# CS 765 Assignment 1

Simulation of a P2P Cryptocurrency Network

Niyati Mehta - 200050091
Parth Dwivedi - 200050100
Hastyn Doshi - 200070025

February 2023

## 1 Introduction

## Question 2

The reason for choosing the exponential distribution is that it models the inter arrival time of a system where at each instant in time there is some probability of an event occurring, in this case a transaction being generated by any peer. Suppose $\beta$ is a parameter proportional to the likelihood that a transaction is generated at any instant. Thus in a very small time interval $\delta$ the probability of an event occurring can be taken as $\beta\delta$. This implies Now the probability that there is no block for a time of $n\delta$, that is equivalent to the probability of the inter arrival time being greater than $n\delta$ is nothing but $(1-\beta\delta)^n$. Let $n\delta$ be denoted by x and the inter arrival time we need to model be a random variable I. Thus we have

$$P(I > x) = \left(1 - \frac{\beta x}{n}\right)^n$$

Now taking $\delta$ tending to zero and to signify that the event can happen at any instant and thus having $n$ tending to infinity to have a non zero time interval we get

$$P(I > x) = \lim_{n\to\infty} \left(1 - \frac{\beta x}{n}\right)^n = e^{-\beta x}$$

This is nothing but the Cumulative Distribution Function for the exponential distribution.

## Question 5

The exponential distribution is given by the probability distribution function of $P(I = x) = \beta e^{-\beta x}$ where higher the $\beta$, higher is the probability of obtaining a lower value on sampling the random variable. Now the mean of this distribution is given by $1/\beta$. This implies for our case

$$\frac{1}{\beta} = \frac{96kbits}{c_{ij}}$$

. This means if $c_{ij}$ is inversely proportional to the mean, it is directly proportional to the $\beta$ parameter of the exponential distribution from which $d_{ij}$ is sampled. This is done as more the link speed $c_{ij}$, more should be the probability of receiving a smaller value of the inter arrival time between queues (represented as a Poisson process), which is nothing but the queuing delay $d_{ij}$. Higher the link speed, faster are the packets transferred across the link and so lesser is the waiting time at the queue.

## Question 7

We want the $T_k$ such that average of the inter arrival time between any two blocks from any two nodes is I. We also want the node with more hashing power to generate a block in lesser time. To accomplish both of these goals we keep a meantime of $I/h_i$ for each peer. This clearly implies nodes with higher hashing power have a lesser mean inter arrival time. Now for the average inter arrival time between blocks from any two peers, this is nothing but the probability none of the peers generates a block in some time. This is given by

$$P(I_1, I_2, ..I_n > x) = P(I_1 > x)P(I_2 > x)...$$

Using the Independence of the random variables. If we take a meantime of I/hi for each peer the above terms becomes

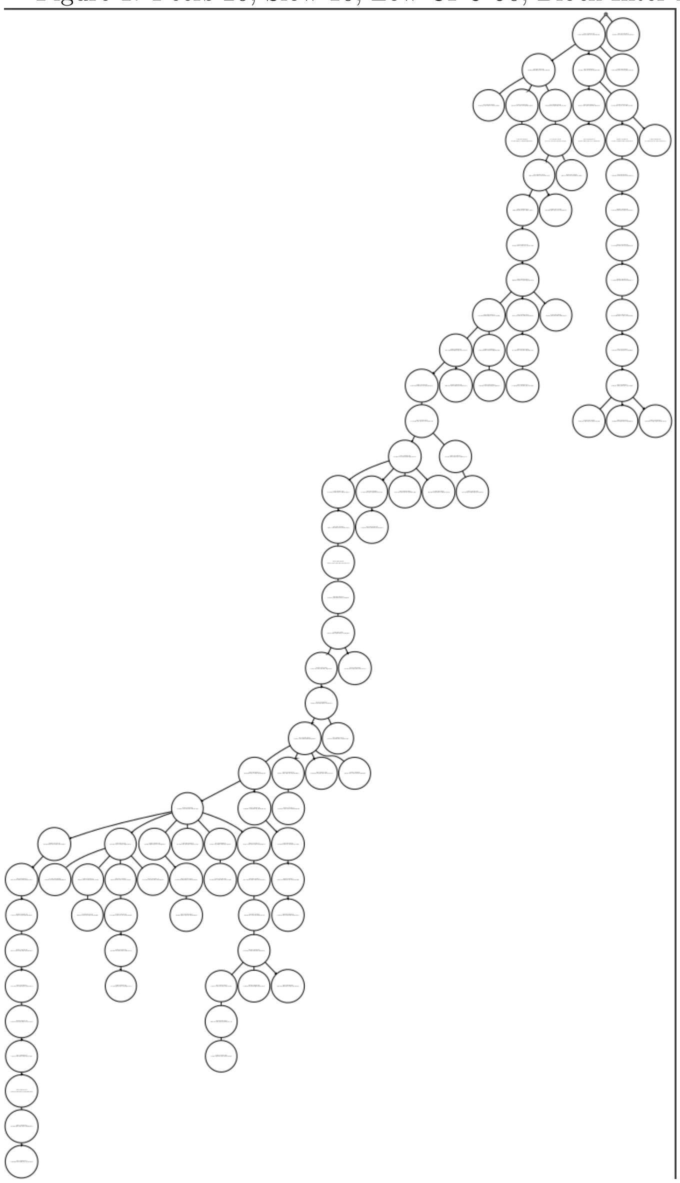
$$\Pi_{i=1..n} e^{-h_i x/I} = e^{-\Sigma h_i x/I} = e^{-x/I}$$

