CS 765 : Report - Project Part 1

Simulation of a P2P Cryptocurrency Network

Aakriti - 190050002
Aditya Badola - 190050006
Gaurang Dev - 19D070024

September 7, 2021

1 What are the theoretical reasons of choosing the exponential distribution for inter-arrival time between transactions?

Let T denote the time we need to wait, before the next transaction occurs (i.e. the inter-arrival time between transactions). If T has an exponential distribution, that implies:

$$\mathcal{P}(\beta < T < (\beta + \Delta\beta)) \propto \Delta\beta$$

This assumption is very practical and hence exponential distribution is a good estimate for inter-arrival time between transactions.
Moreover, the exponential distribution has **memoryless property**, that is, the inter-arrival time between two consecutive transactions is independent of the previous occurrences, and the number of transactions in given time follows Poisson distribution.

2 Algorithm used for connecting the peers randomly such that the resultant network is connected

Define p to be the probability with which two nodes are connected in an arbitrary network. For every two nodes i and j ($i, j \in [n]$), sample a random variable from a uniform distribution $\mathcal{U}(0, 1)$, call it r. If $r < p$, connect the nodes i and j. After iterating it over all the node pairs, we get a network. If the network is connected, we are done. Else, repeat the same algorithm (from scratch) until the network gets connected. *Note:* p determines the sparsity of the network.

3 Why is the mean of d_{ij} inversely related to c_{ij}?

d_{ij} denotes the queuing delay at node i to forward a message to node j. Whereas, c_{ij} is the link speed between nodes i and j. c_{ij} determines how fast or slow the link propagates its queue. Higher the link speed, it will take lesser time to propagate the message and hence the queue will be cleared faster. Therefore, d_{ij} is inversely proportional to c_{ij}.

4 Justify the chosen mean value for T_k.

T_k is the inter-arrival time between blocks. Higher value of T_k would generate less number of blocks and transaction limit for each block will get saturated. On the other hand, lower value of T_k would mean huge number of blocks being generated and there will be fewer transactions in each block. We need an optimal value of T_k, hence mean value of T_k has to be justifiably chosen.

5 Variation with fraction of slow nodes (z)

When we increase the fraction of slow nodes, the fraction of slow links increases as a result of which lesser number of blocks are broadcasted amongst the nodes in the given time of simulation. Therefore, the length of blockchain decreases.
Following are the blockchains for peer 0:

- **Case 1:** $z = 0\%$ (All peers are fast)

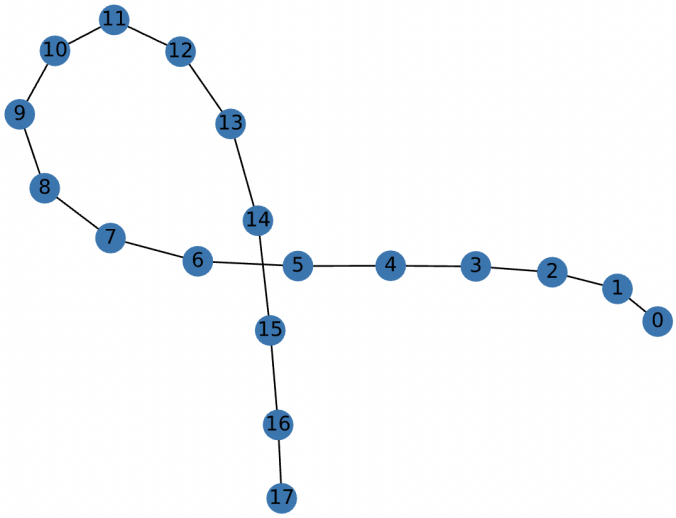


Figure 1: length of blockchain: 18

- **Case 2:** $z = 50\%$ (Half of the nodes are fast)

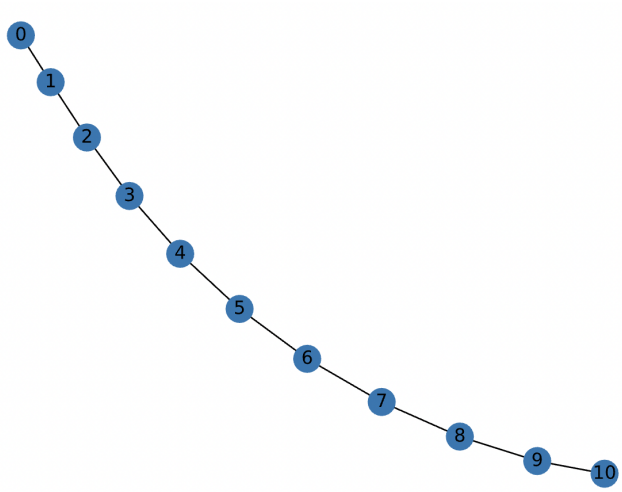


Figure 2: length of blockchain: 11

- **Case 3:** $z = 100\%$ (All nodes are slow)

