between the honest and selfish players.
- **Wait,** or action W: the attacker keeps mining in its private chain and does not publish any blocks.

On the other hand, as it is not in the honest pool's best interest to compromise the system, we assume that it will always publish a block as soon as it is found. In the case where the selfish miners published some or all of their private chain, causing a fork on the public chain, the honest pool can mine either after the honest or the malicious chain. We define these two possible actions as 'Hit' or 'Miss' for the honest pool. We also give the honest player the possibility of a ' No Action' as it is possible that he is not aware of the other players' action or if he has no blocks to respond with. The possible actions for the honest player are therefore:

- **Hit**, or action Ht: it describes when the honest pool adds a block on the public chain or the honest pool's chain, if a fork is active.
- **Miss**, or action Mi: it describes when the honest pool adds a block on the selfish pool's chain. This action is only feasible if a fork is active.
- **No Action**, or Na: the honest pool is mining but has no block, so it takes no actions.

Let $\mathcal{A}_i$ be the set of possible actions for player $i$ for $i \in \mathcal{P} = \{H, S\}$. From what we described above, we set $\mathcal{A}_S = \{\mathrm{A}, \mathrm{O}, \mathrm{Ma}, \mathrm{W}\}$ and $\mathcal{A}_H = \{\mathrm{Ht}, \mathrm{Mi}, \mathrm{Na}\}$

*3) The state space $\mathcal{S}$:* we base our work on the model presented in [4]. We extend their state space to include other parameters such as the number of mined blocks that were not accepted, also known as the stale blocks. We define the state of our game at a stage $t$ as $s^t = (s^t_S, s^t_H, \mathrm{fork})$ where $s^t_i = (n_{im}, n_{ip})$ for $i \in \mathcal{P} = \{H, S\}$ with:

- $n_{im}$ being total number of mined blocks by player $i$.
- $n_{ip}$ number of mined blocks that are either published on the public chain or currently competing to be accepted.
- fork is relevant to define the set of possible actions at a certain state, defined in [4] as either Relevent, Irrelevant or Active. We extend it and add the value resolved to fit our model as follows:
  - **Relevant**, denoted as Re: this means that the honest pool found the last block, and that if the selfish pool has enough blocks, the action match is now feasible.
  - **Irrelevant**, denoted as I: this means that when the attacker found a block the previous block had already propagated through the network. The action Match is therefore ineffective.
  - **Active**, denoted as Ac: meaning that the selfish pool played a Match action and a fork in the public chain is in process. For the honest pool, the action Miss is now possible.
  - **Resolved**, denoted as Rv: the selfish pool took an action Adopt or Override resolving the fork. This state ends the round of the game.

*4) Utilities:* At each stage $t$ of the game, each player $i \in \mathcal{P} = \{H, S\}$ takes an action and moves to stage $t+1$ gaining a utility $u_i(s^t, a^t)$ defined as :

$$u_i(s^t, a^t) = r_i(s^t, a^t) - c_i(s^t, a^t) \tag{2}$$

where

$$c_i(s^t, a^t) = p_b.(\theta_i(s^t, a^t) + c_i.\tau_0) \tag{3}$$

and

$$r_i(s^t, a^t) = p_b.\rho_i(s^t, a^t) \tag{4}$$

- $r_i(s^t, a^t)$ is the reward of player $i$ at state $s^t$, knowing that both players took the action vector $a^t$. As miners mainly gain rewards from mining a block, we set the reward of mining a block to 1 unit and define $r_i(s^t, a^t)$ as the number of blocks published times the probability that a block is formed.
  And since players do not gain any rewards until the current fork in the public network is resolved, we write:

$$\rho_i(s^t, a^t) = \begin{cases} n_{ip} & \text{if fork} = \mathrm{Rv} \\ 0 & \text{Otherwise} \end{cases} \tag{5}$$

- $c_i(s^t, a^t)$ quantifies the cost of a player $i$ at state $s^t$, when the players took the action vector $a^t$. The authors of [8] emphasized that the overall cost is linked to the time spent mining. Indeed, assuming that the pool is always mining at its full capacity, the cost of mining per unit of time depends on external parameters (used equipment, electricity consumption, electricity fees...). We express this in the term $c_i.\tau_0$ with $c_i$ being the cost of mining of player $i$ per unit of time and $\tau_0$ the duration of a stage. However, this value is expected to be negligible compared to the reward of mining a block in the case of mining pools. To have a more significant cost function, we include the lost rewards of blocks mined that were not accepted in the public chain due to the players' actions. The stale blocks are expressed in $\theta_i(s^t, a^t)$ as follows:

