Lab Project: Selfish mining

Ayelet Sapirshtein

Abstract

Mining is the process of adding blocks of transaction to the Bitcoin's block chain. The block chain serves to confirm transactions in the network. Nodes use the block chain to verify the legality of transactions. Each block contains a proof of work to be considered valid. This proof of work is verified by other nodes in the network.

Mining is designed to be difficult so that the frequency of blocks found would be low and constant. Therefore if block creation rate decreases, the difficulty of block creation descends and vice versa. For this reason we are interested in the percentage of blocks we build in the network, instead of the quantity of blocks.

The primary purpose of mining is to reach security in the Bitcoin network. Mining is also used to introduce Bitcoins into the system: When a miner discovers a block, the discoverer is paid for any transaction fees and newly created coins.

A selfish mining attack is a strategy in which a mining pool selectively publishes or withholds blocks, instead of publishing blocks to the network immediately when found. In this strategy the attackers may sacrifice their own revenue but also often publish many blocks all at once and thus are forcing the rest of the network to discard blocks and lose revenue. This does reduce the attackers revenue in the short term, but it reduces everyone else's revenue even more. Therefore it increases the percentage of blocks the attackers build in the network, that means increasing the attackers rewards.

The attack strategy is as follows: The attacker keeps its own private chain, which is separate from the public chain that the rest of the network works on. At the beginning, the private chain and the public chain start out the same. The attacker always mines on the private chain and keeps any blocks that he finds private. If the public chain is longer than the private chain the attackers adopts the public chain and starts a new private chain that starts out the same as the public chain. In other cases, the attackers publish a portion of their private chain, In order to force the rest of the network to discard blocks, and accept the private chain blocks. This strategy is described extensively in the paper ”Majority is not Enough: Bitcoin Mining is Vulnerable”.

In this project we introduce a new strategy that improves the selfish mining strategy for the selfish attackers. We describe an MDP that models the scenario of honest miners that act according to the protocol and attackers who keep their own private chain, as in the original paper, and choose to take action that will yield them the most gains.

The optimal strategy for the attackers depends on their computing power in the network and their ability to convince honest miners to mine on their blocks chain instead of the honest chain when they see the same length chains. Therefore we compute the best deterministic selfish mining strategy for different parameters of the network, and for each parameter we get the optimal strategy for them by solving the MDP problem.

1 Description of the MDP

2 Definition of Variables

- α : Attackers relative computing power
- γ : The relative part in average of the honest that the attackers convince to mine on their blocks chain after Match action.
- $\rho = \lim_{t\to\infty} \frac{\text{Attacker rewards}}{\text{Honest rewards+Attacker rewards}}$ under the optimal strategy
- S_a: Number of attackers blocks, from the network split.
- S_h: Number of honest blocks, from the network split.
- Last build : Who built block in the last round a or h .
- Conviction: What fraction of honest workers on the honest chain : 1 or $1-\gamma$
- S : States in the state machine are of the form $< S_a, S_h,$Last build, Conviction$>$
- $A : S \longrightarrow$ Subgroup of actions : Returns the possible actions according to current state.
- $P : S \times Action \longrightarrow$ Probability vector of next state .
- $imr_a(< s_s, s_t, a >)$: The immediately reward for the attackers, if they moved from s_s to s_t by action a.
- $imr_h(< s_s, s_t, a >)$: The immediately reward for the honest, if they moved from s_s to s_t by action a.
- $R : S \times S \times Action \longrightarrow Reward$.
- $R(< s_s, s_t, a >) = imr_a(< s_s, s_t, a >) \cdot (1-\rho) - imr_h(< s_s, s_t, a >) \cdot (\rho)$

From the definition of R and ρ we can see that in this MDP the optimal strategy gives average reward 0.

3 A definition of the possible actions

1. Wait for formation of a new block in the network
2. Adopt the honest chain
3. Match, the attackers releasing a blocks chain with length as the honest blocks chain length.
4. Override the honest chain by releasing a chain of length K+(the length of the honest), $1 \leq K \leq 20$.

4 A description of the possible actions according to current state

The state machine that fits to modeling the problem is infinite. In this project we prefer to work with a finite state machine, which is close to the true model. Under the assumption that the attacker has less computing power than the honest miners it is unlikely that the attacker will build a long chain that is longer than the honest chain.In addition, we assume that if the honest chain is longer than the attacker, in most cases it is better for the attacker to adopt the honest chain.Therefore in most of the situations we do not reach the situation that $S_a < S_h$ mostly when there is a large gap. For that reason we let ourselves cut the state machine when Sh or Sa are big.