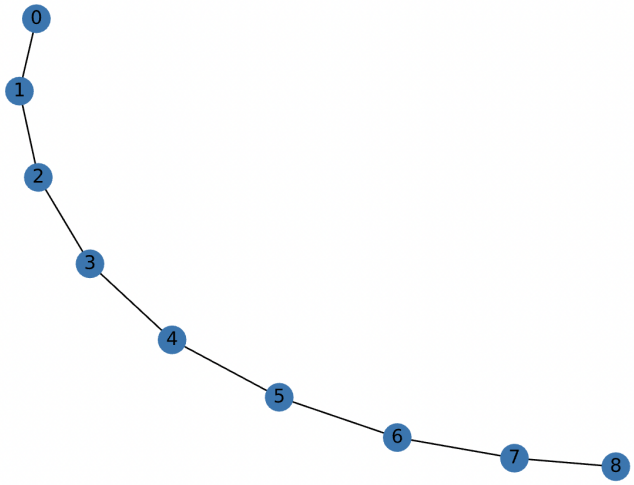


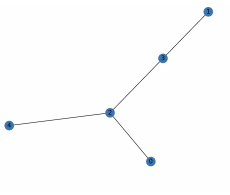
Figure 3: length of blockchain: 9

6 Variation with number of nodes (n):

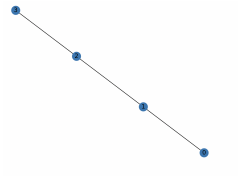
When we increase the number of peers, more peers will be mining for the blocks. So, there will be an increase in the number of block generation events, that is, there will be more blocks in the blockchain.

Following are the blockchains for peer 0:

- **Case 1:** $n = 5$

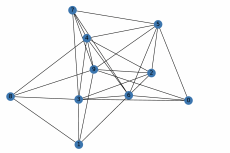


(a) Peer network

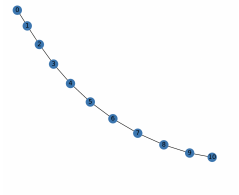


(b) length of blockchain: 4

- **Case 2:** $n = 10$

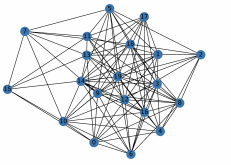


(a) Peer network

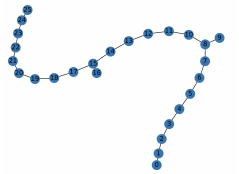


(b) length of blockchain: 11

- **Case 3:** $n = 20$



(a) Peer network



(b) length of blockchain: 25, **forking**

7 Variation with the mean transaction inter-arrival time (T_{tx}):

As $T_{tx}(\propto \frac{1}{\lambda_{tx}})$ increases, the interarrival time between two consecutive transactions increases. So, less number of transactions will be generated given the simulation time, and therefore, each block will have lesser number of transactions.
On the other hand, as T_{tx} decreases, more transactions will be generated by every node, and hence number of transactions in each block increases.
Following are the blockchains for peer 0:

