Profitability of the coin-hopping strategy

Tamás Király* and Lilla Lomoschitz*

Abstract

In the world of cryptocurrencies, new currencies are often created by forking existing ones and changing some of the rules. When the cryptographic hash function used in mining is left unchanged, miners may effortlessly switch between the two blockchains. If there are enough loyal miners on both chains, then the miners who always switch to the more profitable chain can achieve higher profit while the average profit of miners decreases. We study the profitability of this coin-hopping strategy as a function of the hashpower of the coin-hoppers compared to the loyal miners. Our simulations suggest that maximum profit is achieved at a relatively low hashpower (12% of the total). Interestingly, this optimal coin-hopping hashpower does not depend much on the difference of hashpower between the loyal miners on the two chains. However, the asymmetry of the two chains significantly impacts the profit of loyal miners. Somewhat counterintuitively, those on the weaker chain earn more profit than those on the stronger chain if the difference is not too large.

1 Introduction

When multiple cryptocurrencies are using the same hash function, miners can easily switch to another currency even with specialized hardware. *Coin hopping* is a strategic mining behavior that instantly switches if mining another currency becomes more profitable. Although this seems to be the rational behavior, in reality one can observe that a considerable portion of miners are loyal to a particular coin, or are reluctant to frequently switch chains due to other reasons. In this paper we study how the relative hashpower of coin-hoppers and loyal miners influences the profit of these strategies, as well as the total profit made by the miners. It was already observed by Meshkov et al. [1] in a simplified model that coin-hopping decreases the total profitability of the coins. We quantify this decrease in profit using simulations, and also observe several interesting phenomena related to the relative profit of the different strategies. In particular, we show that coin-hoppers achieve maximum profit if they have approximately 12% of the total hashpower. The profit of loyal miners also shows interesting properties if their hashpower is different on the two chains: for some parameters, those with smaller hashpower make considerably more profit.

*MTA-ELTE Egerváry Research Group, Department of Operations Research, Eötvös Loránd University, Budapest. Email: tkiraly@cs.elte.hu, lomolilla@gmail.com

The structure of the paper is as follows. In the remainder of the Introduction, we present the basics of Bitcoin and mention some known strategic attacks. We also describe the difficulty adjustment method of Bitcoin, which is part of our model. In Section 2, we present the simplified coin-hopping model of Meshkov et al. [1], and introduce a more refined model that is used in our simulations. Section 3 presents the details of the simulation and the results.

1.1 Bitcoin

Bitcoin is the first blockchain-based cryptocurrency, introduced in 2008 by Satoshi Nakamoto [2]. It uses digital signatures, and money is represented as values linked to public keys. To make a bitcoin transaction, one needs to know the private key of the sender and the public key of the receiver, and one has to send a properly formatted message into the bitcoin network, signed by the private key. The network consists of nodes that check the validity of the sent transactions, do the bookkeeping by arranging the accepted transactions into blocks, and broadcast these blocks in the network.

Each block is created only by one node, but all nodes should inspect and verify the received blocks. Each node stores in its own memory the exact amount of currency linked to all public addresses. Once a new block arrives (and the node accepts it), the node subtracts the value of the transaction from the sender's account and adds the same value to the receiver's account. If any transaction is faulty in a block (it exceeds the sender's balance), then the whole block is rejected by any honest node. So a rational node will not put any faulty transaction into a block.

Besides the transactions, each block contains additional data. The header includes a link to the previous block (a hash of it), a hash of all transactions in the current block, a timestamp, a difficulty setting, and a nonce. The hash of the header has to be in a given domain: it must be smaller than a predefined value, which is set in the difficulty. To reach this condition, a node has to change the nonce several times, and calculate the hash, until it becomes an acceptable value. In this system, the nodes are competing to be the first to make the next block, so several nodes are calculating the hashes. However, only the first one will be the owner of a block – the others will have to restart hashing after receiving the new block, and processing the transactions therein. This concept is called proof of work, as the nodes have to spend computational power (hashpower) to create a new block, and thus have a say in which transactions to accept and include in the system. This work is also called mining, as the owners of the blocks receive rewards.

There are two kinds of rewards. First, there is a fixed amount of block reward: the owner receives a preset amount of bitcoins if they broadcast an acceptable block. This reward is declared in the first transaction of the block, and it sends bitcoin to the node's public address from nowhere, so this way new bitcoins are generated. The amount of new bitcoins emerging this way is bounded, as the block rewards are halved every 210000 blocks. The other type of reward comes from transaction fees. Each transaction can have a positive difference between the amount sent and the amount received, indicating that the remainder is a fee paid by the sender.