The mean of this distribution is nothing but the I we wanted, thus the choice $I/h\_i$ for the mean time.

## Question 8

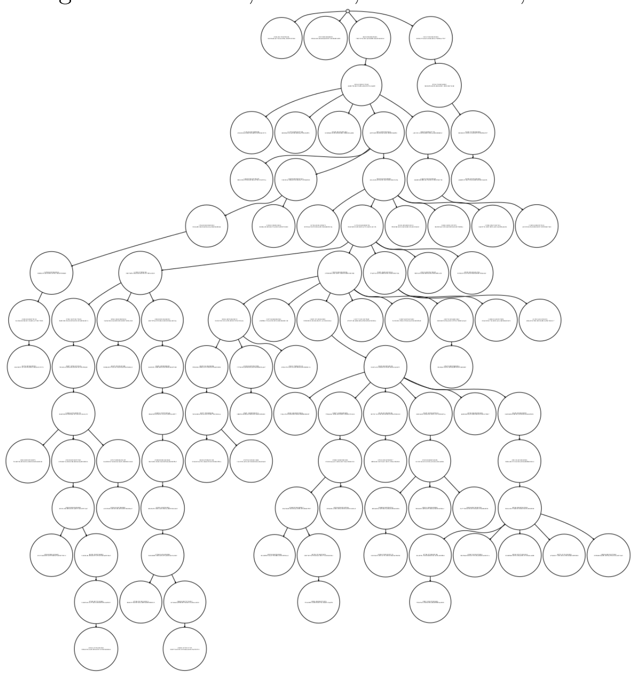
Observations :-

Figure 1: Peers 20, Slow 10, Low CPU 90, Block Inter-Arrival Time 1000



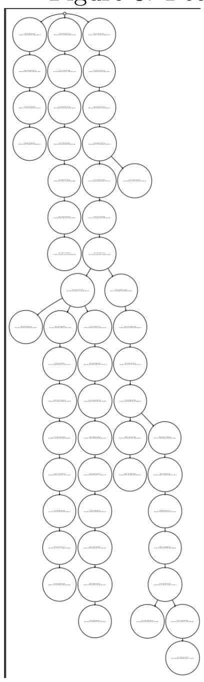
The block inter-arrival time is moderately high for this network but due to high number of peers we see some moderate branching in this tree.

Figure 2: Peers 20, Slow 90, Low CPU 10, Block Inter-Arrival Time 1000



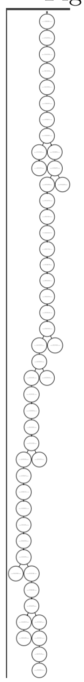
Here, the number of nodes and block inter-arrival time remain the same, but we have increased the number of slow nodes, meaning it takes time for blocks to propagate and reduced the percentage of low CPU nodes, which implies that blocks will be generated fairly quickly. Both these factors with lead to more branching than the previous case.

Figure 3: Peers 5, Slow 10, Low CPU 50, Block Inter-Arrival Time 100



Here, the number of nodes is lower, so the number of simultaneous branches is limited to at most 5. Here, the block inter-arrival time is very less, only 100 ms, and so there are bound to be a lot of branches, which is seen in the tree.

Figure 4: Peers 5, Slow 10, Low CPU 50, Block Inter-Arrival Time 1000

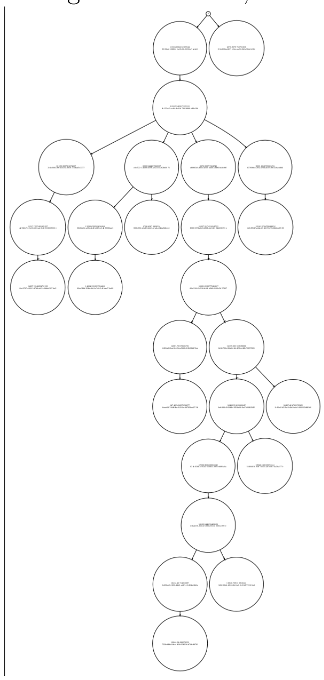


Here, all parameters remain the same save for the inter-arrival block time which has been increased by a factor of 10. Since the inter-arrival time for blocks has increased, it gives more time for the blocks to propagate through the network and so, branching should be decreased.