- **Case 1:** $\lambda_{tx} = 0.8$

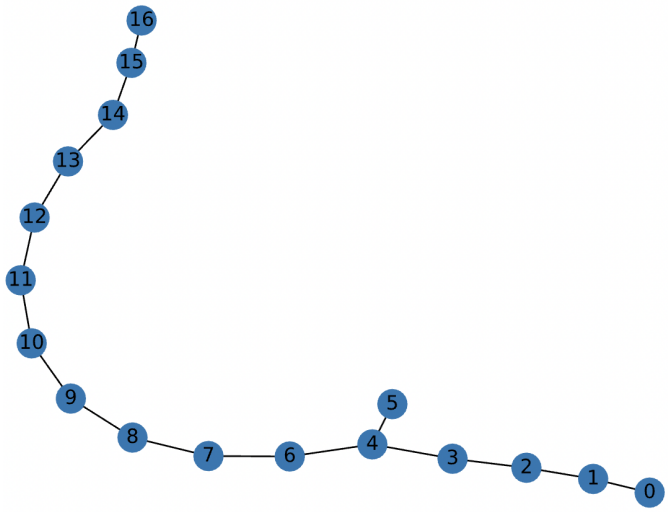


Figure 7: length of blockchain: 17

- **Case 2:** $\lambda_{tx} = 2$

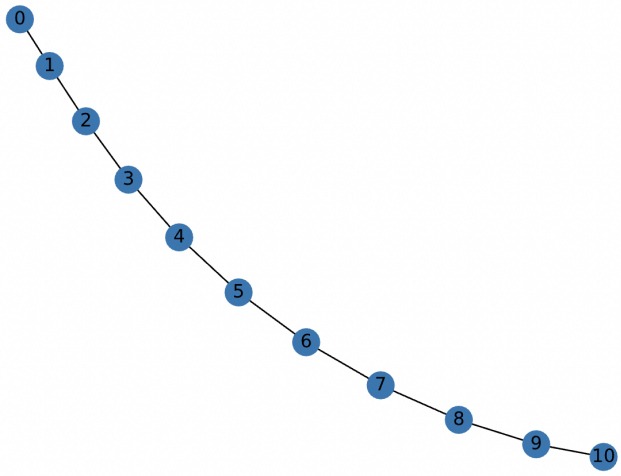


Figure 8: length of blockchain: 11

- **Case 3:** $\lambda_{tx} = 4$

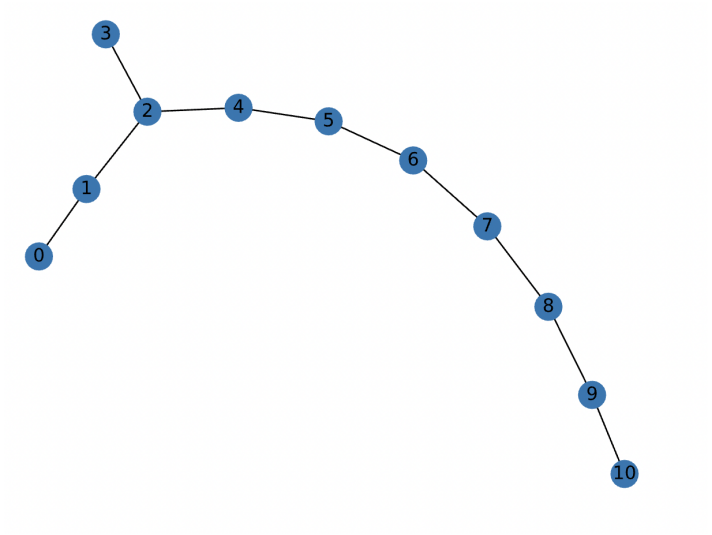


Figure 9: length of blockchain: 11, **forking**

8 Variation with the mean block inter-arrival time (T_k):

As $T_k(\propto \frac{1}{\lambda_k})$ increases, the interarrival time between two consecutive blocks increases. So, less number of blocks will be generated given the simulation time, and therefore, there will be lesser branching.
On the other hand, as T_k decreases, more blocks will be generated by every node, so nodes are extending their longest chain in a very short duration of time. As a result, there will be more forking.
Following are the blockchains for peer 0:

- **Case 1:** $\lambda_k = 0.001$

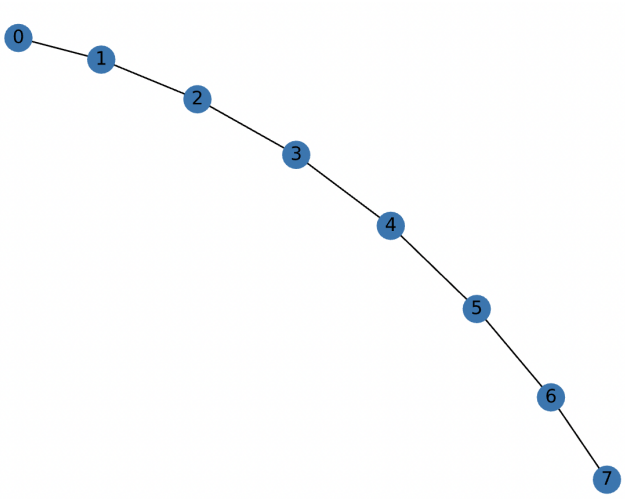


Figure 10: length of blockchain: 8

- **Case 2:** $\lambda_k = 0.0025$

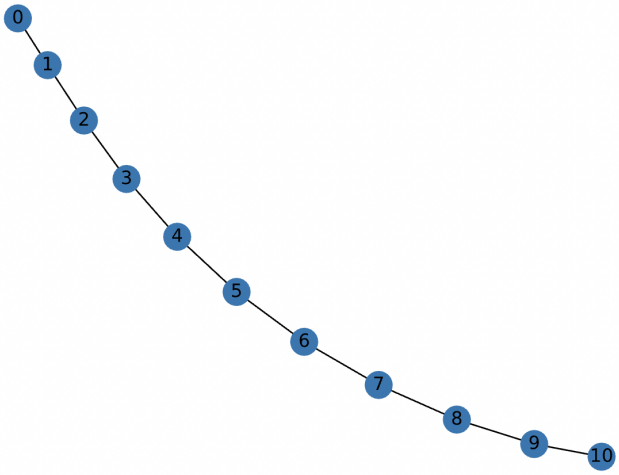


Figure 11: length of blockchain: 11

- **Case 3:** $\lambda_k = 0.005$

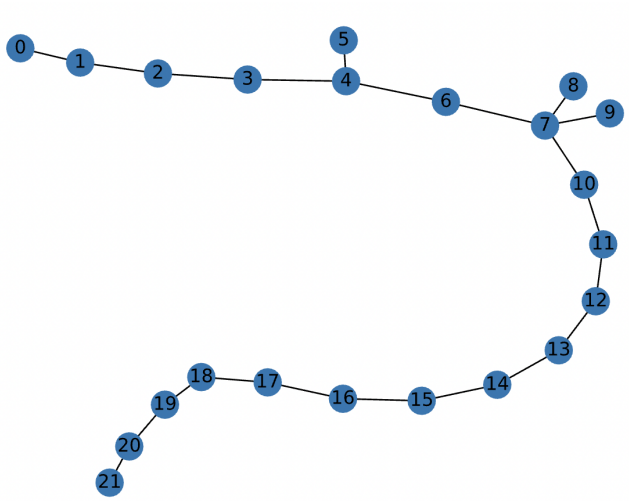


Figure 12: length of blockchain: 22, **forking**

9 Variation of ratio of the number of blocks generated by each node to total number of blocks generated

- **Case 1:** $z = 50, \lambda_k = 0.005$

Node ID	Type of Node	Ratio
0	slow	0.0217391
1	fast	0.152174
2	slow	0.0652174
3	fast	0.108696
4	slow	0.0869565
5	slow	0.0434783
6	fast	0.173913
7	fast	0.152174
8	fast	0.108696
9	slow	0.0434783

- **Case 2:** $z = 50, \lambda_k = 0.0025$

Node ID	Type of Node	Ratio
0	slow	0.0588235
1	fast	0.176471
2	slow	0
3	fast	0.117647
4	slow	0
5	fast	0.117647
6	slow	0
7	fast	0.0588235
8	fast	0.176471
9	slow	0.0588235

- **Case 3:** $z = 50, \lambda_k = 0.001$

Node ID	Type of Node	Ratio
0	fast	0.2
1	fast	0.2
2	slow	0
3	fast	0.4
4	slow	0
5	slow	0
6	fast	0.2
7	slow	0
8	fast	0
9	slow	0

As λ_k increases (or CPU power increases), the mean inter-arrival time between blocks T_k decreases. So, blocks are generated more often. Therefore, the ratio of blocks generated by slow nodes to the total number of blocks becomes almost same as the ratio of blocks generated by fast nodes to the total number of blocks.
On the other hand, when λ_k is low, blocks are generated at a slow rate. So, the latency of propagation between the nodes play an important role. The faster nodes will generate blocks much more often as compared to slow nodes (look at table 3).
As z denotes the percentage of slow nodes, if z is at extreme values, i.e. 0 or 1, the ratio is equal for all nodes regardless of it being fast or slow. On the other hand, when z is say 50%, the ratio is higher for the fast nodes, given that time of simulation and peers are constant.

