CS765-Assignment 1
P2P Cryptocurrency Network

Introduction

We implemented a discrete-event simulator for a peer-to-peer network that maintains an "event-queue" and global clock (i.e., priority queue) from which the earliest event is executed. As a result of present event execution, it may also push further future events. In this report, we describe some design choices and analyze different simulations based on various system parameters specified in the configuration file.

Question 2

Theoretical reasons for choosing the exponential distribution for inter-arrival time

In Bitcoin-like scale-free Networks, the process of transaction generation follows a Poisson process. Let δ be a very small time interval such that $\delta \to 0$. The probability that a transaction occurs within the next δ interval is proportional to δ

$$P(\text{Transaction is generated in the next } \delta \text{ interval}) = \beta \cdot \delta$$

The probability that the next transaction occurs after n such time intervals is given by $(1-\beta\delta)^n$. Substituting $t = n\delta$ and denoting the transaction generation time as the random variable t_x,

$$P(t_x > t) = (1 - \frac{\beta t}{n})^n$$

Taking the limit as $\delta \to 0$ for t to be finite, and $n \to \infty$,

$$P(t_x > t) \sim e^{-\frac{\beta}{t}t} = e^{-\beta}$$

$$P(t_x < t) = 1 - e^{-\beta}$$

The inter-arrival time of two events (in this case, Transaction) follows an exponential distribution.

$$P(t_x = t) = \beta \cdot e^{-\beta t}$$

Therefore, we choose the transaction inter-arrival time to be sampled from an exponential distribution.

Question 5

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

d_{ij} denotes the queuing delay at node i in forwarding a message to node j, and c_{ij} is the link speed between nodes i and j in bits/sec. The queuing delay depends on the rate at which traffic is pushed out of the queue. It would be lower if the link speed is higher, and vice versa. Let us consider n packets in the queue before a packet P. The higher the link speed (c_{ij}) towards the destination, the n packets will be removed from the queue per unit time at a fast rate, and the packet P itself gets dequeued, i.e., reduces the average queuing delay (d_{ij}). Hence, they are inversely proportional.

$$\text{Number of bits queued before} = n$$

$$\text{Packet size} = k \text{ bits}$$

$$\begin{aligned} \mathrm{d}_{ij} &= \frac{\text{Number of bits queued before } P}{\text{Link speed}} \\ &= \frac{nk}{c_{ij}} \end{aligned}$$

Question 7

We decided to set the default value of T_k to 100 seconds, which is approximately 15 minutes. When a new block (let's call it Block A) is mined, it takes about a one-third second to transmit it to each connected peer in the network. So, it will take a few seconds to send Block A to all peers. Our goal is to minimize the occurrence of forks in the blockchain. A fork happens when another node mines a new block before Block A reaches that node. To reduce the chances of forking, we chose T_k to be significantly higher than the transmission time. This ensures that we have a more centralized chain without many forks.

We need to find the right balance for T_{tx} (transaction time) so that each block doesn't end up with too few or too many transactions. To figure this out, we'll run experiments while keeping an eye on the T_k/T_{tx} ratio.

Commands to Run : ./run.sh

Run.sh contains the following:

```
#!/bin/bash
```

```
# Set simulation parameters
N_VAL=40
Z0_VAL=40
Z1_VAL=40
T_TXT_VAL=100
I_VAL=1000

# Clean previous outputs
rm -f graph_program main_program
rm -f logfile.txt edge_trees.txt adjacency_list.txt vectors.txt Graph_peers.png
rm -f *.png

# Compile and run graph program
g++ graph.cpp -o graph_program
./graph_program $N_VAL

# Uncomment below to visualize the peer network (optional)
# python3 graph_visualize.py

# Compile and run main program with specified arguments
g++ -g main.cpp -o main_program
./main_program --n=$N_VAL --z0=$Z0_VAL --z1=$Z1_VAL --T_tx=$T_TXT_VAL --I=$I_VAL

# Uncomment below to get trees visualization (optional)
# python3 tree_visualize.py

# Uncomment below to get statistics (optional)
# python3 statistics.py
```

Visualization & Analysis

Experiment 1

- $N = 20$ - Number of nodes in the network.
- $Z0 = 40$ - Percent of slow nodes"
- $Z1 = 40$ - Percent of low cpu nodes
- $T_{tx} = 10$ - Mean Transaction time in seconds.
- $I = 1000$ - Mean Inter-arrival time of transactions in seconds.