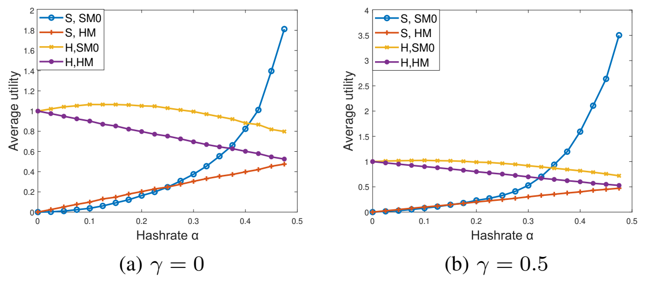
$$\theta_i(s^t, a^t) = \begin{cases} n_{im} - n_{ip} & \text{if fork} = \mathrm{Rv} \\ 0 & \text{Otherwise} \end{cases} \tag{6}$$

In one stage of the game, the mining pools compete to find the solution of the PoW. We recall that the players S and H find a block first w.p. $\alpha$ and $1-\alpha$ respectively. Once a block is mined, players take an action vector depending on the owner of the block and the state they are in. This results in a change of the system's state and the game moves to the next stage. Following this description, a stage of the game is set as the block interval time. As the difficulty adjustment is outside the scope of this paper, we suppose the block interval to be constant, denoted by $\tau_0$. A round of the game stops when $S$ forfeits his private chain or the current fork on the public chain is resolved. The players then restart another round. Setting a selfish mining strategy for a player is equivalent to setting the actions to take depending on the game's state and the block owner. In this sense, the selfish mining strategy defined in [3] is a deterministic strategy for player $S$. On the other hand, when a fork is active, the honest pool plays a mixed strategy of Mi and Ht with probability $\gamma$ and $1-\gamma$ respectively. We set the strategy to be mixed, since at first, $H$ might not be aware that the selfish pool's fork is malicious. Table I illustrates a round of the game. In this example, the selfish pool finds a block first and decides to keep it private (action W), and $H$ takes no action (Na) as he is unaware of this event. In the next stage, the honest pool finds a block (action Ht). The player $S$ responds with action Ma and publishes his private block, creating a fork. Next, $S$ finds a block first and overrides the honest chain. The fork is then resolved and the game restarts.

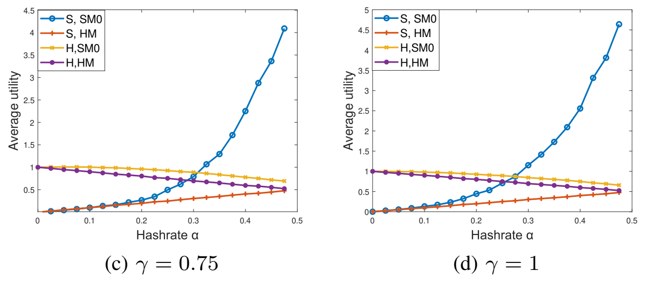
## IV. Numerical Results and Discussion

In this section, we simulate the presented game to analyze the profitability of the players' strategies. To do so, we built a selfish mining simulator using Python. In this paper, we focused particularly on the existing strategies with some variations of relevant parameters. We examine the impact on the selfish player's hashrate $\alpha$ on both players' utilities and determine players' best response to each other. In the remainder of the paper we denote by SM0 the selfish mining strategy presented in [3]. HM denotes the honest strategy where a player publishes a block as soon as it is found. In case of a fork, the player uses a mixed strategy to decide the branch to mine on with a probability $\gamma$.

First, we analyze the utility of strategies when only the average published blocks is considered. To do so, we study the utility of players under SM0 and HM for $p_b = 1$, $c_S = c_H = 0$ and $\theta_S = \theta_H = 0$, for different HM strategies. Fig.2 shows the players average utilities per game round, when $S$ plays SM0 and H plays HM. Figures 2a, 2b, 2c and 2d correspond to $H$ playing HM with a mixed strategy with probability $\gamma = 0$, $\gamma = 0.5$, $\gamma = 0.75$ and $\gamma = 1$ respectively. The results show that if player $S$ focuses solely on the number of published blocks to determine his strategy, he might choose to follow SM0. Indeed, the hashrate's threshold beyond which the utility of SM0 surpasses the utility of HM can be equal or lower than 0.25, which means that he might be getting higher number of blocks in a round of a game. However, this strategy is not profitable for the main reason that this increase is due to the