Figure 5: Peers 5, Slow 10, Low CPU 50, Block Inter-Arrival Time 10000

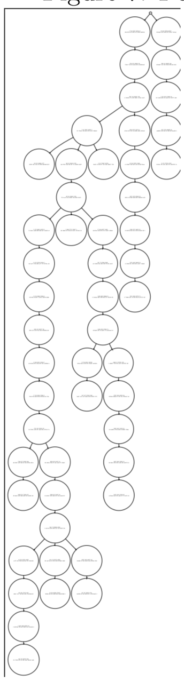
Here, the block inter-arrival time has been further increased by a factor of 10, all other factors remain the same. By the same logic as above, branching here should be decreased even further.

Figure 6: Peers 5, Slow 90, Low CPU 10, Block Inter-Arrival Time 1000



Here, we have set the block inter-arrival time back to 1000, as in Figure 4, but the percentage of slow nodes has increased and Low CPU nodes has decreased, which means blocks are being generated faster and they are taking longer to propagate through the network. Both these factors contribute to more branching and we see this in the tree, compared to Figure 3.

Figure 7: Peers 5, Slow 90, Low CPU 50, Block Inter-Arrival Time 1000



Here, we try to find out the effect of computing power on branching in blockchain trees. Compared to the previous figure, only the percentage of low CPU nodes has been increased to 50. Since, the blocks are taking longer to generate, this gives them more time to propagate through the network and so branching should decrease compared to Figure 6.

These results are visually observed, we shall also provide empirical data to validate our theories. If the ratio defined in the paper increases, it implies lesser branching, and vice-verse

#### Peers

On increasing the number of peers, the branching in general should increase, since each block now has to propagate through a larger network which takes more time, and so, the probability of another block being generated during this time becomes higher.

Taking Slow: 50.0 LowCPU: 50.0 MBlockTime: 100.0, for Peers = 5, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.3233246301131419 (Total blocks : 2298) and for Peers = 20, we get the same ratio as : 0.13333333333333333 (Total blocks : 510) Blocks in Longest Chain : 743
Taking Slow: 50.0 LowCPU: 50.0 MBlockTime: 1000.0, for Peers = 5, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.3233246301131419 (Total blocks : 2298) and for Peers = 20, we get the same ratio as : 0.19491525423728814 (Total blocks : 2280) Blocks in Longest Chain : 473
Taking Slow: 50.0 LowCPU: 50.0 MBlockTime: 10000.0, for Peers = 5, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.9144620811287478 (Total blocks : 2268) and for Peers = 20, we get the same ratio as : 0.7533512064343163 (Total blocks : 373) Blocks in Longest Chain : 743

#### CPU

On increasing the number of low hashing power nodes, each block takes more time to generate and hence, gives more time to resolve forks and hence branching should be less on increasing the number of low CPU nodes.

Taking Peers: 20 Slow: 50.0 MBlockTime: 10000.0, for LowCPU= 10, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.7533512064343163 (Total blocks : 421), for LowCPU = 50, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.7810945273631841 (Total blocks : 373) and for LowCPU = 90, we get the same ratio as Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.836104513064133 (Total blocks : 402)

#### Speed

On increasing the number of slow peers, the branching in general should increase, since each block now takes longer to propagate through the same network, and so, the probability of another block being generated during this time becomes higher.

Taking Peers: 20 LowCPU: 90.0 MBlockTime: 10000.0, for LowCPU= 10, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.8060453400503779 (Total blocks : 402), for LowCPU = 50, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.7810945273631841

(Total blocks : 397) and for LowCPU = 90, we get the same ratio as Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.7663316582914573 (Total blocks : 398)

#### Block Inter Arrival Time

Clearly, if the inter arrival time between blocks is increased, this means that each block now has more time on average to propagate through the whole network, and hence branching should decrease on increasing the block inter-arrival time.

Taking Peers: 20 LowCPU: 50.0 Slow: 50, for MBlockTime= 100, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.13333333333333333 (Total blocks : 510), for MBlockTime= 1000, we get the Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.19491525423728814 (Total blocks : 472) and for MBlockTime= 10000, we get the same ratio as Ratio of Blocks in Longest Chain to Total Number of Blocks : 0.7533512064343163 (Total blocks : 373)

### 1.0.1 Transaction Time

Since the blocks and transactions are independent of each other, changing the transaction time does not cause any appreciable change in the block chain. This was verified by taking multiple values of Transaction Times for the cases above, and not obtaining any appreciable change in the values.