There is an upper limit on the number of transaction in a block, however empty

blocks are acceptable. As the number of transactions to be processed can be more than the amount that fits in a block, it is rational for a transaction sender to give away some transaction fee, in order to get the transaction in a block faster. In [2], it was argued that until the majority (in computational power) of nodes is honest, there is no viable attack against the system.

1.2 Some undesirable mining strategies

In the past few years, several research papers showed that miners without majority may also have the incentive to deviate from honest mining, and thus harm the system. Eyal and Sirer [5] invented the idea of *selfish mining*, where the miner postpones the broadcast of a newly found block in order to lower the chances of other miners to create the next block of the blockchain, and also to make them waste computational power by trying to mine a block with lower probability of being included in the chain. They also show an improvement recommendation, which makes the system resistant to selfish mining by groups smaller than 25% of the total hashpower.

Carlsten et al. [6] studied the difference between the current era where block rewards dominate, and the 'transaction fee era' when block rewards become negligible. They showed that the lack of block rewards enables new kinds of strategic behavior. Namely, it could be beneficial for miners not to continue the longest chain, but mine on top of a previous block, where more transactions (hence transaction fees) are available. The miner can then incentivize mining on top of their block by including only a fraction of the available transactions in the block, an leaving the remaining transaction fees up for grabs. These kinds of strategies have several variations and [6] shows the existence of Nash equilibria with quite compex strategies. This results in a lot of short branches on the blockchain, which undermines the profitability of the whole system.

1.3 Difficulty adjustment

Bitcoin is designed in a way that new blocks should emerge approximately every 10 minutes. If the total hashpower of miners in the system increases, then the average inter-block time decreases, so a mechanism is needed to adjust the average time of finding a block. This is done by changing the difficulty settings. If the hash function's acceptance domain is shrunk, then each nonce has a smaller probability to make an acceptable block, hence the average time to find a new block increases. More precisely, the time intervals between the blocks are exponentially distributed, so if the difficulty threshold (the upper bound on the hash value) is changed by a factor of α, then the expected time between blocks (inter-block time) is multiplied by $1/\alpha$. So by changing the difficulty threshold, a relatively stable block arrival time can be achieved, even if the total hashpower varies. In the bitcoin protocol, difficulty adjustment happens every 2016 blocks. If the 2016 blocks are found α times faster than the target (2016*10 minutes), then the difficulty threshold is modified by a factor of $1/\alpha$. In this context, the *difficulty* of the chain is defined as the inverse of the difficulty threshold.

Not all cryptocurrencies have the same mechanism for difficulty adjustment. For example, Bitcoin Cash (BCH) has a similar structure as Bitcoin, but the difficulty is

updated after each new block, using a moving average over 144 blocks [3]. Before the introduction of this method, Bitcoin Cash had a more complicated difficulty adjustment method [4], as it wanted to mitigate the risk of very long inter block times due to lack of hashpower. The main idea was to lower the difficulty by 20% if the majority of the latest inter-block times were over 12 hours. For this, Median Time Past (MTP) was used, which is the median of the last 11 block finding times. If the MTP was more than 12 hours for the last 12 blocks, then difficulty was lowered by 20%. This created large fluctuations in the profitability of mining BCH, which resulted in a lot of miners rapidly switching from BTC to BCH when the difficulty was low. After mining 2016 blocks very fast, the BCH difficulty was readjusted, and miners partially returned to the BTC network. The problem with this situation was that on average BCH was mined much more rapidly than the target, resulting in a rapid growth of the number of BCH coins.

2 Coin hopping

Coin hopping, sometimes called chain hopping, is a strategic mining behavior, where the miner constantly monitors the difficulty and the price of several cryptocurrencies, and always decides to mine the most profitable one. This is only typical for cryptocurrencies using the same hash function, as having the same hash function makes the change between mining algorithms easier. Although coin hopping is not really an attack, in this paper we consider coin-hoppers as adversaries to loyal miners who stay on the same chain forever.

To our knowledge, the first to publish a scientific paper about coin hopping were Meshkov et al. [1], who also suggested a possible countermeasure. They assumed that there are two identical cryptocurrencies (C_1, C_2), with the same amount of loyal miners on each chain, who do not leave their chain, and there is an adversary who can switch between the chains any time, and tries to maximize its profit. The adversary stays on chain C_2 until a difficulty adjustment occurs, when it switches to C_1, and it moves back to C_2 after the next difficulty adjustment, and keeps on hopping from one coin to the other after each adjustment.