CS 765 : Report - Project Part 2

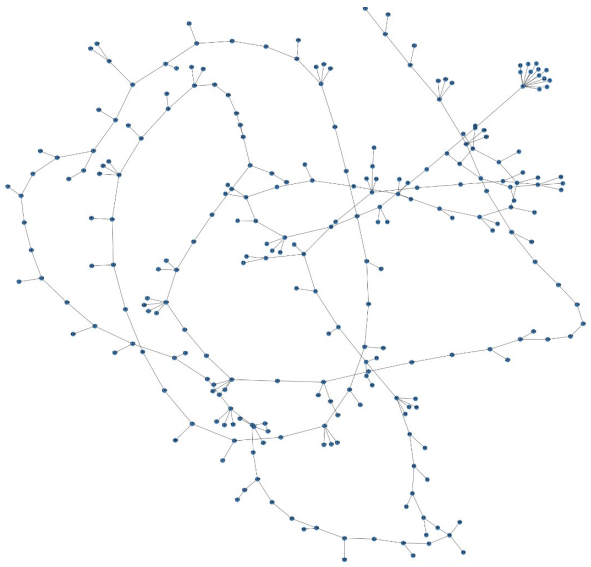
Simulating a selfish mining attack using the P2P Cryptocurrency

Aakriti - 190050002
Aditya Badola - 190050006
Gaurang Dev - 19D070024

October 17, 2021

1 Terminology and Assumptions

- Simulation time is taken to be $500s$.
- We have taken number of peers (N) to be 20 instead of 100. This is because, the visualisations are better for smaller N. For large enough Ns, the blockchain trees were cluttered and inconclusive.
  We experimented for $N = 100$, simulation time $= 100s$ and obtained the following blockchain tree for the default parameters:



- For each experiment, we have included the following in the report:

MPU Adversary	$\frac{\text{\# blocks mined by adversary in main chain}}{\text{total \# blocks mined by adversary}}$
MPU Overall	$\frac{\text{\# blocks in main chain}}{\text{total \# blocks mined by all nodes}}$
Effective α	$\frac{\text{\# blocks mined by adversary in main chain}}{\text{total \# blocks in the main chain}}$

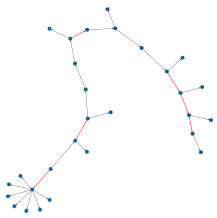
- There is also an 'Upper bound on effective α'. This is because, at the end of simulation, the adversary might have had a lead but could not release her private chain. So, this bound takes into account the situation when the adversary was able to release her private chain.
- Note that the simulations may not show a trend, because the experiments are non-deterministic. There is randomness involved in terms of latency, inter-arrival time between blocks and transactions, etc.
- **In the blockchain plots that follow, the red lines indicate that the block that follows it, is created by the adversary. The dark blue lines indicate that the block followed is in the private chain of the adversary.**

2 Selfish Miner

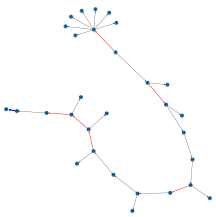
2.1 Variation with fraction of nodes, an adversary is connected to (ζ)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\lambda_k = 0.0025$, $\alpha = 0.3$ constant.
As ζ increases, more fraction of honest nodes are connected to the adversary. So, the adversary will be able to broadcast her blocks to a greater extent when at state $lead = 0'$. That is, γ increases, increasing the effective α and MPU Adversary.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\zeta = 0.25$



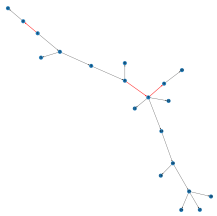
(a) An honest peer



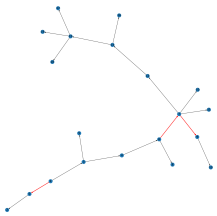
(b) Selfish miner

MPU Adversary	0.714286
MPU Overall	0.5
Effective α	0.333333
Upper bound on effective α	0.375

- **Case 2:** $\zeta = 0.5$



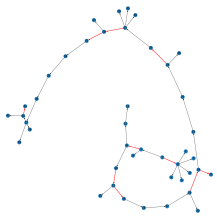
(a) An honest peer



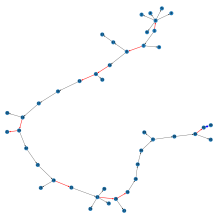
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.55
Effective α	0.272727
Upper bound on effective α	0.272727

- **Case 3:** $\zeta = 0.75$



(a) An honest peer



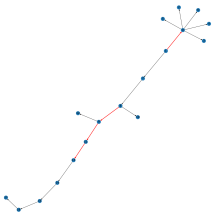
(b) Selfish miner

MPU Adversary	0.818182
MPU Overall	0.568182
Effective α	0.36
Upper bound on effective α	0.384615