Results

- **Number of blocks mined:** 50
- **Number of blocks in the longest chain:** 50

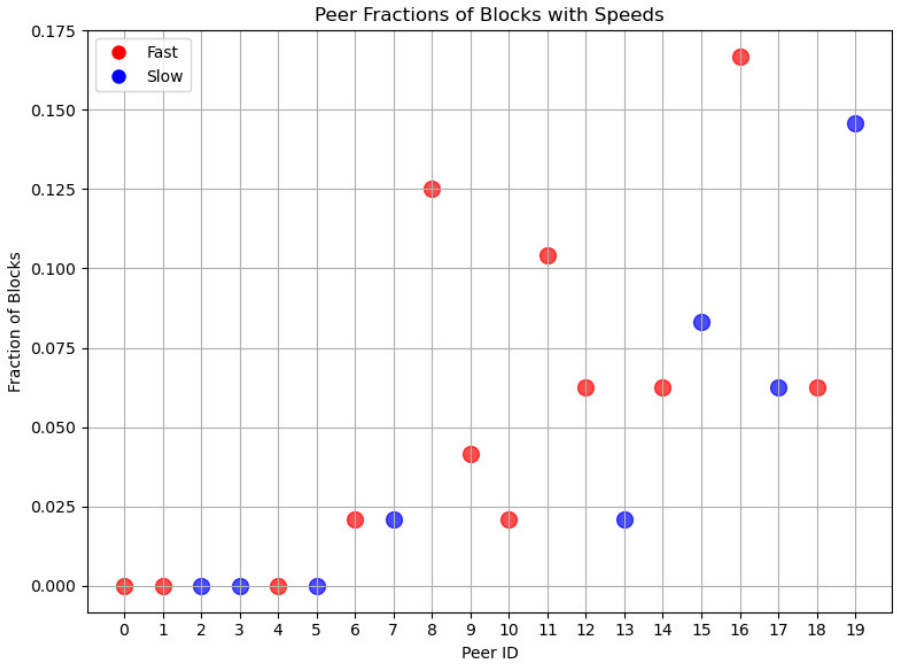
High Hash Power	Fast Node	Block Fraction
True	True	0.59
True	False	0.33
False	True	0.06
False	False	0.02

Figure 1: Block Chain Tree for Experiment 1

The anticipated network delay within a link is 0.5, significantly less than the specified inter-arrival time (I) with a value of 1000. Given that I is substantially higher than the network delay, the occurrence of side chains is unlikely

Statistics

We notice that the proportion of blocks created by a peer, compared to the total number of blocks in the longest chain, is entirely influenced by the hash power. The network delay does not play a role in this scenario.



Experiment 2

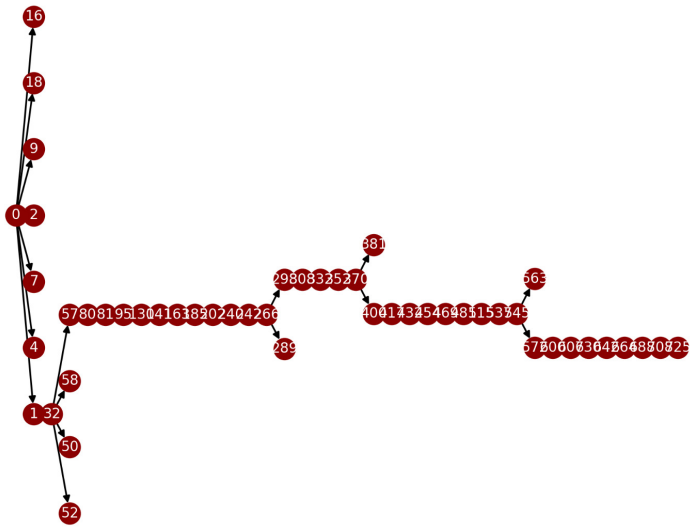
We reduced I to match T_tx, and as a result, we expected an increase in forks within the blockchain.

- $N = 20$ - Number of nodes in the network.
- $Z0 = 40$ - Percent of slow nodes"
- $Z1 = 40$ - Percent of low cpu nodes
- $T_{tx} = 5$ - Mean Transaction time in seconds.
- $I = 5$ - Mean Inter-arrival time of transactions in seconds.