The model in the paper assumes that the difficulty adjustments happen at the same time on both chains, which is not how they happen in reality. Nevertheless, we can get some insights from this model, so suppose that this is how the system works. In this scenario, the adversary speeds up the block findings on the chain on which it is mining, so after the difficulty adjustment this chain would become harder to mine. On the other hand, on the blockchain where there is no adversary, the average inter-block period will be longer than intended (because the adversary was there at the time of the difficulty adjustment, but now it is not there anymore), so at the next adjustment the difficulty will drop. This way the adversary is always on the chain that has smaller difficulty and hence higher profitability: it gets more coins per unit hashpower than the loyal miners.

As in [1], let R_0 denote the hashpower of honest miners of both C_1 and C_2, and R_a the hashpower of the adversary, where $R_a < R_0$. Let $p := R_a/R_0$ denote the size of

the adversary compared to the size of the miners who always stay on chain C_1. Let Δ be the target inter-block time (10 minutes in Bitcoin), and let M be the number of blocks between difficulty adjustments (2016 in the case of Bitcoin). The difficulty is defined as $D = 1/T$ where T is the highest permissible output of the hash function. If the adversary switches to the other chain at each difficulty adjustment, then the chain it leaves will get a difficulty of $(R_0 + R_a) \cdot \Delta$, and the chain that it enters will have a difficulty of $R_0 \cdot \Delta$. So the adversary will need $R_0 \cdot \Delta$ hashpower to get a block in a unit of time on average. For the other players, the required hashpower will alternate: either $R_0 \cdot \Delta$ or $(R_0 + R_a) \cdot \Delta$. This means that the adversary will find $M \cdot R_a/R_0$ blocks on average in the period between two difficulty adjustments, while the other, loyal players will only get $(M \cdot R_0/(R_0 + R_a) + M \cdot R_0/(R_0))/2$ blocks on average. Altogether, the adversary does not only decrease the income of the other miners, but decreases the total profitability of the two blockchains as well.

In reality, even if we assume that there are two coins with identical protocols and there is a point when they have a difficulty adjustment at the same time, the two chains will not stay synchronized, so the above model is not realistic. When the adversary appears on one of the chains, this chain becomes faster, and the next difficulty adjustment will happen earlier compared to the other chain. The adversary will therefore leave this chain before the adjustment happens on the other chain. If the adversary does not want to suspend mining, it will go to the other chain and thus make it faster and raise the difficulty there as well, but only to a smaller extent. As a result, the timing of difficulty adjustments will not be regular. Moreover, the hashpower only determines the expected length of the period between blocks; deviations from the expected value cause further irregularities. Because of these complexities, analytic description of coin hopping seems to be out of reach. In this paper, we use simulations to discover interesting phenomena related to coin-hopping.

We mention that our model is not supposed to reproduce real-life miner behavior. Indeed, miners have many options beyond coin-hopping and being loyal to a chain; furthermore, real-life miner behavior is largely influenced by coin prices, which we assume to be fixed. Our sole purpose is to analyze the profitability and implications of the coin-hopping strategy. For an interesting economic model involving variations in coin prices and risk analysis, see a recent paper by Bissias et al. [7].

3 Simulation and results

3.1 The model

Our simulations feature two blockchains C_1 and C_2, with total hashpower of 2 units. Some of the miners always mine on chain C_1, some others are always on C_2, while the third group of miners (the *adversary*) greedily chooses the chain which is more profitable at the moment. Let H_1 and H_2 denote the hashpower of loyal miners on C_1 and C_2, respectively, and let $H_a := 2 - H_1 - H_2$ denote the hashpower of the adversary. We assume that the value V_i of coin i is $H_i + H_a/2$ $(i = 1, 2)$. The motivation is that these are the coin values giving equal profitability on the two chains if the hashpower

of the adversary is divided equally. The coin values do not change during the process.

Our simulation of mining works as follows. For each group, we use a random sample from an exponential distribution depending on the hashpower of the group, and the difficulty of the chain it was mining at (D_1 or D_2) to model the time until the next block finding. So the parameters of the exponential distributions are $\lambda_1 = H_1/D_1$, $\lambda_2 = H_2/D_2$, and $\lambda_a = H_a/D_i$ when the adversary is on chain C_i. The algorithm then chooses the minimum of these values, and gives the coin to the group with the smallest one. Only the winner's block finding time is used and stored, the others are discarded; this is allowed because of the memoryless property of the exponential distribution. In the next round, another three samples are used, and again the minimum value determines the winner of the next coin. For each block finding the following are stored: which group found the block, on which chain, and how much time it took.