(a) $\gamma = 0$ (b) $\gamma = 0.5$



(c) $\gamma = 0.75$ (d) $\gamma = 1$

Fig. 2. Players utilities under SM0 and HM with $pb = 1$, $c_S = c_H = 0$ and $\theta_S = \theta_H = 0$

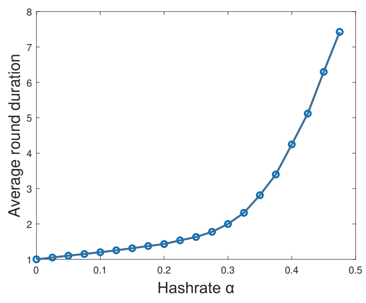


Fig. 3. Round duration w.r.t to player S hashrate.

fact that the more hashrate $S$ has, the longer a round of the game will take. This also explains why the utility of the player $H$ also increases under a strategy SM0 of player $S$. To verify this, we plot the average duration of a game round, with $\tau_0$ as a unit of time, w.r.t the hashrate of player $S$ in Fig.3. We indeed notice that the utility of $S$ follows the layout of the curve shown in Fig.3.

Fig.4 represents the players' round utilities averaged by the expected duration of a round when $S$ follows SM0 and $H$ follows HM $\gamma = 1$. The figure shows that, under this parameterization, his utility is still approaching his expected utility when he plays HM. Since this is the best-case scenario for player $S$, this result supports the claim in [8], that SM0 is not profitable no matter the hashrate of the pool.

In this part, we fix $p_b = 1$ and focus on analyzing players' utilities when the cost is taken into account. Fig.5 represents players utilities for SM0 and HM strategies, averaged by the expected round duration. Figures 5a and 5b correspond to $H$ playing HM with $\gamma = 0.5$ and $\gamma = 0$ respectively. First, we notice how the term $\theta_i(s^t, a^t)$ in the utility function allows

TABLE I
EXAMPLE OF A GAME ROUND

| Stage | Current State | actions | Next State | Stage utility |
|---|---|---|---|---|
| 1 | ((0,0),(0,0),.) | (W,Na) | ((1,0),(0,0),I) | $(-c_S.\tau_0, -c_H.\tau_0)$ |
| 2 | ((1,0),(0,0),I) | (Ma,Ht) | ((1,1),(1,1),Ac) | $(-c_S.\tau_0, -c_H.\tau_0)$ |
| 3 | ((1,1),(1,1),Ac)) | (O,Na) | ((2,2),(1,0),Rv) | $(2-c_S.\tau_0, -1-c_H.\tau_0)$ |

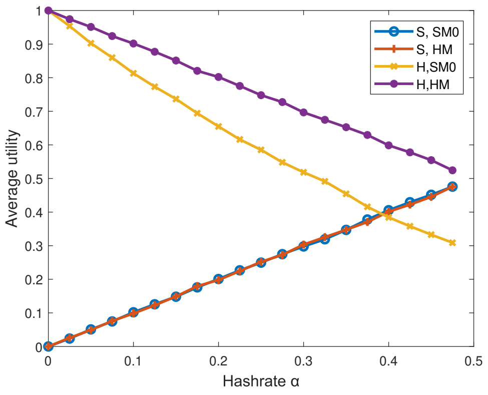
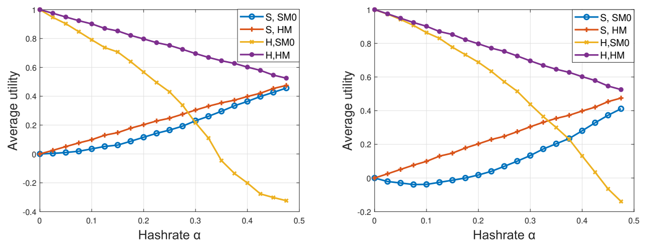


Fig. 4. Utility of player $S$ averaged by round duration w.r.t the hashrate $\alpha$.