4.1 Boundary states

If $S_h = 20$ the only allowed action is Adopt (Action 2), else if $S_a = 20$ the only allowed action is Override (Action 4) by 1.

4.2 The honest leads $S_a < S_h$

If $S_a < S_h$ the only allowed actions are Wait (Action 1),Adopt (Action 2).

4.3 The attackers leads $S_a \geq S_h$

- Action Wait (Action 1) is allowed.
- Action Adopt (Action 2) is allowed if and only if the conviction is 1.We can convince ourselves that if conviction is $1 - \gamma$ than $0 \leq S_a - S_h$, therefore Adopt (Action 2) is not the recommended action.
- Action Match (Action 3) is allowed if and only if the last built is h and $0 \leq S_a - S_h$. If the honest are the ones who built the last block,it means that the last block in the chain of honest was just created. Therefore there are nodes that have not heard about it yet, and they are still vulnerable to be convinced,by another chain with identical length.In that case we can do the Match (Action 3) action.Otherwise,the honest are not the ones who built the last block, it means that the attacker built the last block, and that all the nodes in the network have heard about it,and then we can not convince anyone by another chain with identical length.
- Action Override (Action 4) is possible for every state that satisfies $1 \leq K \leq S_a - S_h$ and $1 \leq S_a$.

5 A description of the actions effect according to the current state

The transitions between the states under action is possible under the condition of block formation.In every transition one block is created.In this way caused every action takes the same amount of time.

- Action Wait (Action 1)
  When a block of the main chain created, all the honest node start to work on the main chain.The attacker loos his conviction advantage, $Conviction = 1$.

$NS \backslash S$	$(S_a, S_h, a, 1-\gamma)$	$(S_a, S_h, \cdot, 1)$
$(S_a+1, S_h, a, 1-\gamma)$	$(\alpha, 0)$	
$(S_a - S_h, 1, h, 1)$	$((1-\alpha)\gamma, (1-\rho)\cdot S_h)$	
$(S_a, S_h+1, h, 1)$	$((1-\alpha)(1-\gamma), 0)$	
$(S_a+1, S_h, a, 1)$		$(\alpha, 0)$
$(S_a, S_h+1, h, 1)$		$((1-\alpha), 0)$

- Action Adopt (Action 2)

$NS \backslash S$	$(S_a, S_h, \cdot, 1)$
$(1, 0, a, 1)$	$(\alpha, -\rho \cdot Sh)$
$(0, 1, h, 1)$	$((1-\alpha), -\rho \cdot Sh)$

- Action Match (Action 3)

$NS\backslash S$	$(S_a, S_h, h, 1))$
$(S_a + 1, S_h, a, 1 - \gamma)$	$(\alpha, 0)$
$(S_a - S_h, 1, h, 1)$	$((1 - \alpha) \cdot \gamma, (1 - \rho) \cdot Sh)$
$(S_a, S_h + 1, h, 1)$	$((1 - \alpha) \cdot (1 - \gamma), 0)$

- Action Override (Action 4)
  $K = (Numberofreleasedblocks) - Sh$

$NS\backslash S$	$(S_a, S_h, \cdot, \cdot)$
$(S_a - S_h - K + 1, 0, a, 1)$	$(\alpha, (1 - \rho) \cdot (S_h + K))$
$(S_a - S_h - K, 1, h, 1)$	$((1 - \alpha), (1 - \rho) \cdot (S_h + K))$

6 Implementation

In order to solve the problem, we decided to represent the problem as an MDP linear program.
A description of the LP

$$\begin{aligned} &\text{minimize} && \sum_{j \in S} v_j \\ &\text{subject to} && \sum_{j \in S} (\delta_{ij} - P(<i, j, a>)) v_j \geq 0, && for\ every\ (i, a) \in (S \times A) \\ &\text{subject to} && v_i + \sum_{j \in S} (\delta_{ij} - P(<i, j, a>)) \cdot u_j \geq \sum_{j \in S} P(<i, j, a>) \cdot R(<i, j, a>), && for\ every\ (i, a) \in (S \times A) \end{aligned} \tag{1}$$

for every v_i such that i is a strongly connected state in the graph, the value of the variables v_i is the average reward in the MDP under the optimal policy.
One can see that the average reward of the this MDP is:
$Let : atr = Attacker\ rewards$
$hr = Honest\ rewards$
$Average\ reward = (atr) \cdot (1 - \rho) - (hr) \cdot \rho.$
From the definition of ρ we can conclude that the average reward of the MDP should be 0.
Since $\rho = \lim_{t \to \infty} \frac{atr}{hr + atr}$
But we don't know the rho value and we want to find it. The way we find ρ is by using binary search.

