CS 765 Assignment 1 - Report

Q2: What are the theoretical reasons for choosing the exponential distribution?

A2:
This follows from modelling the interarrival time in a discrete manner, with the discrete unit of time being Δ . We assume that the probability of mining a block in one unit of time is β*Δ, where β is a measure of the hashing power and I is the interarrival time. This is a realistic assumption. Hence, P(I=n*Δ)=β*Δ*(1- β*Δ)^(n-1) [since for the first n-1 intervals, no packet arrives and then one must arrive]. Hence, P(I>n*Δ)=(1- β*Δ)^n, which is a geometric distribution. Now, by putting x=n*Δ taking the limit Δ-->0, the geometric distribution transforms into the exponential distribution, P(I>x)=e^- β*x. Hence, theoretically, the probability distribution of interarrival time between transactions generated by any peer is an exponential distribution. Furthermore, the exponential distribution has several properties that simplify further analysis such as the memorylessness property, because of which the expected time till a packet arrives is 1/β regardless of how much time has already elapsed.

Q4: Correct random sampling along with justification of chosen distribution. You may need to reference an appropriate research paper here.

A4:
Research paper:
https://www.researchgate.net/publication/3235764_Building_low-diameter_peer-to-peer_networks

The paper suggests a way to connect a new peer to an existing network and shows that with a high probability the diameter of O(log N). It also ensures a completely connected network. Also the protocol tries to maintain the degree of each between [D,C+1], where C should be > 3*D+1. The idea is that there will be a cache which has K peers with degree >D but <C. For a new peer we randomly choose D peers from cache to pair with. If the peer in cache reaches degree C, then it will be replaced with another appropriate(method discussed in paper) peer.

Q5: Why is the mean of dij inversely related to cij?

A5:
We know that the arrival of packets is a Poisson process and hence the packets arrive in bursts. The queueing delay is proportional to the traffic intensity, which is inversely proportional to the link speed . Hence, the mean of the queueing delay is inversely proportional to link speed.

Intuitively as well, one can see that if the host's packet processing powers are assumed to be roughly the same, then a greater link speed would lead to a larger number of packets waiting in the queue on average (since arrivals are bursty). And a larger number of packets waiting in the

queue would directly lead to longer waiting times on average. These waiting times correspond to the queueing delay and hence the mean of dij is inversely proportional to cij.

Q7: Justify the chosen mean value for Tk?
(Note that smaller mean value for Tk is equivalent to saying k has more CPU power in a proof-of-work system)

A7:
As Tk increases, producing new blocks becomes harder. If Tk is very high, then the sizes of the blockchains generated on each node become very small, whereas very low values of Tk lead to high branching. Thus to simulate a network that is close to reality we should choose intermediate values of Tk, that are at least 5x the Ttx.

Q8:(refer below)

Q: Use an appropriate visualization tool to study the blockchain tree.
A: Networkx along with pydot is used to get blockchain trees as an image. The images are stored in the ‘trees’ folder. The value in images in the 'trees' folder represent block ID. Block ID may not be continuous in an image since we are using a global counter for block ID.

Q:
Experiment with choosing different values for different parameters (n, z, Ttx, Tdij , mean of Tk, etc.)

Find the ratio of the number of blocks generated by each node in the Longest Chain of the tree to the total number of blocks it generates at the end of the simulation. How does this ratio vary depending on whether the node is fast, slow, low CPU, high CPU power etc.?

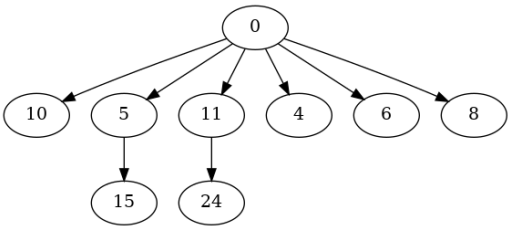
How long are branches of the tree measured in the number of blocks?

A)
n Analysis:
As n increases, keeping all other variables constant, the size of each blockchain decreases and hence the given ratio increases. This is because we are limiting the number of events we process and having a higher n implies we process a lower number of events for a given peer and hence a blockchain size decreases.

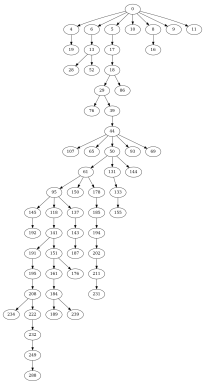
z Analysis:
In the regime of very small z, slow nodes are at a disadvantage in terms the ratio, whereas in the regime of large z, the distinction between slow and fast nodes vanishes (depending on the topology).