Results

- **Number of blocks mined:** 50
- **Number of blocks in the longest chain:** 38

High Hash Power	Fast Node	Block Fraction
True	True	0.35
True	False	0.51
False	True	0.11
False	False	0.03



The fraction of blocks in the longest chain decreased as a result of the reduction in the mean block mining time

Statistics

We observed that the dependence of the fraction of blocks mined in the longest chain on hash power decreased compared to Experiment 1. This change is attributed to the influence of network delay, which becomes more significant when the inter-arrival time (I) is reduced

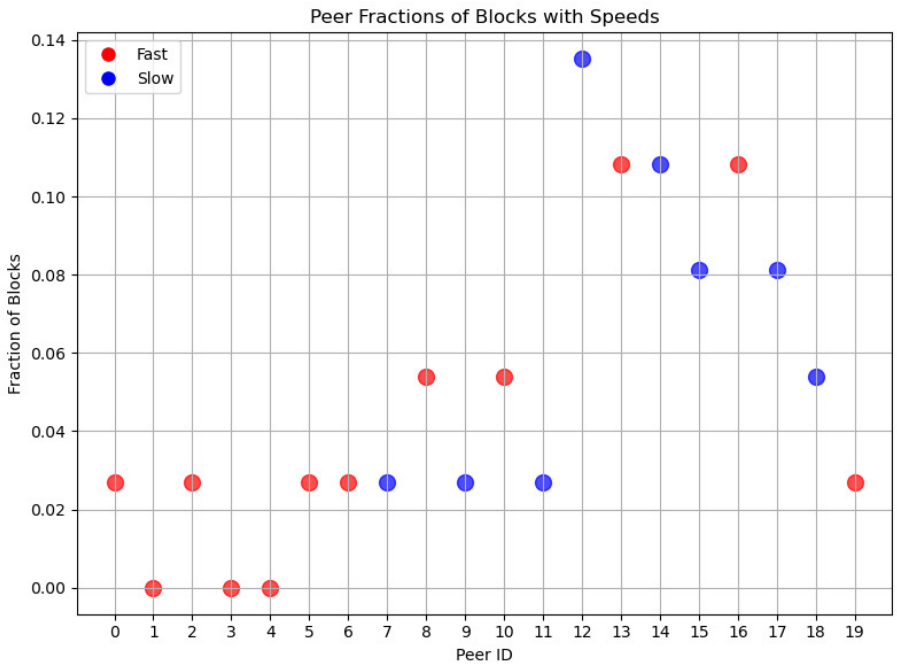


Figure 2: Statistics for Experiment 1

Experiment 3

We further reduced T_tx and I, and we expected our results to be entirely dependent on network delay.

- $N = 20$ - Number of nodes in the network.
- $Z0 = 40$ - Percent of slow nodes"
- $Z1 = 40$ - Percent of low cpu nodes
- $T_{tx} = 0.1$ - Mean Transaction time in seconds.
- $I = 0.2$ - Mean Inter-arrival time of transactions in seconds.

Results

- **Number of blocks mined:** 50
- **Number of blocks in the longest chain:** 12

High Hash Power	Fast Node	Block Fraction
True	True	0.72
True	False	0.27
False	True	0
False	False	0

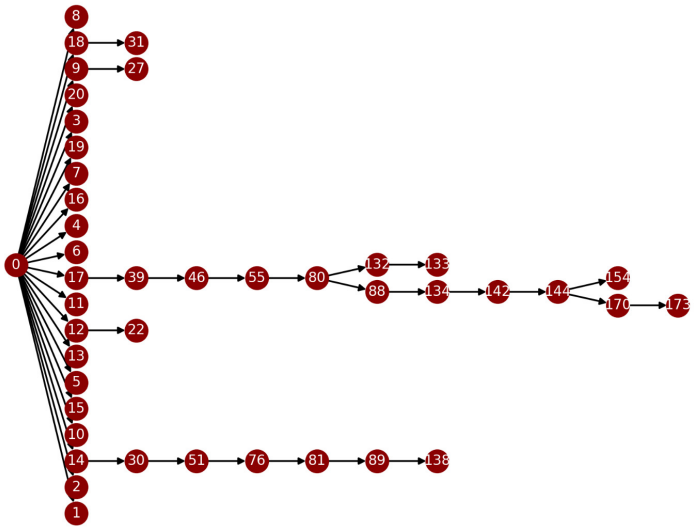
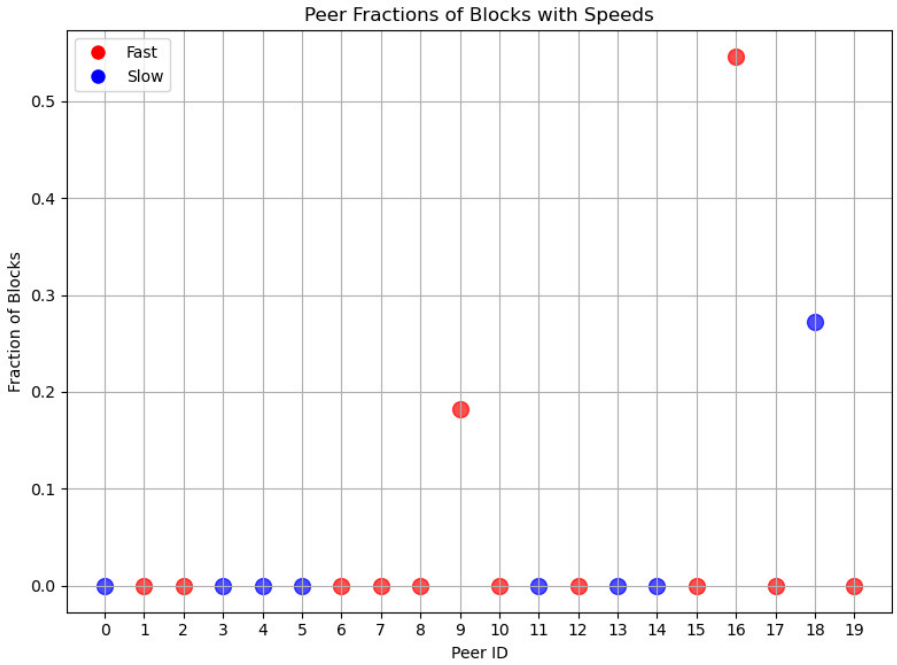


