REPORT- PROJECT PART 1

Maloth Maheer	180050054
Bhukya Varun Tej	180050025
Bandameedi Harshvardhan	180050020

2) What are the theoretical reasons for choosing the exponential distribution?
Ans:
If the probability that the event occurs during a certain time interval is proportional to the length of that time interval, then the time T we need to wait before an event occurs has an exponential distribution.

Mathematically,
Consider a small time interval t, with **β** as a constant proportional to total hashing power. Then,
Probability of mining a block in t time = **β**.t

Probability that the next block is mined after n.t = $(1 - \boldsymbol{\beta}.t)^n$
Substitute n.t = T. then,

Probability that the next block is mined after T = $(1 - \boldsymbol{\beta}.T/n)^n$
Now, as t→0, n→∞,
lim(probability) i.e $\lim\limits_{n \to \infty} (1 - \boldsymbol{\beta}.T)^n$ tends to $e^{(-\boldsymbol{\beta}.T)}$
Hence,
Probability that the next block is mined after T = $e^{(-\boldsymbol{\beta}.T)}$
Hence, the interarrival between transactions generated by any peer is chosen from an exponential distribution with mean **β**

4) Each peer is connected to a random number of other peers, such that the resulting network is connected. Correct random sampling along with justification of chosen distribution.

Ans:

We used power law distribution forming connections in peer to peer networks.

This is the distribution most close to that used in the bitcoin network.

The degree distribution P(k):

$$P(k) \sim c.k^{-y}$$

with c normalizing constant and γ parameter of the distribution known as scaling parameter and k is the degree of the node.

The algorithm is as follows:

- We assumed each peer to have at least "m" edges.
- For the first m+1 peers, we ensured there is full connectivity.
- For the next nodes, we are calculating probability of connecting to the existing nodes by Πi = ηi.ki/ Σ(ηj.kj)
- Where ηi is the strength of a node (assume high value for fast node and low value for slow node. Both greater than 0).
- Then we choose m nodes based on the probability we calculated.

This ensures connectivity in the network.

Link to research paper : https://arxiv.org/pdf/1804.02350

5) Why is the mean of D_{ij} inversely related to C_{ij}? Give justification for this choice.

Ans:

C_{ij} denotes the link speed between nodes i and j. D_{ij} is the delay at the receiver's end. If the link speed is high, then the delay will be low and vice versa.

6)

Ans:

Loopless forwarding: This is ensured by the forwarded attribute of block and peer class. This forwarding table is initialized with false for every pair of nodes and after one flow of transaction/block it is made True. So next time before forwarding when we consult the table, we get True value and the flow is not repeated.

7)

Ans:

We have kept the value of T_k as 10000ms. This is because our latencies for various blocks were coming in the order of 200-300 ms depending on the block size. So we kept the mean of T_k for all blocks around 50 times that of block latency. This mean represents the hashing power of a peer and we have changed it several times for observations.

Fork resolution:
Forks do happen in our implementation. But they are resolved as per the rules of the longest chain. After every block generation, a peer waits for some time(given by exponential dist) and confirms that the chain on which he had mined the block is indeed the longest chain. Only then does he add the block to his tree and broadcast this. We only abort mining in case a chain of higher length arrives, so that takes care of the conflicts in case another chain of same length arrives,

8)
Ans:
We are showing the trees at the end of simulation and the observations regarding the blockchain tree are in “Peerx_Analysis.txt” for a peer with id x and the file containing time of arrival of various blocks is “Peer x blocks.txt”.
Inferences:

- Upon increasing “n” more blocks will be generated in every peer’s blockchain tree.
- Increasing T_{tx} will not affect the number of blocks or branching. Just that the size of the blocks will be more and it’ll take more time to forward a block
- In general if we decrease T_k, we are basically increasing hashing power, so more blocks will be generated in the network and each node will have more blocks in the tree, also in the longest chain.

    Also if all blocks have similar values of T_k, branching will be high across all nodes as all nodes have similar mining power. On the other hand if one block has higher value of T_k, then branching will be low as the blocks generated by that node will dominate in every peer’s tree

- If a node is slow, it’ll take more time for transactions and blocks to reach it. So it might happen that a fast node has more blocks in its tree than a slow block. These can be verified by altering the value of “z”.