High Speed Networks: Homework #1

Due on February 13, 2018 at 11:55pm

Prof. Vinay Ribeiro

Akshit Goyal, Ayush Bharadwaj, Nikhil Gupta

2014CS50278, 2014CS10091, 2014CS50462

Problem 2

The inter-arrival between transactions generated by any peer is chosen from an exponential distribution whose mean time can be set as a parameter of the simulator.

Solution

The reason for choosing an exponential distribution is that it is the only distribution with lack of memory property i.e after waiting a minute without a transaction, the probability of a transaction arriving in the next two minutes is the same as was the probability (a minute ago) of getting a transaction in the following two minutes. The chance of something happening neither increases nor decreases with waiting time. It is because of these Poisson arrivals that the inter arrivals become independent and identically distributed. The number of arrivals in time interval of size T becomes Poisson distributed.

Problem 4

Each peer is connected to a random number of other peers. Provide justification of the chosen distribution for random sampling.

Solution

The network has fast nodes and slow nodes which is decided by a probability parameter Z. Now for forming the connections in the network, we assign a connection number to each of the nodes which is the minimum number of connections needed for that node in the graph. The connection number of fast nodes is taken to be $1.5 * \log n$ and the connection number for slow nodes is $\log n$. The reason for choosing this is that $\log n$ gives good connectivity in a peer to peer network and we want more connections for a fast node than the slow node which should be significantly less than n and so the connection number for fast nodes is chosen to be $1.5*\log n$.

Now, for forming the connections, we choose a node and check it's type (fast or slow) and then choose another node randomly. Now, depending on the types of the pair of nodes, we have different probabilities assigned for different connections which are fast-fast - 0.1, slow-slow - 0.2, fast-slow - 0.5. It can be justified by the fact that fast nodes would be geographically distributed with slow nodes near them having higher probability to link with them. So, depending on these probabilities, we make or do not make an edge between the two. This process is repeated until the connection numbers of all the nodes is satisfied. In the end, we check whether the graph is completely connected by a DFS traversal and if it is not, we form the graph again from scratch as graph generation is time efficient and also in most cases, the graph generated is connected.

The above mentioned random sampling of the connections between peers is done by randomly choosing random nodes and so randomness is ensured. Moreover, the parameters are set such that connection numbers of fast nodes are high and probability of forming edges between fast-fast nodes is the highest followed by fast-slow and slow-slow. This ensures the connections generated are such that the network remains fast and it takes lesser time to go from one node to another.

Problem 5

1. Why is the bottleneck speed considered while simulating latencies between pairs of peers?
2. Why does the mean of the exponential distribution of the queuing delay on the path depend on the bottleneck speed?

Solution

Part One

Latency is the time between which a message m was transmitted from sender i and received by another node j. Now, the bottleneck speed between 2 nodes depends on the maximum flowing speed which can be achieved between the nodes which is the minimum of all allowed speeds in the path. The latency between the nodes will be inversely proportional to this bottleneck speed as the time taken will increase if the maximum flowing speeds decreases in the path from i to j.

Part Two

Queuing delay is the sum of the delays encountered by a packet between the time of insertion into the network and the time of delivery to the addressee. Now, delays will be encountered by the packets at nodes where they will be queued because the delivery speed of those packets might have been greater than the transmitting speed of that node. Now, minimum transmitting speed is the bottleneck speed of the path and and if bottleneck speed is less, we should expect queuing delays to be more which is achieved by setting the mean of the exponential distribution to be inversely proportional to the bottleneck speed. Queuing delays would be more if bottleneck speed is less as the packets would have to wait at most of the nodes if bottleneck speed becomes less and this adds to the delays. Thus, the dependence of the queuing delay on the bottleneck speed is explained.

Problem 7

Peer k generates a random variable T_k on hearing a block at time t_k. T_k is chosen from an exponential distribution with some mean which can be set as a simulation parameter. Justify the chosen mean value for T_k. Note that smaller mean value for T_k is equivalent to saying k has more CPU power in a proof-of-work system.

Solution

Every network has a maximum propagational delay which is related to the maximum latency between any pair of nodes i.e the maximum time taken to broadcast a block. Now, an ideal scenario for a blockchain system would be that no new block is generated within the propagational delay when a block is being broadcasted. For this to happen, the T_k should be much greater than the propagational delay as the probability of block generation is equal to $1 - e^{-\lambda p}$ where p is the propagational delay. This means that probability of block generation would be low if $e^{-\lambda p}$ is large which will happen when λ is very small. λ small means that mean is large since mean $= 1/\lambda$. So, the mean is chosen to be very large than p.
Thus, the mean value is chosen such that the above condition is satisfied which leads to lesser branching as probability of newer blocks being generated during block broadcast becomes less. Also, on receiving a block, the new T_k values are generated from a exponential distribution with mean as the mean received from the original exponential distributin for that peer. This is to make sure that a peer with high computational power generates a smaller T_k everytime.