Figure 3: Block Chain Tree for Experiment 1

The forks in the above tree were very high, attributed to the extremely low values of T_tx and I.

Statistics

We observed that only a few peers mined all the blocks in the chain. This is due to their central position in the network, allowing them to experience lower network delays compared to others, making it easier for them to mine blocks



Experiment 4

In this experiment, we attempted to reduce the number of peers by half compared to Experiment 1. We anticipate similar results to those observed in Experiment 1

- $N = 20$ - Number of nodes in the network.
- $Z0 = 40$ - Percent of slow nodes"
- $Z1 = 40$ - Percent of low cpu nodes
- $T_{tx} = 10$ - Mean Transaction time in seconds.
- $I = 1000$ - Mean Inter-arrival time of transactions in seconds.

Results

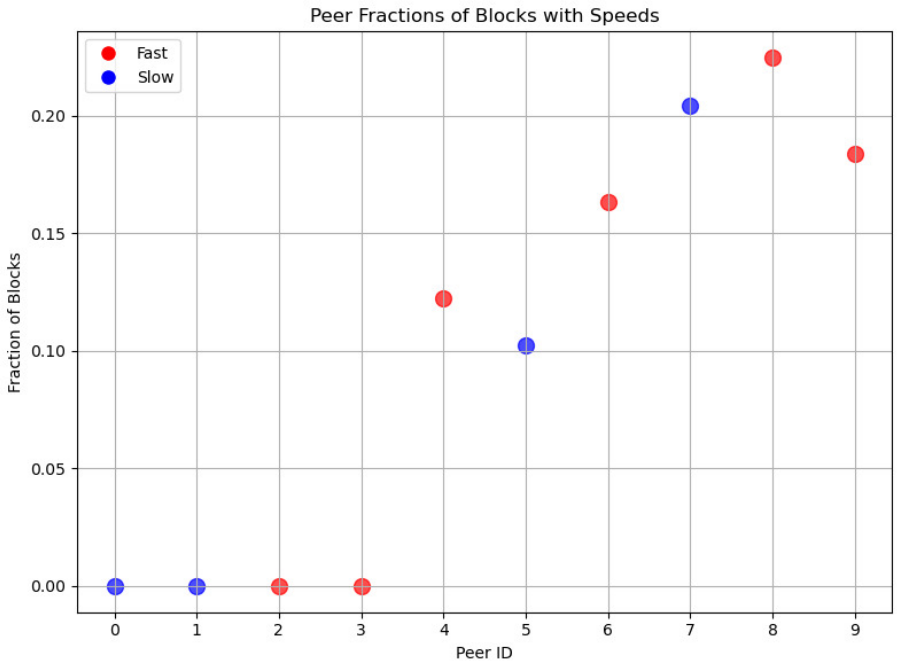
- **Number of blocks mined:** [Provide the actual number]
- **Number of blocks in the longest chain:** [Provide the actual number]

High Hash Power	Fast Node	Block Fraction
True	True	0.69
True	False	0.31
False	True	0.00
False	False	0.00

Figure 4: Block Chain Tree for Experiment 1

We can conclude that the number of peers has no significant effect on the longest chain when the inter-arrival time (I) is sufficiently large compared to network delay

Statistics


Peer Fractions of Blocks with Speeds
Fast
Slow
Fraction of Blocks
0.00
0.05
0.10
0.15
0.20
0
1
2
3
4
5
6
7
8
9
Peer ID