Difficulty is adjusted on each chain after every $M = 2016$ blocks found on that chain. The formula is $D_{new} = D_{old} \cdot \Delta / avgtime$, where Δ specifies the desired average inter-block time (10 minutes), and *avgtime* denotes the average inter-block time for the last M blocks. So the new difficulty is the old difficulty multiplied by how fast we found the blocks compared to what was planned. The profitability of the groups (P_1, P_2, P_a) is the number of blocks they find multiplied by the coin value V_i, and divided by the hashpower they spent on mining. This means that without an adversary and 1 unit of hashpower on each chain, the average profitability would be 0.1. At the beginning, both blockchains have the same difficulty and the adversary is randomly assigned to one of the chains. After each difficulty adjustment, the adversary decides whether to stay or switch to the other chain, based on the expected profitability.

3.2 Difficulty variation under different hashpower settings

To illustrate how difficulty oscillates on the two chains during a longer time period, we present three examples with different hashpower settings. Figures 1, 2 and 3 show the changes in difficulty until a total of 100,000 blocks are mined. The horizontal axis is the time; the blue line shows D_1/V_1, and green corresponds to D_2/V_2 (note that $V_1 = V_2$ on Figures 1 and 2, while $V_1 = 1.1$ and $V_2 = 0.9$ on Figure 3). A red line indicates that the adversary is on chain C_1 in that time period, otherwise it is on C_2. In all three cases, $\Delta = 10$. The difference between the examples is in the parameters H_1, H_2 and H_a.

On Figure 1, $H_1 = H_2 = 0.9$, and the adversary has a relatively small hashpower of 0.2. Here both coins have the same value, since the hashpowers are the same. From the 100,000 coins, the loyal miners on C_1 got 44698, those on C_2 got 44592, and the adversary got 10710 coins altogether (with 5373 from chain C_1, and 5337 from chain C_2). The total duration was 507302 time units, which is 1.5% more than the expected time without the adversary. The profitabilities are $P_1 = 0.0979$, $P_2 = 0.0977$, and $P_a = 0.1056$. It can be seen from the figure that the changes in the difficulty are balanced, and each chain alternates between high and low diffficulty.

On Figure 2, $H_1 = H_2 = 0.7$, and $H_a = 0.6$. Like in the previous case, both coins have the same coin value. From the 100,000 coins, loyal miners on C_1 got 34241, those on C_2 received 33156, and the adversary got 32603 (16012 from C_1,