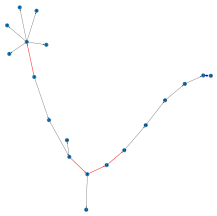
2.2 Variation with mean inter-arrival time between blocks (λ_k)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\zeta = 0.5$, $\alpha = 0.3$ constant.
As $T_k(\propto \frac{1}{\lambda_k})$ increases, the inter-arrival time between two consecutive blocks increases. So, less number of blocks will be generated given the simulation time, and therefore, there will be lesser branching. Due to less branching, lesser blocks are discarded because of forking. So, MPU overall increases. There is no conclusive effect on MPU Adversary and effective α.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\lambda_k = 0.001$



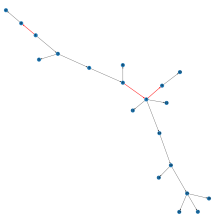
(a) An honest peer



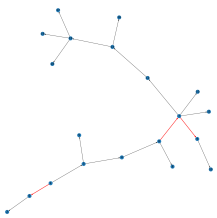
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.578947
Effective α	0.363636
Upper bound on effective α	0.416667

- **Case 2:** $\lambda_k = 0.0025$



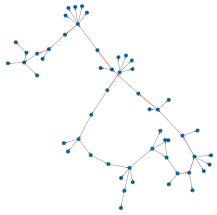
(a) An honest peer



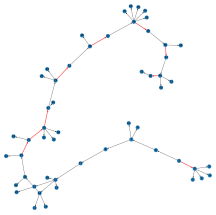
(b) Selfish miner

MPU Adversary	0.714286
MPU Overall	0.5
Effective α	0.333333
Upper bound on effective α	0.375

- **Case 3:** $\lambda_k = 0.005$



(a) An honest peer



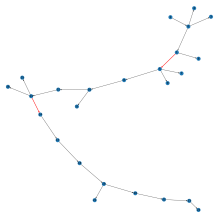
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.444444
Effective α	0.375
Upper bound on effective α	0.375

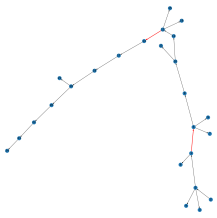
2.3 Variation with hashing power of adversary (α)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\lambda_k = 0.0025$, $\zeta = 0.5$ constant.
As α increases, the hashing power of the adversary increases. So, the selfish miner will mine blocks more often, giving the advantage to grow her private chain longer, therefore discarding the efforts of honest miners. So, effective α will increase more rapidly as α increases. MPU Adversary increases. MPU Overall steadily decreases. Forking increases since the adversary has more resources and mines blocks more often.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\alpha = 0.1$



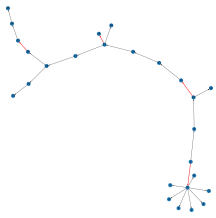
(a) An honest peer



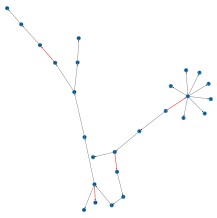
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.6
Effective α	0.133333
Upper bound on effective α	0.133333

- **Case 2:** $\alpha = 0.2$



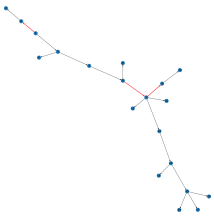
(a) An honest peer



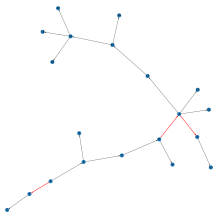
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.515152
Effective α	0.294118
Upper bound on effective α	0.294118

- **Case 3:** $\alpha = 0.3$



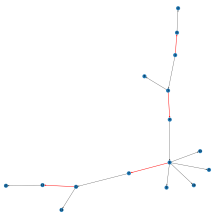
(a) An honest peer



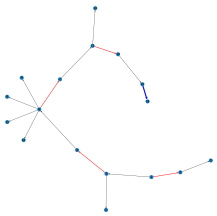
(b) Selfish miner

MPU Adversary	1
MPU Overall	0.55
Effective α	0.272727
Upper bound on effective α	0.272727

- **Case 4:** $\alpha = 0.4$



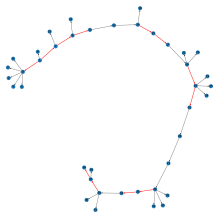
(a) An honest peer



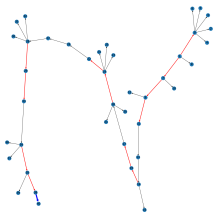
(b) Selfish miner

MPU Adversary	0.8
MPU Overall	0.529412
Effective α	0.444444
Upper bound on effective α	0.5

- **Case 5:** $\alpha = 0.5$



(a) An honest peer



(b) Selfish miner

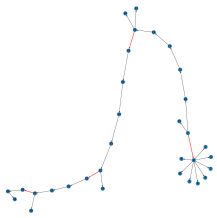
MPU Adversary	1
MPU Overall	0.487805
Effective α	0.6
Upper bound on effective α	0.619048

3 Stubborn Miner

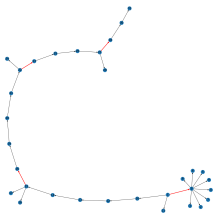
3.1 Variation with fraction of nodes, an adversary is connected to (ζ)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\lambda_k = 0.0025$, $\alpha = 0.3$ constant.
As ζ increases, more fraction of honest nodes are connected to the adversary. So, the adversary will be able to broadcast her blocks to a greater extent when at state $lead = 0'$. That is, γ increases, increasing the effective α and MPU Adversary.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\zeta = 0.25$