6.1 Binary search

We know that ρ should be in the range $[0, 1]$
Let us define the guessed ρ to be :$0 <=' \rho <= 1$.
if $'\rho < \rho$ then the average reward of the MDP will be greater than 0.
Since $'\rho < \rho = \frac{atr}{hr + atr}$
$0 < (atr) \cdot (1 -' \rho) - (hr) \cdot' \rho$
Similarly if $'\rho > \rho$ the average reword of the MDP will be less than 0.

Therefore we guess ρ value $'\rho$ Solve the MDP for $\rho =' \rho$.
By looking at the value of the average reward of the aforementioned we can conclude the average reward value if $\rho <' \rho$ or $\rho >' \rho$
Therefore we can make a binary search for ρ value, and find approximation to ρ .

6.2 Optimal strategy

In order to find an optimal strategy we look at the LP Solution of the MDP with ρ value $'\rho$ such that $|'\rho - \rho| < \epsilon$.
By looking at the equations that the constraint is tight:

$$v_i + \sum_{j \in S} (\delta_{ij} - P(< i, j, a >)) \cdot u_j = \sum_{j \in S} P(< i, j, a >) \cdot R(< i, j, a >)$$

Through these equations we get that a is the optimal action in state i. In this way we will get the optimal strategy.

7 Graphical results

The boundary states are $s_a = 20$ and $s_h = 20$

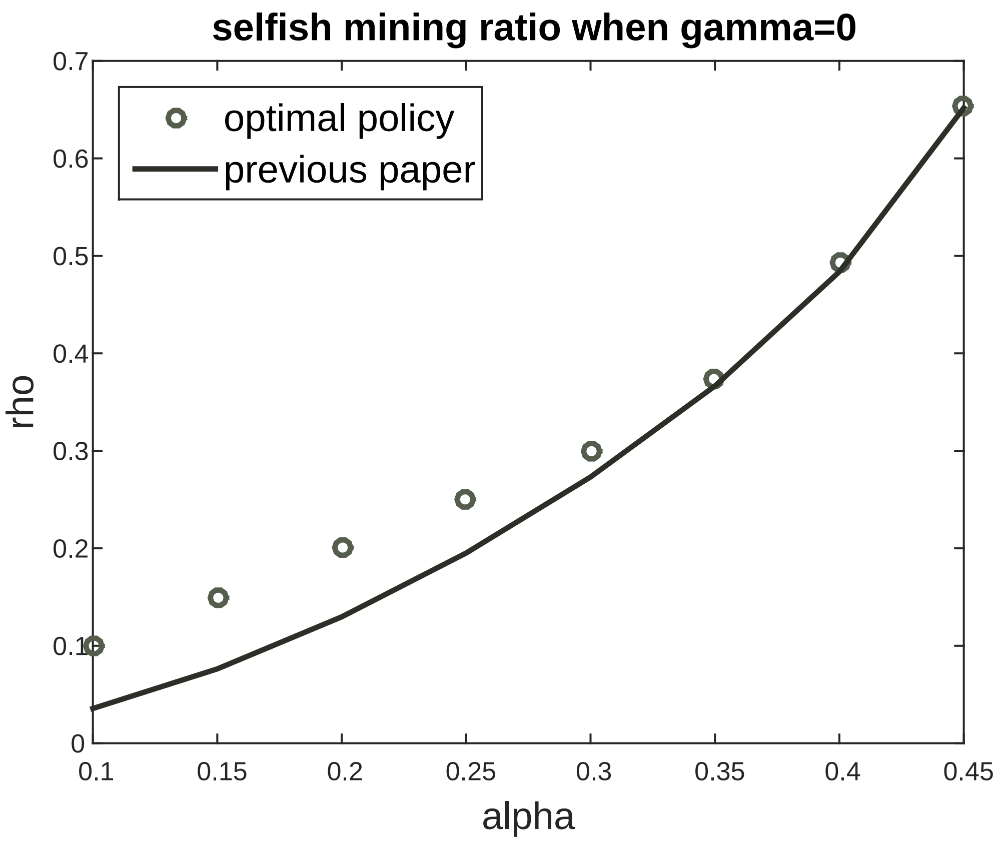

Figure 1:

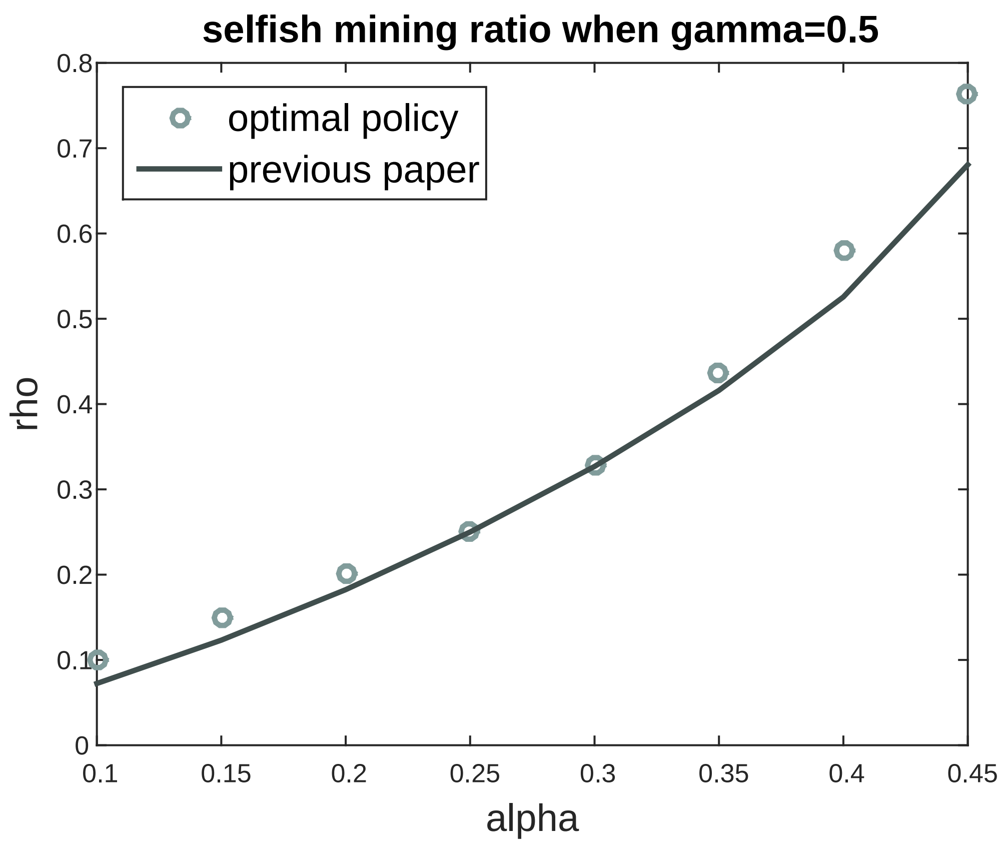
selfish mining ratio when gamma=0.5
optimal policy
previous paper
rho
alpha
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45

Figure 2:

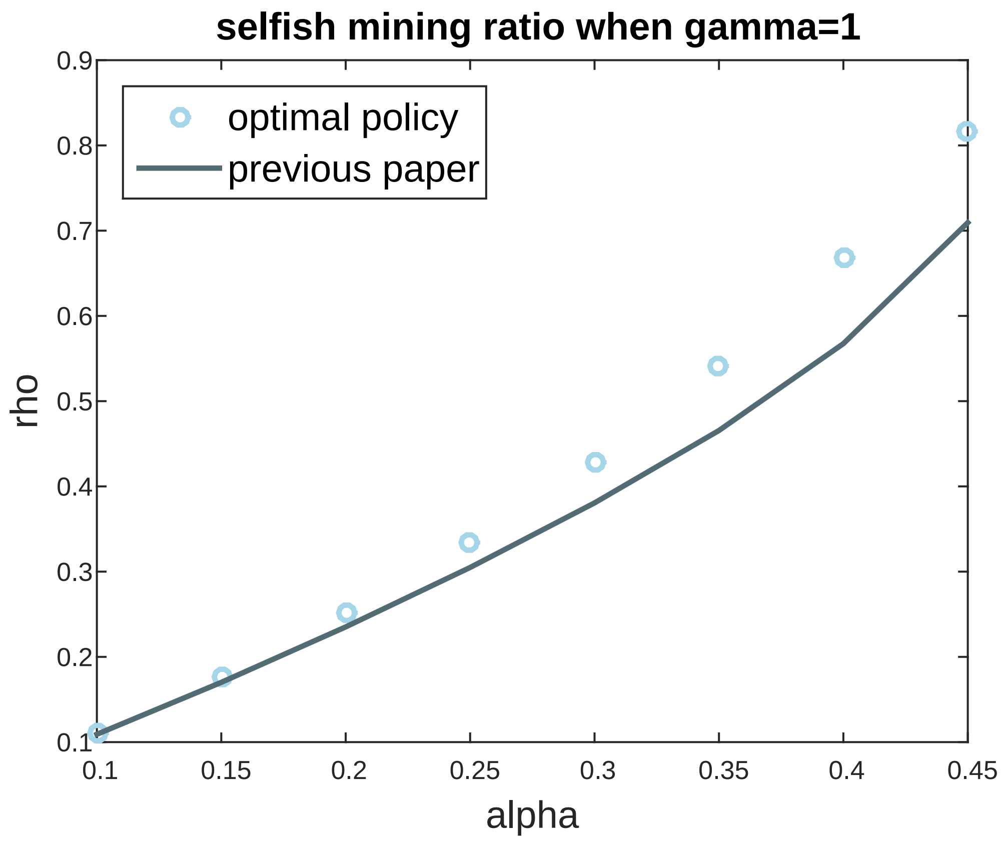
selfish mining ratio when gamma=1
optimal policy
previous paper
rho
alpha
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45

Figure 3:

8 Description of strategy

A description strategy we found for $\alpha = 0.4$, and boundary states $s_a = 20$ and $s_h = 20$. More details can be seen in the attachments files.

8.1 $\gamma = 0, \alpha = 0.4$

Result: $\rho = 0.4931848$

Suggestion:
Denote $gap := s_h - s_a$. Then,

- Adopt whenever

$$\begin{aligned}
&(s_h = 1 \ \wedge \ gap = 1) \ \vee \\
&(2 \leq s_h \leq 4 \ \wedge \ gap = 2) \ \vee \\
&(6 \leq s_h \leq 8 \ \wedge \ gap = 3) \ \vee \\
&(10 \leq s_h \leq 11 \ \wedge \ gap = 4)
\end{aligned}$$

- Override whenever

$$s_h \geq 1 \wedge gap = -1$$

- Wait in any other case

8.2 $\gamma = 0.5, \alpha = 0.4$

Result: $\rho = 0.5625$

- Adopt whenever

$$\begin{aligned}
&(1 \leq s_h \leq 2 \ \wedge \ gap = 1) \ \vee \\
&(4 \leq s_h \leq 5 \ \wedge \ gap = 2) \ \vee \\
&(7 \leq s_h \leq 10 \ \wedge \ gap = 3) \ \vee \\
&(12 \leq s_h \leq 13 \ \wedge \ gap = 4)
\end{aligned}$$

- Override whenever

$$\begin{aligned}
&LastBuild = a \wedge s_h \geq 3 \wedge gap = -1 \vee \\
&LastBuild = h \wedge s_h \geq 6 \wedge gap = -1 \vee \\
&Partial\ conviction \wedge s_h \geq 6 \wedge gap = -1
\end{aligned}$$

- Match whenever $Lastbuild = h$, Match is feasible, and we did not choose Override.
- Wait in any other case

8.3 $\gamma = 1, \alpha = 0.4$

Result: $\rho = 0.6679688$

- Adopt whenever
  $s_h = 1 \wedge s_a = 0$
- Match whenever Match is feasible.
- Wait in any other case.

One can see that when $s_a < s_h$, the greater s_a is ,the higher is the risk the attackers are willing to take (select Wait instead of Adopt). Similarly when $s_h < s_a$, the greater s_a is ,the smaller is the risk the attackers are willing to take (select Override instead of Wait or Match). In addition the greater the attacker's impact strength is ,the higher is the risk the attackers are willing to take.