(a) $\gamma = 0.5$ (b) $\gamma = 0$

Fig. 5. Players utilities for SM0 and HM when the stale blocks are included, w.r.t the hashrate $\alpha$.

us to quantify the loss of $H$'s hashpower, resulting from the selfish attack. Let us consider the scenario where, at first, player $H$ unaware of player $S$ attack, mines honestly with a fair mixed strategy $\gamma = 0.5$. As shown in Fig.5a $S$ playing SM0 can reduce $H$'s utility by a noticeable amount, even with a hashrate of 0.1. Detecting this sudden decrease, player $S$'s best response would be to play HM with $\gamma = 0$. In this case, as we can see in Fig.5b, the utility of $S$ will be slightly lower than 0 for a hashrate lower than 0.17. Under this threshold, it is in $S$'s best interest to switch to an honest strategy, as SM0 would be costly compared to HM. We also notice that under certain thresholds, $H$ 's utility becomes negative. In this case, we acknowledge the possibility that the honest pool might defect and chose to mine selfishly, compromising the blockchain system. However, under $H$'s best defense ($\gamma = 0$) this threshold is close to 0.43, which is hard to reach.
Under realistic values of $c_i$, which are negligeable compared to the block reward, the term $c_i\tau_0$ in the cost function has, as expected, little to no impact on players' utilities.

So far, the best defense strategy for the honest pool was to always play Hit when a fork was active ($\gamma = 0$). However, it is also possible that $H$ changes some blockchain parameters to defend against selfish mining. Indeed, we assumed that the honest pool has the majority hashrate, which is considered equivalent to the pool size in the literature. This means that, with a majority vote, they could parameterize the blockchain to lower the attacker's utility. Let us assume that we are in the case where $H$ is best defending with $\gamma = 0$ but he still notices an important decrease in his utility. The system is at high risk of being compromised if $H$ decides to deviate from bitcoin's original mining strategy. In this case, lowering the block formation probability $p_b$ for the sake of lowering S's utility would be an option to protect the system. In this game, we considered a stage duration to be equal to the block interval $\tau_0 = 10$min. Calculations from [11] show that transaction arrival rate per block interval can be estimated to $\lambda = 2100/\tau_0$. Fig.6 shows the players' utilities w.r.t the required number of transactions to form a block, $K$, when $H$ plays HM with $\gamma = 0$ and $S$ plays SM0 and $\alpha = 0.42$. We can see that when $H$ increases the required number of transactions $K$, which is equivalent to increasing the required block size, the utility of $S$ decreases significantly. Under this situation, the selfish pool might be forced to go back to honest mining.

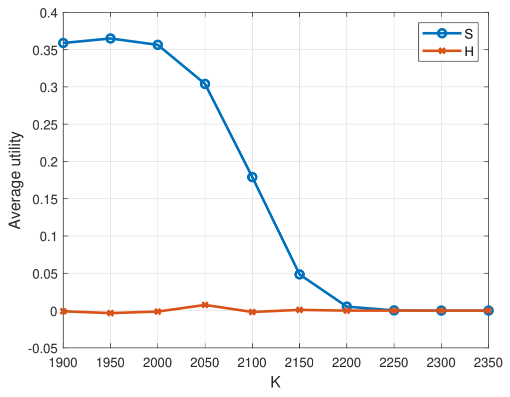


Fig. 6. Players utilities under SM0 w.r.t the required number of transaction per block $K$.

Finally, we simulate the stubborn strategy presented in [12] and compare it to HM and SM0. In this strategy, $S$ waits until $H$ has the lead of $N$ blocks to give up on his private chain. We denote by ST($N = n$) the stubborn strategy where the player waits n blocks before accepting the longest chain. We set $\gamma = 0$ and $p_b = 1$. Fig.7 shows both players utilities when $S$ plays SM0 and ST($N = 2$). We notice that SM0 is slightly more aggressive than ST($N = 2$). Fig.8 represents $S$'s utility when playing ST for different values of $N$. We can see that, the utility of $S$ decreases with the increase of $N$, making SM0 more profitable for $S$ regardless of the chosen value of $N$.

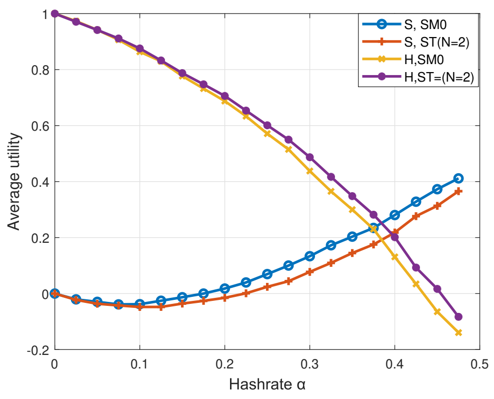


Fig. 7. Players' utilities when $S$ plays SM0 and ST(N=2) w.r.t $S$'s hashrate.