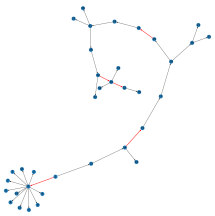
(a) An honest peer



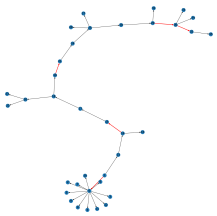
(b) Stubborn miner

MPU Adversary	1
MPU Overall	0.59375
Effective α	0.210526
Upper bound on effective α	0.210526

- **Case 2:** $\zeta = 0.5$



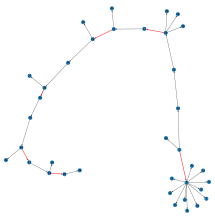
(a) An honest peer



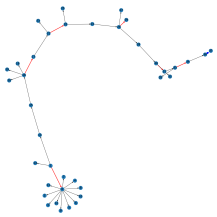
(b) Stubborn miner

MPU Adversary	1
MPU Overall	0.444444
Effective α	0.3125
Upper bound on effective α	0.3125

- **Case 3:** $\zeta = 0.75$



(a) An honest peer



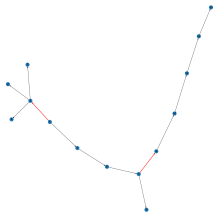
(b) Stubborn miner

MPU Adversary	0.714286
MPU Overall	0.421053
Effective α	0.3125
Upper bound on effective α	0.352941

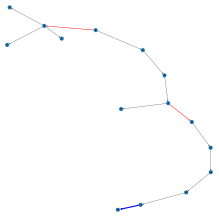
3.2 Variation with mean inter-arrival time between blocks (λ_k)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\zeta = 0.5$, $\alpha = 0.3$ constant.
As $T_k(\propto \frac{1}{\lambda_k})$ increases, the inter-arrival time between two consecutive blocks increases. So, less number of blocks will be generated given the simulation time, and therefore, there will be lesser branching. Due to less branching, lesser blocks are discarded because of forking. So, MPU overall increases. There is no conclusive effect on MPU Adversary and effective α.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\lambda_k = 0.001$



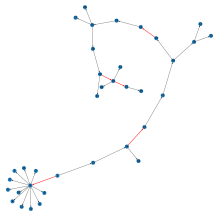
(a) An honest peer



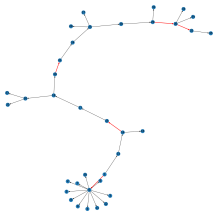
(b) Stubborn miner

MPU Adversary	0.666667
MPU Overall	0.666667
Effective α	0.2
Upper bound on effective α	0.272727

- **Case 2:** $\lambda_k = 0.0025$



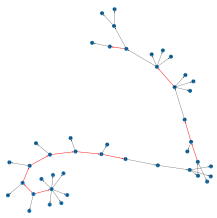
(a) An honest peer



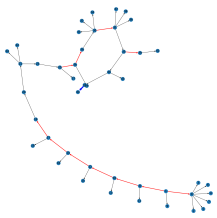
(b) Stubborn miner

MPU Adversary	1
MPU Overall	0.444444
Effective α	0.3125
Upper bound on effective α	0.3125

- **Case 3:** $\lambda_k = 0.005$



(a) An honest peer



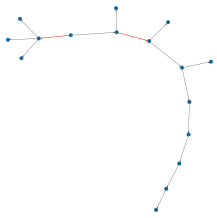
(b) Stubborn miner

MPU Adversary	0.833333
MPU Overall	0.422222
Effective α	0.526316
Upper bound on effective α	0.55

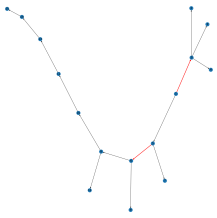
3.3 Variation with hashing power of adversary (α)

Keeping $N = 20$, $\lambda_{tx} = 0.5$, $\lambda_k = 0.0025$, $\zeta = 0.5$ constant.
As α increases, the hashing power of the adversary increases. So, the stubborn miner will mine blocks more often, giving the advantage to grow her private chain longer, therefore discarding the efforts of honest miners. So, effective α will increase more rapidly as α increases. MPU Adversary increases. MPU Overall steadily decreases. Forking increases since the adversary has more resources and mines blocks more often.
Following are the blockchains for an honest peer and the adversary:

- **Case 1:** $\alpha = 0.1$



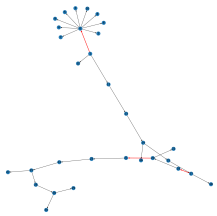
(a) An honest peer



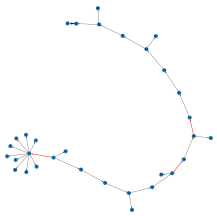
(b) Stubborn miner

MPU Adversary	1
MPU Overall	0.625
Effective α	0.2
Upper bound on effective α	0.2

- **Case 2:** $\alpha = 0.2$



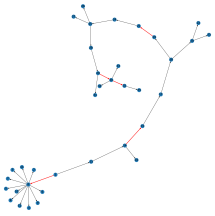
(a) An honest peer



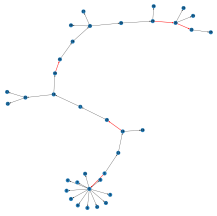
(b) Stubborn miner

MPU Adversary	0.5
MPU Overall	0.5
Effective α	0.125
Upper bound on effective α	0.176471

- **Case 3:** $\alpha = 0.3$



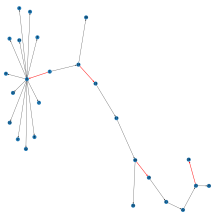
(a) An honest peer



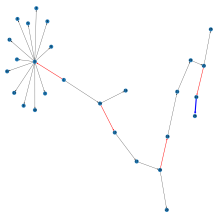
(b) Stubborn miner

MPU Adversary	1
MPU Overall	0.444444
Effective α	0.3125
Upper bound on effective α	0.3125