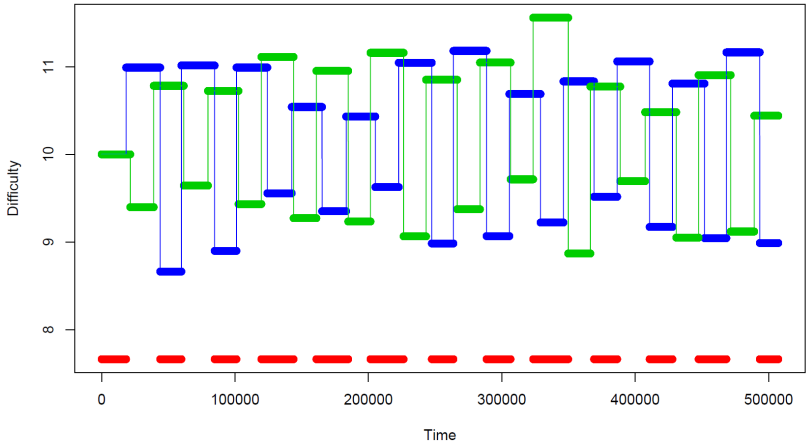

Figure 1: One run with $H_a = 0.2$, $H_1 = H_2 = 0.9$

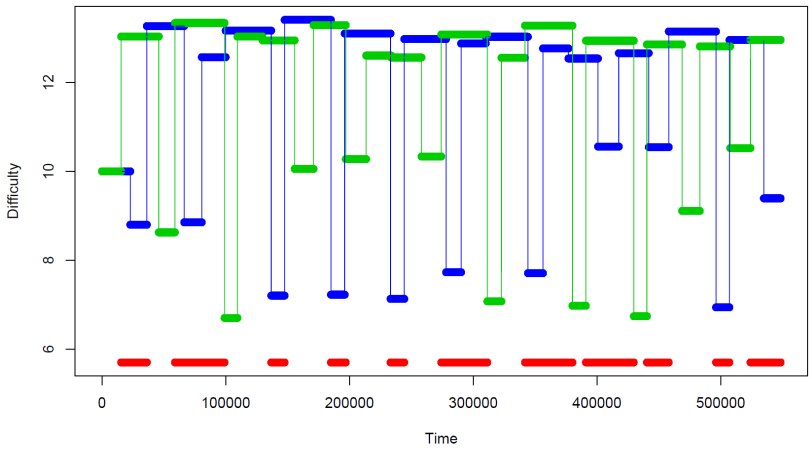

Figure 2: One run with $H_a = 0.6$, $H_1 = H_2 = 0.7$

and 16591 from C_2). This took altogether 548436 time units, which is 9.7% more than the expected time without the adversary. The profitabilites were $P_1 = 0.0892$, $P_2 = 0.0864$, and $P_3 = 0.0991$. Here not only the loyal miners, but also the adversary got less profitable compared to the situation where everyone stays on one chain. The variation of the difficulties is larger than on Figure 1, and also the inter-block times vary more widely. One can observe that while the 'high difficulty' periods are similar, there is considerable variation in the difficulty of the 'low difficulty' periods, which are also shorter.

On Figure 3, C_1 and C_2 have assymetric roles: $H_1 = 1.0$, $H_2 = 0.8$, and $H_a = 0.2$. As the chains are assymetric, the coins have different coin values: $V_1 = 1.1$ and $V_2 = 0.9$, since the coin values are determined by splitting the adversary equally between the two chains.

From the 100,000 coins, the loyal miners on C_1 got 44888, those on C_2 got 44503, and the adversary got 10609 coins altogether (5537 from C_1, and 5072 from C_2). In coin value, these are 49376.8, 40052.7, and 10655.5, respectively. The total duration

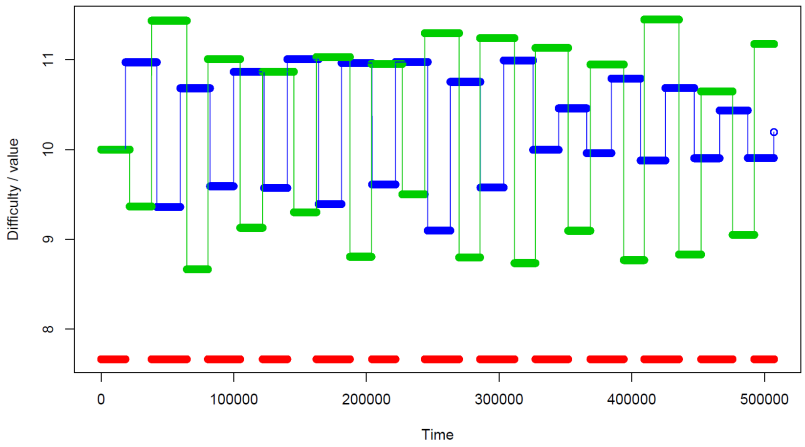

Figure 3: One run with $H_a = 0.2$, $H_1 = 1.0$, and $H_2 = 0.8$

was 507300.9 time units, which is 9% more than the expected duration without the adversary. The profitabilites were $P_1 = 0.0973$, $P_2 = 0.0987$, and $P_a = 0.105$. The figure somewhat resembles the first one, with a noteworthy difference: the chain C_2 has much higher variance in difficulty than C_1. The reason for this is the higher impact of the adversary on chain C_2 compared to C_1, due to the smaller loyal hashpower. It is also apparent that the adversary spends more time on C_1 (indicated by the red intervals), which explains why the loyal miners on C_1 are less profitable.

3.3 Profitability under various hashpower settings

The figures in the previous section were just a few examples intended to illustrate how the difficulty changes during a long mining period. Our main goal is to study the impact of the hashpower of the different groups on their profitability. To analyze this, we ran simulations with different hashpower settings, and for each setting we took the average of 100 runs. Difficulty was adjusted every 2016 blocks, and each run was 100,000 blocks long.

The results are presented on four figures on the next pages. The colored circles are the profitability values: blue marks P_1, green is for P_2, and red denotes P_a. On each figure, the total hashpower on the two chains is 2 units, and $H_2 - H_1$ is constant; the horizontal axis indicates H_a. In the first figure, C_1 and C_2 are symmetric, i.e. the non-adversarial hashpower is divided equally among them. In the three other figures, the difference between H_1 and H_2 is 0.2, 0.4 and 1.0 respectively.

The most interesting observation is that in all cases the adversary's profitability peaks at a hashpower of 0.25, which is 12.5% of the total hashpower. This peak profitability is approximately 0.105, which is 5% higher than the profitability when only loyal miners are present. If adversarial hashpower is increased beyond 0.25, then the profitability of the adversary decreases, and it reaches the original profitability of 0.1 at around 0.6 units of hashpower in most cases. The exception is the setting in Figure 7, where the difference in loyal hashpower between the two chains is very large (1 unit). Here, the adversary's profitability shrinks faster, and it returns to 0.1 at a

hashpower of 0.5. After this point, the decrease of profitability seems to be linear in the increase of hashpower in all cases.