Ttx Analysis:
When we decrease Ttx, the size of the blockchain (no. of nodes in the tree on each node

diminishes rapidly, this is because most of the time of the simulation is spent on receiving and forwarding transactions (eid = 2), however as Ttx increases, the size of the blockchain increases significantly.



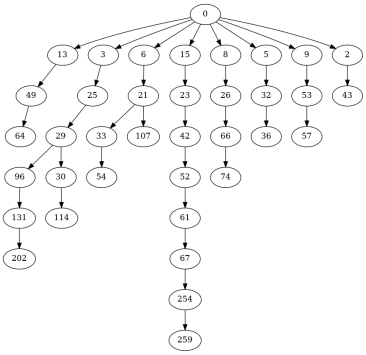
Low Ttx



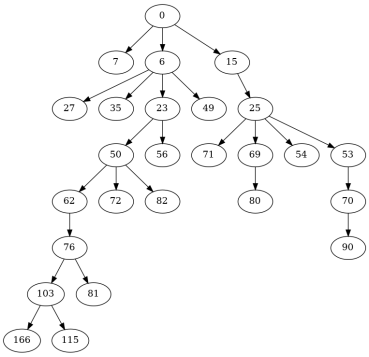
High Ttx

Tk Analysis:

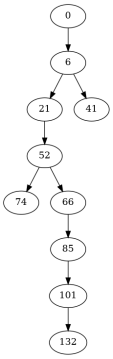
As the mean of Tk increases, branching in the tree decreases and the average length of each branch also decreases. This is because, for very small values of mean(Tk), every node can quickly generate blocks before the latest updates are propagated. Hence, for small mean(Tk), we are in the regime of high branching. But as mean(Tk) increases, blocks are harder to produce and hence we have less branching.



Small Tk



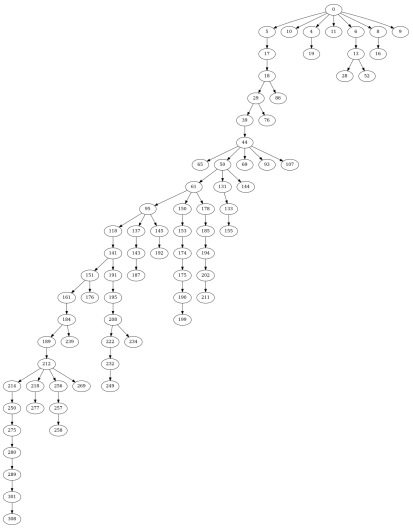
Intermediate Tk



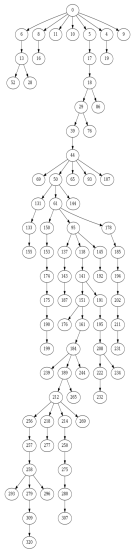
Large Tk

CPU power Analysis:

As the CPU power increases, the ratio of the number of blocks generated by each node in the Longest Chain of the tree to the total number of blocks it generates at the end of the simulation increases on average. This is because with higher processing power, you can be sure that a greater number of your mined blocks will be a part of the final blockchain, since you will mine them fast enough for them to have ample time to propagate.
Also, very high CPU power correlates with less branching in the final tree for the same reason.



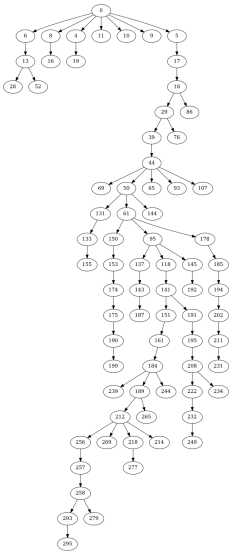
Low CPU Power



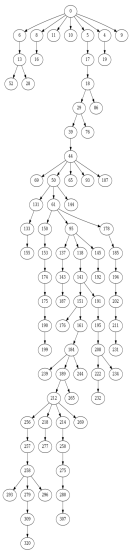
High CPU power

Fast/slow node Analysis:

Slow nodes will have a lower ratio of the number of blocks generated by each node in the Longest Chain of the tree to the total number of blocks it generates at the end of the simulation increases on average as compared to fast nodes if we keep the CPU power almost constant. This effect becomes more pronounced as CPU power increases and it is a function of the topology that we generate. This is because if in a given topology, enough fast nodes are connected to each other, then they can essentially lock the slow nodes out.



Slow, High CPU power



Fast, High CPU power

Branch Analysis:

Slow nodes have blockchains of shorter lengths and have greater branching than fast nodes. (Assuming z is high and the topology is appropriate so that fast nodes can communicate among themselves and forward and receive more blocks than the slow nodes).

Low Tk leads to higher branching as explained above.

If the node has significantly higher CPU power than the rest of the nodes, then it will have lower branching, however, if most nodes have similar CPU powers, then if this average CPU power is higher, we will have higher branching.