COL867 Assignment 1

Aayan Kumar - 2014CS10201
Shreyan Gupta - 2014CS10485
Aditi - 2014CS10205

13th February 2018

1 Implementation Details

- Refer to README.md in Assignment1 on github. *Click here*

2 Report Questions

- **Choosing exponential distribution for inter-arrival time of transactions**
  - Generating transactions with exponential distribution implies that transactions follow Poisson process, which is a very good approximation of the real scenario.
  - All peers act independently of each other.
  - The memory less property is followed, which means, the transaction generation does not depend on history of transactions generated.
- **Random sampling for choosing connections in network**
  - The network is generated randomly each time.
  - Firstly, a random spanning tree is generated to ensure graph is connected
  - Then, random edges are added to the graph to get desired density (can be modified)
  - The average number of connections can be set as a parameter in parameter.py
- **Reasons for considering the bottleneck speed**
  - Communication over a real P2P network takes place using packets as a unit.
  - Messages as large as a block ($\tilde{1}$MB) cannot fit into one packet, and thus must be split up into multiple packets. This is the reason why only Blocks have the bottleneck speed component.
  - Due to bandwidth limitations, the max speed with which packets are transmitted is limited by the bottleneck speed. In the best case scenario, the minimum of the speeds of sender and receiver is the bandwidth possible.
- **Why does d_{ij} depend on c_{ij}**

  - Queuing delay (d_{ij}) depends on the bottleneck link speed (c_{ij}) because the bottleneck link speed determines the number of packets that will be sent for a block, and the amount of content in one packet.
  - The queuing delay will depend on the average number of packets in the queue, as well as the size of each packet.
  - It is inversely proportional to the bottleneck speed, since for the same number of packets, lower bottleneck speed will lead to longer queues on average, hence increasing the queuing delay.

- **Choice of mean of T_k**
  - The choice of T_k is dependent on 2 factors, the transaction generation time and the Network latency
  - We've experimented with the dependency of T_k and the transaction generation time. We found that if txn genertion mean was comparable to T_k, then the blocks created had a very small number of (even zero) transactions.
  - When the value of T_k was comparable to the Network latency, then there was a lot of forking and branching in the blockchain created as before a block could be reveiced by the peers/nodes, another block was created and circulated.
  - The Network latency also depends on the density of the network of peers/nodes. If the network is dense, ie. there are a lot of connections per peer, then the forking goes down.
  - The visualization of all these experiments is shown in the observation section.

3 Observations

- We experimented with different block generation mean times, keeping network delay constant, on a network of 50 peers & average 20 neighbors per peer. If block generation time is comparable to the network delay, then the blockchain tree that would be generated would have a lot of branches. This is because, while a generated block is being propagated to all the peers, more blocks are generated. This creates a large amount of forking. On the other hand, if T_k is much higher than network delay, we observe very less forking.

