CS 6466 2Chainz

Email:	wdaviau@protocol.ai
CUNet ID:	wtd37
Name:	Wyatt Daviau
Date:	March 25th 2018

1. 2Chainz – Miner Dynamics Across Blockchains

Items wrapped in * * need significant further thought Big goals of paper:

1. Exploring fundamentals
   (a) Define the model we used, communicate its strengths and shortcomings
   (b) Come up with a reasonable "greedy" mercenary miner strategy. Justify why miners would want to run this strategy and how it works.
   (c) Explore the strategies implications from first principles. Plot expected unstable regions. Present a diagram of the four relevant quantities and how their ordering and difficulty adjustment creates sustained oscillations of hash power between the chains
   (d) Describe the simulation framework and methodology. Present simulation results. Show the empirical breakdown of the parameter space into regions. Quantify how the smaller chain's hash power distribution changes over the space. Quantify the profitability of the scheme as compared to staying loyal to the first chain.
2. Extending Miner strategies
   (a) *specify a strategy and discuss whether it is optimal*
   (b) *Compare to greedy with simulations*
3. Blockchain Protocol Gaming Greedy Mercenary Miners
   (a) Define slight updates to the model. The protocol is trying to optimize for hashrate on the chain and reducing the number of coins minted compared to the other chain.
   (b) Describe the update protocol and motivation for it coming from the diagrams of A.2
   (c) *Do some analysis of the optimality of this protocol, the gained hashrate for a target minting threshold as a function of parameters*
   (d) *Simulations that run this protocol and record the minting savings and hashrate purchase*
4. D. Are miners following the mercenary strategy when they could be? 1. plot of thresholds over time highlighting regions where the miners could have switched. Point out and places where it looks like they do switch

2. Abstract

We model the scenario where two blockchains have the same proof of work hash function and therefore miners can switch among chains at will. We devise a principled simple strategy for choosing among competing blockchains that miners can follow. We demonstrate that this strategy is profitable throughout a large region of the considered parameter space and classify different regions of the parameter space based on the behavior of miners following this profitable strategy. We present an analysis of historical BTC / BCH data in light of our

Furthermore we investigate what an optimal switching strategy would look like if miners can make reasonable predictions about the future. We also propose a candidate strategy for a blockchain designed to compete with an existing chain for hash power without flooding the market for its native token by minting coins at an excessive rate.

3. Introduction

The recent Bitcoin Cash (BCH) fork of the Bitcoin (BTC) blockchain serves as an example of a relatively powerful fork of a blockchain network. Such forks naturally lead to multiple blockchain networks using the same proof of work function. In such a world "mercenary" miners can potentially increase profits by dynamically switching their hash power across chains. Recent events have shown that this can have detrimental effects on the networks involved [1] due to drastic changes in profitability among chains and a slow response to large changes in hashrate that trace back to details in bitcoin's difficulty adjustment algorithm.

The goal of our analysis is to study the scenario in which miners are willing to switch between two blockchains that run the same difficulty adjustment protocol to maximize their net profits. We define a plausible parametrization of this system and then study potential chain-switching strategies that miners might employ in an attempt to increase their profits. We begin by trying to understand the profitability over time of what we deemed the "greedy mercenary" approach, and the regions in which this leads to stable and unstable hash power allocations across chains. From first principles we come up with a simple metric mercenary miners could use to decide on the best chain and end up rediscovering the so-called "Difficult Adjustment Reward Index" or DARI that miners actually use during operation.

[Ahaan discussion of further optimal algorithms or optimality of greedy mercenary]

We model the system consisting of two chains both running BTC's difficulty adjustment protocol but sweeping different reward fractions, "loyal" hashrates and mercenary hashrate. Based on our model we introduce a simple visualization for the stability characteristics of different points in the parameter space and an associated prediction function determining whether a point in the parameter space will be unstable. We built simple simulation and analysis frameworks for generating and interpreting data representing execution of these systems in the presence of mercenary miners following the greedy strategy. Leveraging these frameworks we present a view into the different stability regions of the parameter space and reach the conclusion that the greedy mercenary mining strategy is profitable throughout a wide section of the parameter space. We confirm our theoretical prediction of the instability region concluding that in the BTC / BTC difficulty adjustment case the region of instability in the parameter space is significant and [impacts the transaction throughput of both systems TODO analysis function and figure quantifying this].