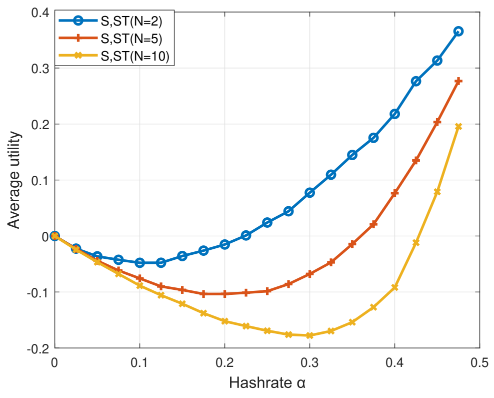


Fig. 8. Utility of player $S$ when playing ST for $N$=2,5 and 10.

## V. Conclusion

In this paper, we investigated the profitability of selfish mining using game theory and modeled the repeated competition between pools as a stochastic game. We expressed players' utilities to include the reward and cost of mining in terms of accepted blocks, stale blocks, mining cost and the probability of block formation. It allowed us to analyze the profitability of mining strategies, derive possible game outcomes and highlight scenarios where the system could be compromised. Since a blockchain network contains multiple pools, this proposed approach could be generalized to study a selfish mining attack with multiple attackers and derive possible equilibria of the whole system. Although we focused the analysis of the profitability on the pool's scale, we believe this work offers a framework that could be extended, in future work, to study the profitability on a single miner's scale.

## References

[1] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: *Cryptography Mailing list at https://metzdowd.com* (Mar. 2009).

[2] Mauro Conti, E. Sandeep Kumar, Chhagan Lal, and Sushmita Ruj. "A Survey on Security and Privacy Issues of Bitcoin". In: *IEEE Communications Surveys Tutorials* 20.4 (2018), pp. 3416–3452. DOI: 10.1109/COMST.2018.2842460.

[3] Ittay Eyal and Emin Sirer. "Majority Is Not Enough: Bitcoin Mining Is Vulnerable". In: vol. 8437. Nov. 2013. DOI: 10.1007/978-3-662-45472-5_28.

[4] Ayelet Sapirshtein, Yonatan Sompolinsky, and Aviv Zohar. "Optimal Selfish Mining Strategies in Bitcoin". In: *Financial Cryptography*. 2016. DOI: 10.1007/978-3-662-54970-4_30.

[5] Qianlan Bai, Xinyan Zhou, Xing Wang, Yuedong Xu, Xin Wang, and Qingsheng Kong. "A Deep Dive Into Blockchain Selfish Mining". In: *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*. 2019, pp. 1–6. DOI: 10.1109/ICC.2019.8761240.

[6] Hanqing Liu, Na Ruan, Rongtian Du, and Weijia Jia. "On the Strategy and Behavior of Bitcoin Mining with N-attackers". In: May 2018, pp. 357–368. DOI: 10.1145/3196494.3196512.

[7] Arthur Gervais, Ghassan O. Karame, K. Wüst, Vasileios Glykantzis, Hubert Ritzdorf, and Srdjan Capkun. "On the Security and Performance of Proof of Work Blockchains". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016).

[8] Cyril Grunspan and Ricardo Pérez-Marco. "On profitability of selfish mining". In: *ArXiv* abs/1805.08281 (2018).

[9] Yonatan Sompolinsky and Aviv Zohar. "Secure High-Rate Transaction Processing in Bitcoin". In: Jan. 2015. ISBN: 978-3-662-47853-0. DOI: 10.1007/978-3-662-47854-7_32.

[10] Hafsa Bennadi, Mohammed Jouhari, Khalil Ibrahimi, and Abderrahim Benslimane. "Securing IoT Transaction Against Double Spending Attacks Based On Signaling Game Approach." In: *IEEE GLOBECOM 2021* (Dec. 2021).

[11] Raheel Memon, Jian Li, and Junaid Ahmed. "Simulation Model for Blockchain Systems Using Queuing Theory". In: *Electronics* 8 (Feb. 2019), p. 234. DOI: 10.3390/electronics8020234.

[12] Kartik Nayak, Srijan Kumar, Andrew Miller, and Elaine Shi. "Stubborn Mining: Generalizing Selfish Mining and Combining with an Eclipse Attack". In: *2016 IEEE European Symposium on Security and Privacy (EuroS P)*. 2016, pp. 305–320. DOI: 10.1109/EuroSP.2016.32.