Figure 1: $T_k = 10$, Txn generation Mean = 5

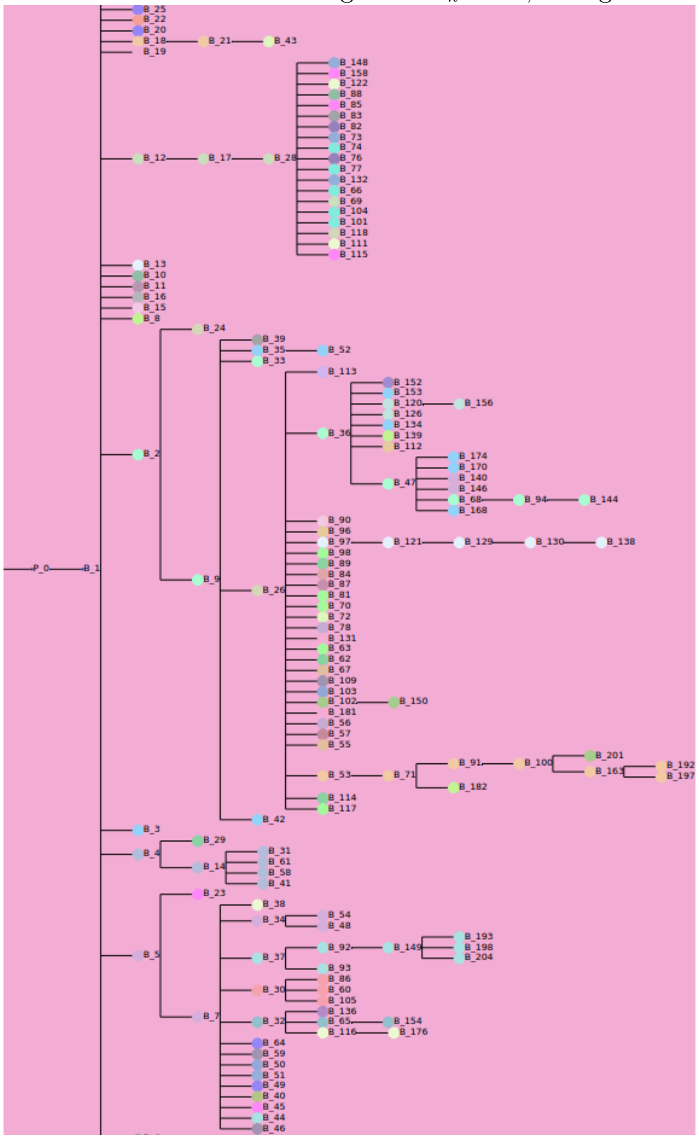


Figure 2: $T_k = 50$, Txn generation Mean = 5

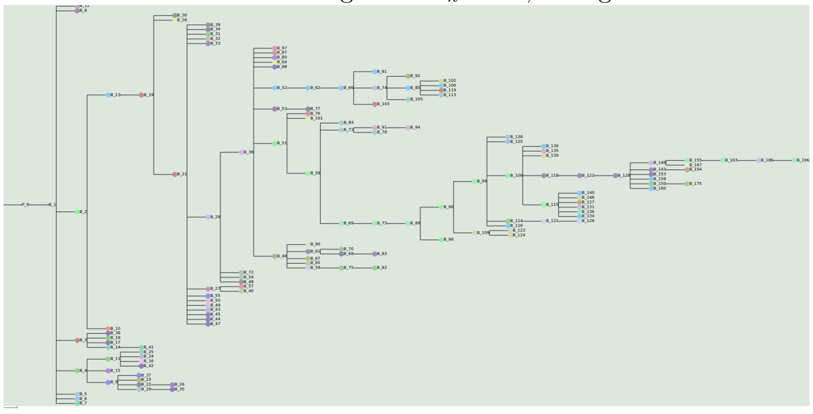
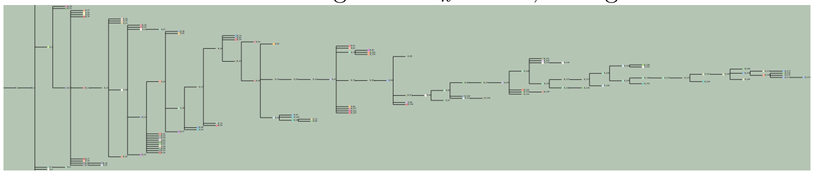


Figure 3: $T_k = 200$, Txn generation Mean = 5



- We did experiments with graph density by keeping high T_k. If the peers are sparsely connected, then the time required to propagate the blocks can be large, leading to more branches. But with dense network, we see less branching.

Figure 4: $T_k = 10$, Txn generation Mean = 5

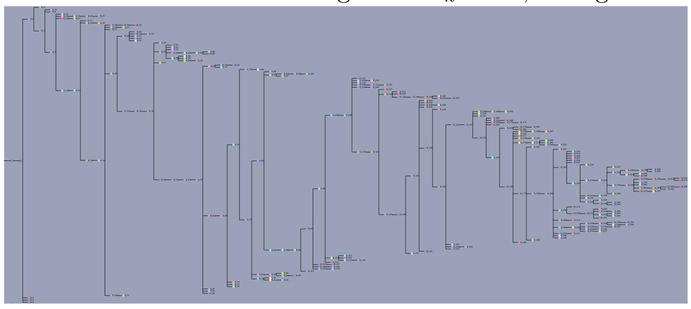


Figure 5: T_k = 50, Txn generation Mean = 5

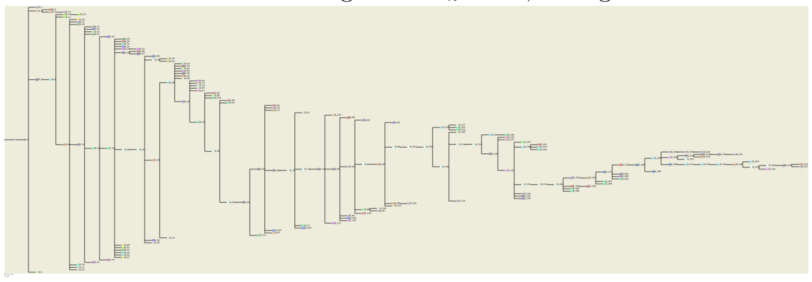
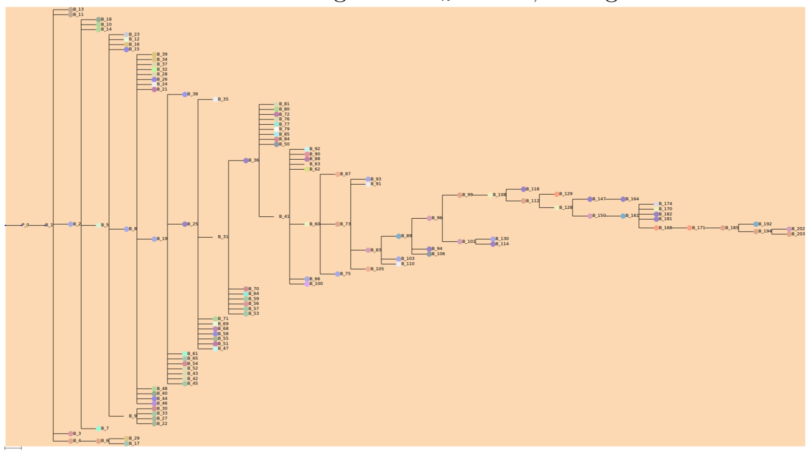