[Arnesh + Wyatt TODO BCH / BTC, BCH / BCH simulation results would be a great addition to this paper and seems within reach with one week remaining]

[Arnesh + Ahaan analysis of historical data sets, maybe this should be skipped as DARI is already plotted by fork.lol?]

4. Model and Assumptions

We make many simplifying assumptions when modeling this system. The goal is to make enough assumptions to keep analysis tractable while at the same time parameterizing the problem in a way that gives insight to real systems of competing blockchains, and paves the way for more advanced modeling. The following are global system assumptions

1. There are two chains in the system, chain 1 and chain 2. Both chains use the same proof of work and difficulty adjustment algorithm as the Bitcoin protocol [TODO when BCH difficulty is added in this will change]

2. Each chain rewards miners with a coinbase transaction that pays out tokens to the miner that solves the block puzzle. Additionally miners are rewarded by the transaction fees tied to the transactions serialized in the blocks. We assume that the value of the total reward (in, say USD) of chain 1 is a constant f_1 and the value of the token of chain 2 is a constant f_2. We do not account for variations in transaction fees or reductions in the coinbase payout in this analysis. Only the ratio of these two parameters $\frac{f1}{f2}$ is relevant in our analysis.

3. We model the total hash rate of the system as H (hashes / second) and assume that it does not change over the period of analysis in question.

4. Fractions of the hashpower belong to three different entities, α, β_1, β_2. $\alpha + \beta_1 + \beta_2 = 1$.

5. We assume a fraction α of the hash power, from here on out called the mercenary miners, is willing to switch between chains in order to make a profit. In the case of our analysis of the Greedy Mercenary Mining strategy we do NOT need to make the assumption that the hashpower α belongs to a single pool. This is elaborated further in the following section.

6. β_1 and β_2 corresponding to miners who are loyal to chains 1 and 2 respectively. These hashpower fractions do not move from their respective chains regardless of the profitability of switching. We need not consider the hash power of β_i as belonging to a single pool.

7. The difficulty of each chain is adjusted every b blocks. The time it takes for b blocks to be mined in epoch $i = \Delta_i$. To calculate the difficulty adjustment the protocol specifies that: $d_{i+1} = \frac{d_i \cdot b \cdot t_{\text{target}}}{\Delta_i}$, as in bitcoin. The target is 600s.

8. We assume the system begins in a steady state where the difficulty of each chain reflects the initial allocation of the hash power. The mercenary miners begin by mining on chain 1. Throughout the analysis chain 1 is given the asymetric advantage that $\frac{f1}{f2} \leq 1$

4.1. Calculating the initial difficulty

From assumption (8) above we can calculate the initial difficulty, which is an initial condition needed by the simulation framework. We include the derivation here because the discussion is relevant for the subsequent derivation of the switching criterion for the greedy mining strategy.

The following discussion applies to both chains and examines chain 1 without loss of generality. By our steady state assumption that the mining epoch before the period of history in which our analysis takes place lasted exactly $(600s)(b)$. The number of hashes tried until a block is successfully mined follows a geometric distribution and by the properties of this distribution the inverse of the probability of one success is equal to the mean number of hashes until success. Given our steady state condition and initial allocation we have the following relation

$$600H(\alpha + \beta_1) = \frac{1}{p}$$

Where p is the probability of one hash solving the block puzzle. Recall that the difficulty of a chain is given by

$$\frac{F_{\max}}{F}$$

where F is the threshold value below which a block with the given hash solves the puzzle. As the range of the hash function used in the chain puzzle is binary strings of 256 characters, $p = \frac{F}{2^{256}}$ and therefore

$$\begin{aligned} 600H(\alpha + \beta_1) &= \frac{2^{256}}{F} \\ 600H(\alpha + \beta_1)F_{\max} &= \frac{2^{256}F_{\max}}{F} \\ \frac{600H(\alpha + \beta_1)F_{\max}}{2^{256}} &= d_{10} \end{aligned}$$

where d_{1i} is the difficulty of chain 1 in the ith adjustment period of chain 1. Factoring out constants we have

$$\begin{aligned} \mathcal{D} &= \frac{HF_{\max}}{2^{256}} \\ d_{10} &= 600\mathcal{D}(\alpha + \beta_1) \\ d_{20} &= 600\mathcal{D}(\beta_2) \end{aligned}$$