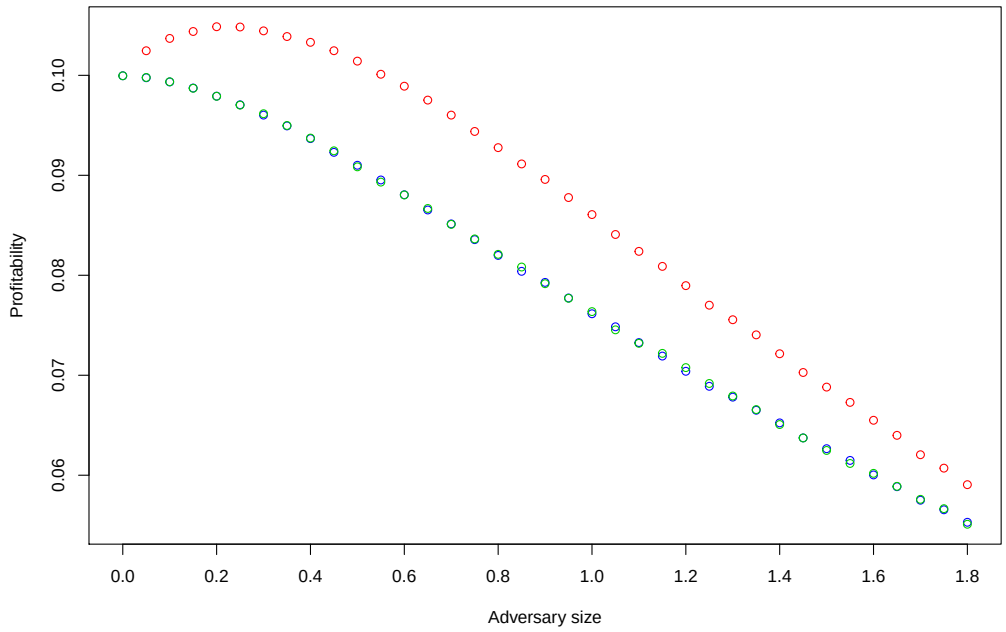

Figure 4: Profitability of the 3 groups as a function of the hashpower of the adversary (H_a), when $H_1 = H_2$.

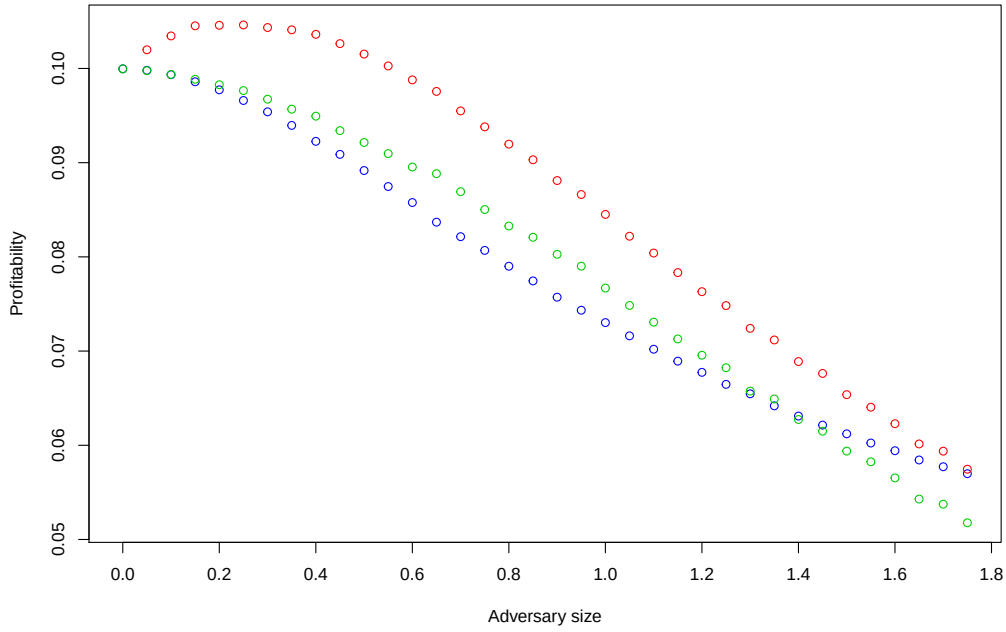

Figure 5: Profitability of the 3 groups as a function of H_a, when $H_1 - H_2 = 0.2$.