- **Case 4:** $\alpha = 0.4$



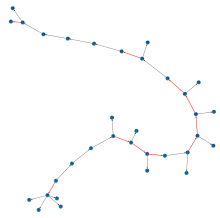
(a) An honest peer



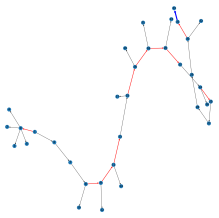
(b) Stubborn miner

MPU Adversary	0.8
MPU Overall	0.407407
Effective α	0.363636
Upper bound on effective α	0.416667

- **Case 5:** $\alpha = 0.5$



(a) An honest peer



(b) Stubborn miner

MPU Adversary	0.909091
MPU Overall	0.588235
Effective α	0.5
Upper bound on effective α	0.52381

4 Comparision between Selfish and Stubborn miners

It can be observed that selfish mining is not the optimal malicious type of mining. Stubborn mining is more profitable since it holds back blocks more often discarding the blocks generated by honest blocks frequently. Although the results are not very clear indicative of that, this is because these are just a few tens of experiments performed under restricted N and simulation time.

5 Theoretical $\gamma = 0$, $\gamma = 1$ along with selfish, stubborn plots

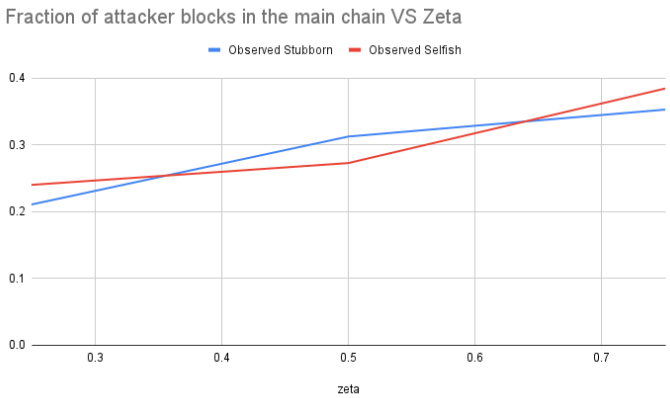


Figure 23: Variation with ζ

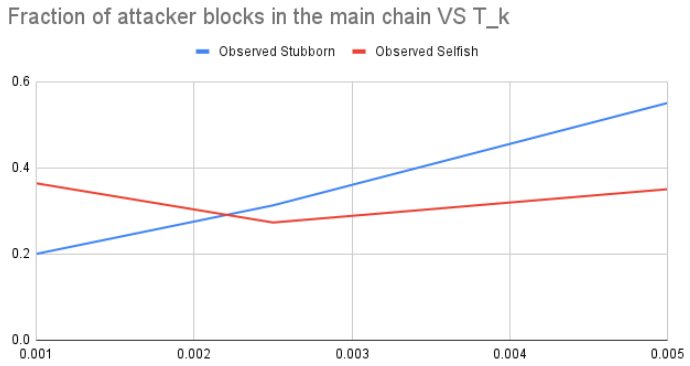


Figure 24: Variation with λ_k

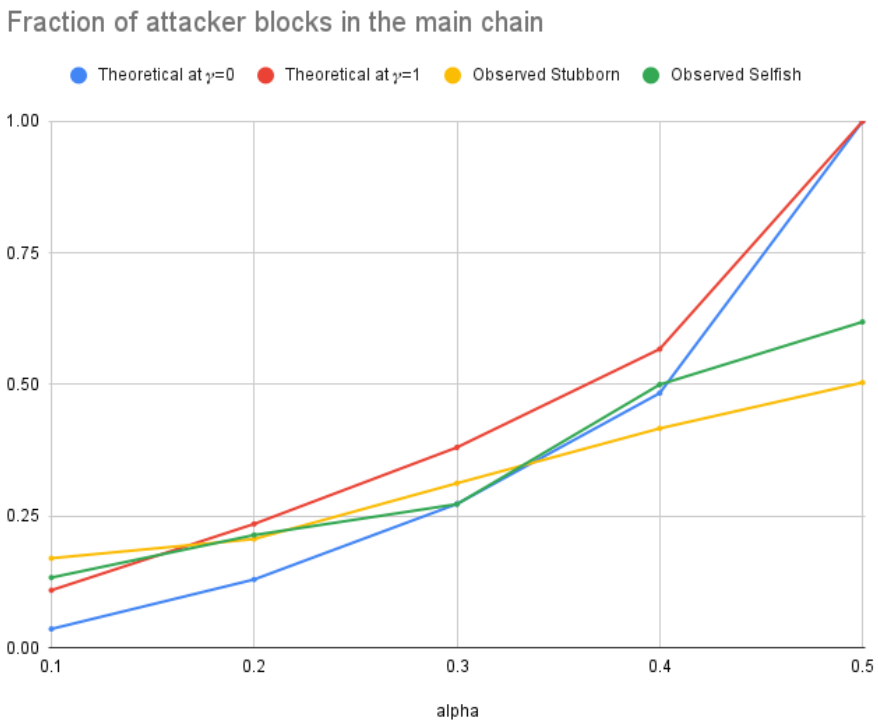


Figure 25: Variation with α