Figure 6: T_k = 200, Txn generation Mean = 5



- We also experimented with the values of T_k for each peer. We found out that the peer with very high computation power (inversely proportional to T_k) is the one whose blocks eventually form most part of the longest chain. T_k was divided by 1.5^{PeerID} for each peer.

Figure 7: 4 Peer Network : Exponential T_k

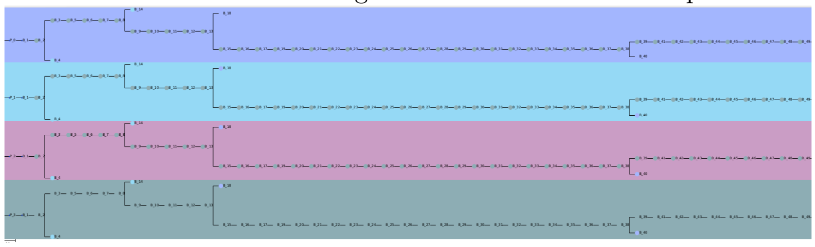
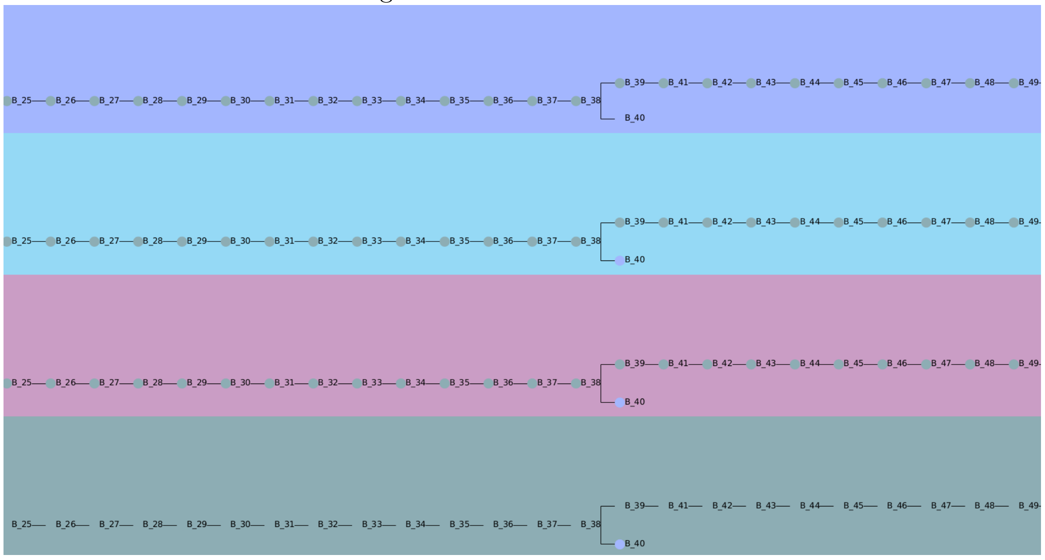


Figure 8: 4 Peer Network : Zoomed In



- Eventually, all the peers converge on the block chain tree structure and only the latest blocks (blocks closest to the current_block of each peer) are different.