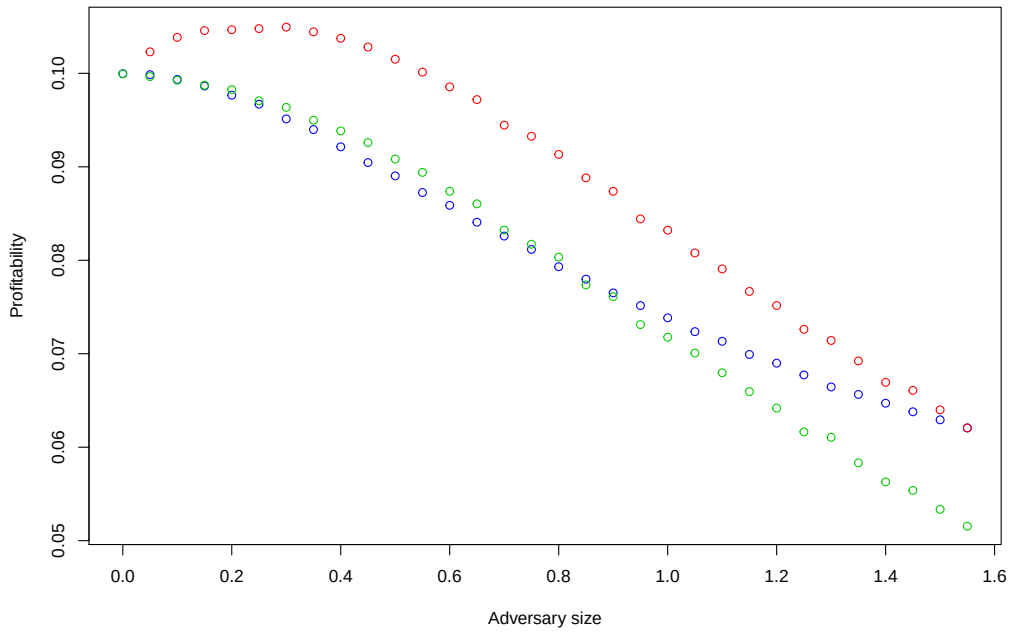

Figure 6: Profitability of the 3 groups as a function of H_a, when $H_1 - H_2 = 0.4$.

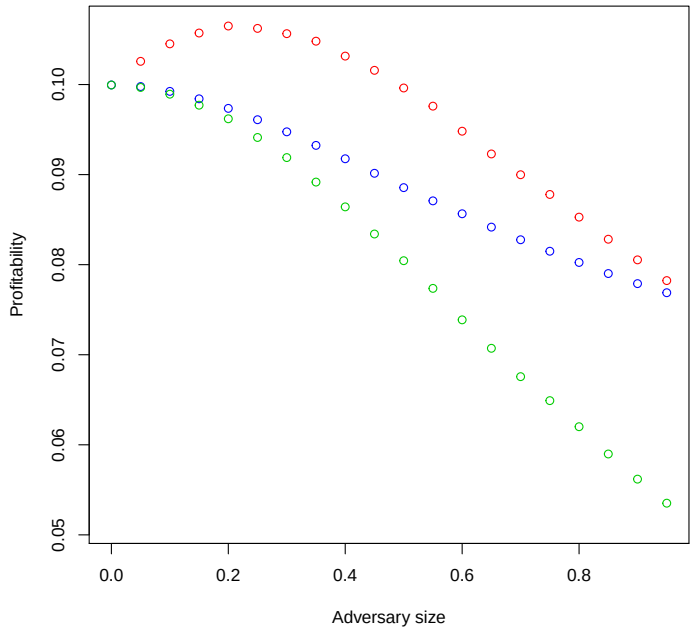

Figure 7: Profitability of the 3 groups as a function of H_a, when $H_1 - H_2 = 1$.

The profitability of loyal miners also exhibits interesting properties. It is clear that the profitability of loyal miners should decrease as the adversary gets stronger, and it is also natural that on Figure 4 the two types have the same profitability. However, the simulations for the asymmetric cases gave some surprising and somewhat counterintuitive results.