5. The Greedy strategy

What happens if the switching miner decides to maximize for profit with a simple greedy algorithm, that is, an algorithm making no predictions about the future hash allocations or reward schedules of the two chains? The sensible way to do this is to compare the expected rate of reward on both chains at any time. We now show that the simplified form of the expected reward rate at difficulty adjustment period i is simply

$$\frac{f_1}{d_i} \cdot \alpha$$

. Let $\mathcal{E}_{ij}[T]$ be the expected time to mine a block on chain j in difficulty period i. The expectation is taken over the randomness in solving hash puzzles while assuming the difficulty d_i is known. To start we assume that there is only one mercenary mining pool with hash power α, and later show how to relax this assumption.

As we saw in the last section the expected time to find a block can be related to the chain difficulty:

$$\begin{aligned} \mathcal{E}_i[T](\alpha + \beta_j)H &= \frac{1}{p} \\ \mathcal{E}_i[T](\alpha + \beta_j)H &= \frac{2^{256}}{F_i} \\ \mathcal{E}_i[T](\alpha + \beta_j)H &= d_{ij}\frac{2^{256}}{F_{\max}} \\ \mathcal{E}_i[T] &= \frac{d_{ij}}{\mathcal{D}(\alpha + \beta_j)} \end{aligned}$$

Now we can calculate the expected profit to mine on chain j during difficulty period i as follows

$$\begin{aligned}
&\frac{\text{reward per block}}{\text{expected time per block}} \cdot (\text{expected winning fraction}) \\
&= \frac{f_j}{\mathcal{E}_i[T]} \cdot \frac{\alpha}{\alpha + \beta_j} \\
&= \frac{f_j \cdot \mathcal{D}(\alpha + \beta_j)}{d_{ij}} \cdot \frac{\alpha}{\alpha + \beta_j} \\
&= \frac{f_j \cdot \mathcal{D}\alpha}{d_{ij}}
\end{aligned}$$

As $\mathcal{D}$ and α appear in the profit rates of both chains, the greedy mercenary mining strategy checks the difficulty of the current period on each chain d_1 and d_2 and mines on chain

$$\arg\max \frac{f_j}{d_j}$$

In our simplifying model the fractional reward does not change and therefore the miner will only benefit from switching when one of the chains adjusts its difficulty. When one profit rate is strictly greater than the other the mercenary miner does not stand to benefit in the short term from allocating some hashpower to the less profitable chain so all α goes to the most profitable chain. In the case both profit rates are equal the greedy mercenary miner chooses a chain at random.

A key observation regarding the profit rate is that while changes in the block reward and difficulty values influence which chain is most profitable, the addition or subtraction of other mercenary hash power within the same period does not affect the outcome on expectation. Intuitively this is because the expected rate of block emission changes exactly inversely proportionally to the change in the expected winning fraction of a given mining pool. This is encoded in the algebra above by the appearance of $(\alpha + \beta_j)$ in both the numerator and denominator. As a consequence the choices of one greedy mercenary miner is not affected by the choices of another, at least in our simplifying fixed reward model. This means that we can model α as the hash power of ALL mercenary miners, and that our results are easier to apply to this more general case.

5.0.1. Application to real mining

After deriving our simple greedy strategy independently, it turns out that our algorithm is apparently used by real-life mercenary miners, or as they are sometimes called "chain-hoppers". Our switching criterion is known as the Difficulty Adjustment Reward Index by miners in the BCH community and is tracked by at least one BCH analytics site [2].