Figure 9: Initial Block Trees for All Nodes

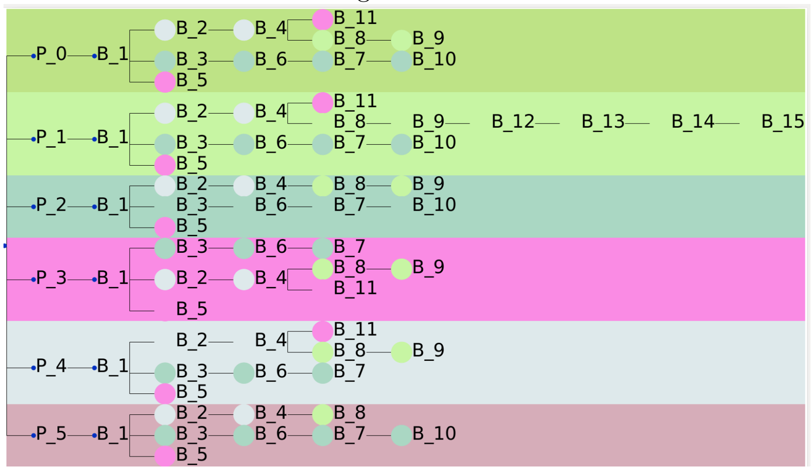
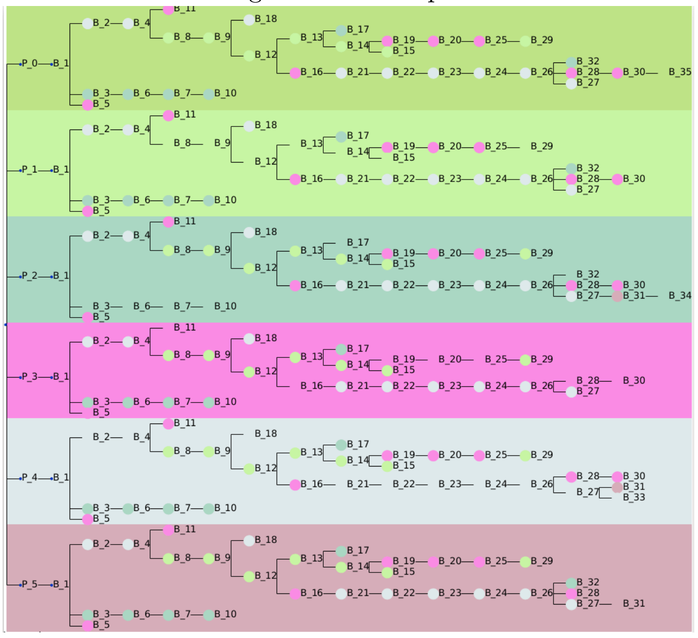


Figure 10: Subsequent Block Trees : Converged till B29/B26



- We also experimented with different ratios of slow nodes in a network of size 6. We observed more forking with increasing % age of slow nodes, justified by the fact that the average network delay is increased with more number of slow nodes.

Figure 11: 0% Slow nodes

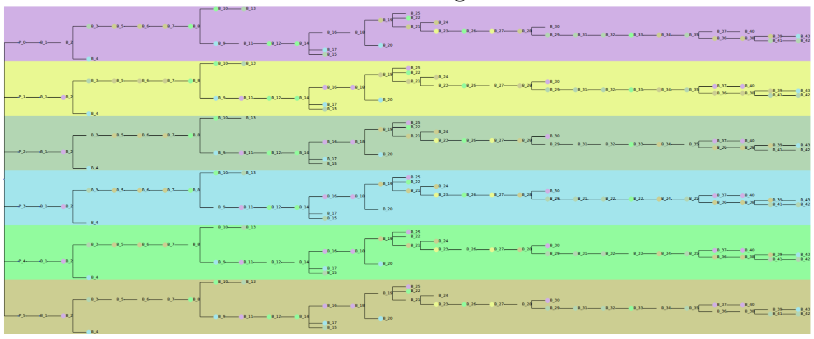


Figure 12: 50% Slow nodes

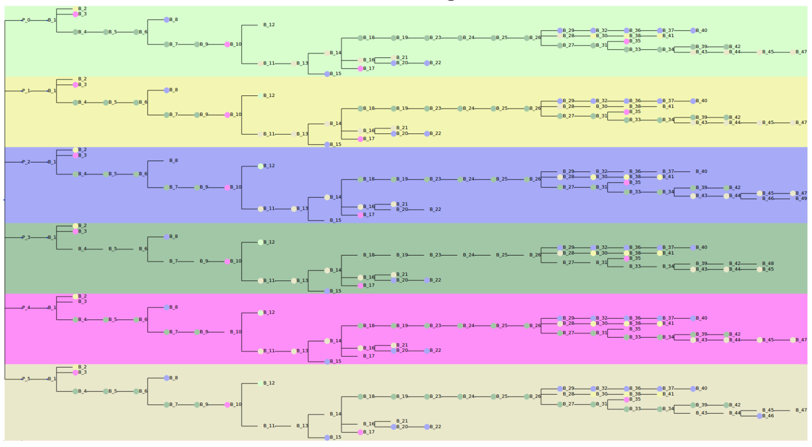


Figure 13: 100% Slow Nodes