With no adversary, the profitabilities are the same, since this is how the coin values were defined. However, as we increase the hashpower of the adversary, P_1 and P_2 decrease differently. On Figures 5 and 6, where the hashpower differences are 0.2 and 0.4 respectively, loyal miners on C_2 (those with smaller hashpower) are more profitable than those on C_1 when the adversary is not too large. In both cases, the two types have the same profitability again when the ratio of H_1 and H_2 is approximately 2 (On Figure 5, this is at $H_a = 1.4$, $H_1 = 0.4$ and $H_2 = 0.2$, while on Figure 6, the parameters are $H_a = 0.8$, $H_1 = 0.8$ and $H_2 = 0.4$). We see no intuitive explanation for this balance of profitability at the hashpower ratio of $2 : 1$, so this phenomenon requires further study.

After this threshold of $H_1/H_2 = 2$ is reached, further increase of H_a results in a higher rate of decrease for P_2 than for P_1 (remember that $H_1 - H_2$ is constant on each figure, so H_1/H_2 increases with the increase of H_a). This explains why the profitability functions do not cross on Figure 7: in that setting, the ratio of H_1 and H_2 is always larger than 2.

Another interesting phenomenon in the asymmetric cases is that P_1 approaches P_a as H_2 shrinks to 0. This can be explained as follows. The impact of the adversary is huge on chain C_2, as H_a is very high compared to H_2. Hence, the adversary's presence on C_2 speeds up blockfinding so much that D_2/V_2 becomes very large at the next difficulty adjustment. At that point, the adversary switches to C_1, and the loyal miners on C_2 need a lot of time to mine the 2016 blocks until the next difficulty adjustment on C_2. Meanwhile, the difficulty increases on C_1, but D_1/V_1 can still remain smaller than D_2/V_2 because $V_1 > V_2$. Consequently, the adversary is mining on C_1 most of the time, and it has approximately the same profitability as the loyal miners there.

4 Conclusion

In case of two identically functioning blockchains, a coin-hopping adversary can earn higher income compared to the miners that are loyal to one particular blockchain. According to our simulations, the benefit of the coin-hopping adversary is highest if it has approximately 12% of the hashpower, in which case it attains approximately 5% more profit than without coin hopping. An adversary using a coin hopping strategy not only gets higher profits, but also hurts the incomes of the other miners. If more than 30% of miners are coin-hopping, then the result is detrimental to all miners, including the coin-hoppers.

We also examined the profits of loyal miners on both chains. Our simulations suggest that those on the chain with more loyal miners are actually worse off, provided they have less than twice the hashpower of the loyal miners on the other chain. How-

ever, if the ratio of loyal hashpower on the two chains is larger than 2 : 1, then coin-hopping is more damaging to loyal miners on the chain with less hashpower.

Our results were obtained under the assumption that the value of each coin is proportional to the initial hashpower on the chain (the loyal miners plus half of the adversary), and it does not change during the process. Further research is needed to get a more complete picture on the role of coin values. As it is pointed out in a recent paper by Bissias et al. [7], miners cannot spend their coins immediately, so they have to deal with risk due to the high volatility of cryptocurrencies. Bissias et al. develop an economic model based on risk tolerance that adequately explains real-world miner behaviour.

We also plan to study the consequences of coin-hopping when the two blockchains have different difficulty adjustment methods – as real-world examples like Bitcoin Cash show, alternative methods may have a huge impact on the effects.

Acknowledgement

We would like to thank Ádám Antal for the helpful discussions. The research was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

References

[1] D. Meshkov, A. Chepurnoy, M. Jansen, *Revisiting Difficulty Control for Blockchain Systems*, In: Data Privacy Management, Cryptocurrencies and Blockchain Technology. ESORICS 2017, DPM 2017, CBT 2017. LNCS, volume 10436 (2017), 429–436, `https://doi.org/10.1007/978-3-319-67816-0_25`

[2] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008, `https://bitcoin.org/bitcoin.pdf`.

[3] BitcoinABC, *Difficulty Adjustment Algorithm Update*, `https://www.bitcoinabc.org/2017-11-01-DAA/` as of 2018.06.28

[4] J. Song, *Bitcoin Cash Difficulty Adjustments*, `https://medium.com/@jimmysong/bitcoin-cash-difficulty-adjustments-2ec589099a8e` as of 2018.06.28

[5] I. Eyal and E. G. Sirer. *Majority is not enough: Bitcoin mining is vulnerable.* International conference on financial cryptography and data security. Springer, Berlin, Heidelberg, 2014.

[6] M. Carlsten, H. Kalodner, S. M. Weinberg, *On the instability of bitcoin without the block reward*, Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, 2016.

[7] G. Bissias, B.N. Levine, D. Thibodeau, *Using Economic Risk to Model Miner Hash Rate Allocation in Cryptocurrencies*, arXiv preprint (2018), `https://arxiv.org/abs/1806.07189`