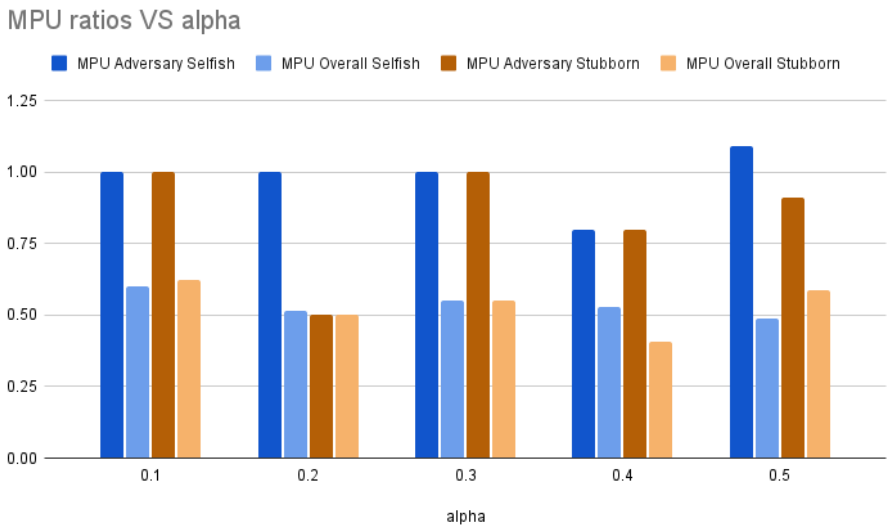


Figure 26: Variation of MPU w.r.t. α

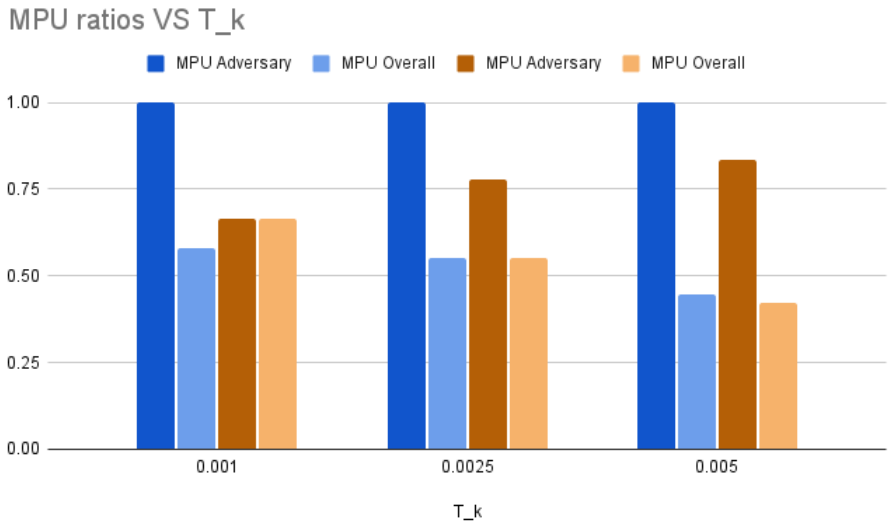


Figure 27: Variation of MPU w.r.t. λ_k

CS 765 : Report - Project Part 3

Building a layer-2 DAPP on top of Blockchain

Aakriti - 190050002
Aditya Badola - 190050006
Gaurang Dev - 19D070024

November 20, 2021

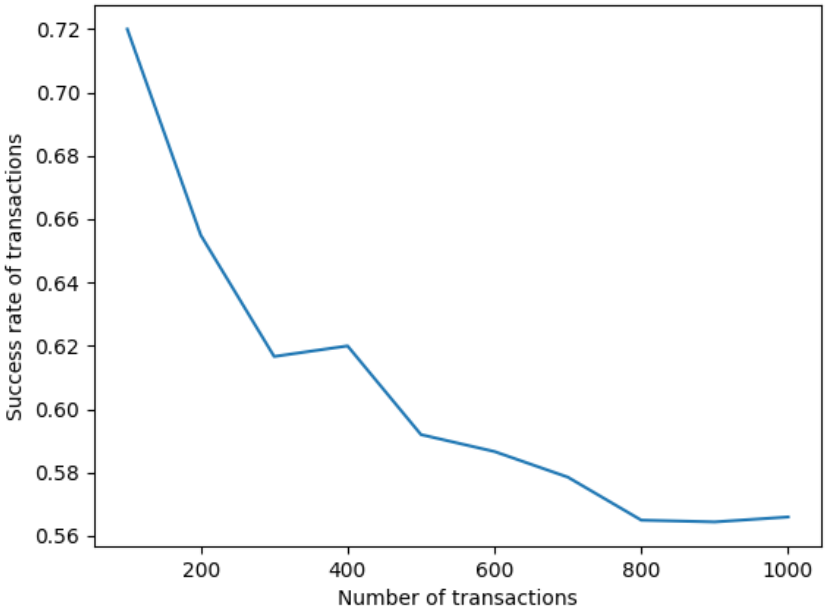


Figure 1: Success Rate versus Number of Transactions

Number of transactions	Ratio
100	0.72
200	0.655
300	0.617
400	0.62
500	0.592
600	0.587
700	0.5785
800	0.565
900	0.5644
1000	0.566

The function `sendAmount(user_id_1, user_id_2)` function is used to transfer 1 coin from `user_id_1` to `user_id_2`. The transaction fails in case every path from `user_id_1` to `user_id_2` contains at least one node unable to furnish the required number of coins from its joint account for co-extending the path. Thus, the path gets blocked despite the graph being connected. With each transaction a set of balances over the contributing edges, drops. Thus, the number of blocking nodes increases with the increasing total number of transactions, thereby causing a decline in the success rate of transactions.