5.0.2. Theoretical Analysis

Given the mechanism of the greedy mercenary mining strategy and the properties of the protocol difficulty, one useful characterization of a two chain system is the ordering of the four values:

$$\frac{f2}{\alpha + \beta_2}, \frac{f2}{\beta_2}, \frac{f1}{\alpha + \beta_1}, \frac{f1}{\beta_1}$$

In a rough sense the value $\frac{fj}{\alpha+\beta_j}$ can be thought of as the least profitable expected profit rate and $\frac{fj}{\beta_j}$ the most profitable expected profit rate on chain j. This is because the difficulty on chain 2 comes from a distribution with a mean proportional to the average hash rate on the previous interval and the greatest and least possible such average hash rates are $\alpha + \beta_j$ and β_j respectively. In the case where miners switching in the middle of one of chain j's adjustment periods due to a profitability adjustment triggered by the other chain adjusting its difficulty, then the average hash rate, and therefore expected difficulty value on chain j in the next period will be between these two extreme values.

6. Simulations

To confirm our hypotheses with greater confidence we built a small simulation and analysis framework in under 600 lines of python. This simulation framework generates and saves data sets recording data for each difficulty adjustment period. Sweeps are made to record this data throughout the parameter space. The analysis framework reads data files and applies filters across subsections of data to flexibly generate values suitable for plotting from the raw data. These techniques provided us with clear visualizations of the profitability and instability of different regions of the parameter space.

6.1. Methodology

Our code is located on github at: https://github.com/ZenGround0/2Chainz

6.1.1. Simulation Framework and Sweeps

The simulation framework generates a blockchain history from input parameters α, β_j, f_j, d_{0j}, b and t_{target}. It mines blocks on each chain with a time drawn from an exponential distribution, records difficulty adjustments and places the mercenary hash power on the chain given by the strategy defined in section 5. Each period of time between switching events is recorded as 2 4 tuples. Each chain records (time, blocks mined, difficulty, hash power on chain)

We broke our parameter space into 3 standard degrees of freedom: α, β_1, and $\frac{f2}{f1}$. We swept price fractions and β values at precision of 0.01 and ran each such sweep with alpha values of 0.01, 0.05, 0.1, 0.2, 0.3. We began by running 5 trials for each point and running for 100 difficulty adjustments. Datasets are persisted to disk as csv files and each such sweep generates about 3 GB of data.

We performed a simple statistical analysis on the mean and standard deviations of the oscillation count data set to measure how much greater a sample size we should take to increase our confidence in the results. The test showed that apart from a small subset of points well understood to be highly variant, 5 trials was more than enough to be 90% confident in our samples. Furthermore only 5 more trials would be sufficient to get to 90% confidence in the small subset of high variance points.

[TODO not necessary but we could potentially justify far fewer difficulty adjustment periods to get meaningful data (expected profit, oscillations / hash power residency). This would take some initial work but would result in much faster and probably more principled simulations. Initial work would center around plotting out individual runs and looking for convergence patterns. Alternatively coming up with a map function measuring a meaningful notion of convergence in the parameter space and running a normal plot

job. I am confident 100 runs is well past convergence, and also a little much given that it is 2 years of acutal time]

6.1.2. Analysis Framework

We wrote a data processing and plotting script that leverages a simple map reduce framework to easily compose operations and make generation of plots easy to reason about. We built a simple map reduce framework: the map function processes the data of a single trial's history and the reduce function aggregates together all of the values of all trials occurring at the same point in the parameter space. The plots generated are of the entire parameter space sweep. Each plot represents one value of α, i.e. the combined hash rate of all mercenary miners. The x-axis measures β_1, the hash rate loyal to the more powerful chain 1, and the y-axis measures the fraction $\frac{f2}{f1}$ between the values of 0 and 1.

[TODO plots of individual runs. This is a precursor for the previous TODO. It is not necessary but might help present results + is necessary for reducing the number of difficulty adjustments in simulations in a principled way. Also this is a pretty simple endeavor 2 hours and it could shave lots and lots of time from simulations this week]

6.2. Results

7. Other Mercenary Strategies

[TODO Ahaan]

8. Blockchain Protocol Leveraging Mercenaries

[TODO Wyatt]

9. Historical Analysis of Data

[TODO Arnesh]

9.1. References

[1] https://bitcoinandtheblockchain.blogspot.co.uk/2017/08/chain-death-spiral-fatal-bitcoin.html?m=1
[2] https://fork.lol/reward/dari/btc