Problem 8

Each node maintains a tree of all blockchains heard since the start of the simulation. The node stores the time of arrival of every block in its tree. This information is written to a file at the end of the simulation.Use an appropriate visualization tool to study the trees of different nodes (suitable choices can be gnuplot, matlab, or any other visualization tool). Experiment with choosing different values for different parameters (n, z, mean of transaction interarrival etc.). Summarize the structure of the tree for different types of nodes (fast, slow, low CPU, high CPU power etc.). How long are branches of the tree? Give detailed insights to explain your observations.

Solution

We will do multiple simulations varying the number of nodes in the network, number of slow nodes in the network, inter-arrival time as well. We will also observe the differences between the blockchain network recorded by different types of node in the network i.e slow-fast nodes and the nodes with low-high CPU power.

In Fig.1 and 2, we are looking at the trees produced by nodes different in the network speed and CPU power.

- In general, the nodes with high network speeds should have slightly greater number of blocks registered. This can be explained as for every fast node, a large number of transactions and blocks are received. This happens because the network latencies are lesser. However, as the latencies are more in slow nodes, we will have slightly lesser blocks. The difference is visible only towards the end of the simulation and the difference in the blocks is very less. It can be seen in Fig.1 and 2 that the difference is not significant.
- There should be no change in the structure of the tree as structure should be maintained. We are having nearly the same blockchains for the two nodes because the parameters are chosen such and maybe the simulation time could be small. The only change which happens should be in the end of the tree because we break at a time and due to network latency, some blocks do not receive the last few blocks and the slower nodes have greater possibility of missing the end blocks and so can have lesser blocks but the structure remains the same.

In Fig.3, 4, 5, and 6, we have changed the number of peers and increased it to 40 in the network and have generated the trees for nodes having different network speed and CPU power like we did previously.

- The two points mentioned in the previous observations hold when we have increased the number of peers from 20 to 40. This can be easily seen by seeing a slight increase in the number of blocks in case of nodes with no change in the strucutre of the blockchain.
- It is also visible that as the number of nodes increase, the number of blocks in the tree increases. This happens because as the number of nodes increase, the block generation by all of them add to make the number of blocks larger in this case.
- In addition, as the number of nodes increase, the branching in the block tree also increases. The reason for this is that as larger number of nodes means larger time for a block to travel from one end of the network to another. This will mean that each node would accept a number of blocks before finding out the block which extends it's longest chain. As a result, we have increased branching when the number of nodes increases.

In Fig.7, we changed the mean of the exponential distribution of the inter-arrival time between transactions and observed the change in the block tree generated in a node.

- The structure of the block tree depends on factors like latency in the network, frequency of generation of blocks and none of the factors change on changing the mean of the inter-arrival time. So, the block tree structure should not change with change in this parameter. This is why the tree structures are nearly same for the three cases above. The changes that are visible in Fig.7 are not significant and are because of the fact that these results are for different runs and for the same node so the block generations in the different runs is different because of the randomness in the system.
- However, the number of transactions in every block tree should increase and we verified this in our code.

In Fig.8, we changed the mean of the exponential distribution for T_k which is such that smaller mean value means higher computational power for that peer and observe the changes in the block tree of a node.

- As the mean value of T_k is decreased, we get more number of blocks in the block tree. The reason for this is that more blocks are generated within the same time interval which leads to more blocks being added in the block tree.
- In addition, with the decrease in the mean value of T_k, the branching in the tree also increases. This is because as the mean decreases, the probability of block generation within the propagational delay increases and as a result, we have more branching. Blocks are generated rapidly as a result of decrease in the mean and the conflicts in the block trees of nodes increases leading to more branching.

In Fig.9, we are varying the fraction of fast and slow nodes in the network. In our case, higher z means more faster nodes in the network. We will observe the changes in the block trees of a node by varying z.

- It can be seen that length of the longest chain in the tree increases with increase in the value of z i.e length of longest chain increases with increase in the number of fast nodes in the network. When z decreases, the number of slow nodes in the network increase and thus the time taken for the block to travel from one node to another increases. As a result, the length of the longest chain in the network would decrease and thus in general, the length of the longest chain in the block tree of a node decreases. This is what is being observed from our simulations as well. However, the difference is not that significant.

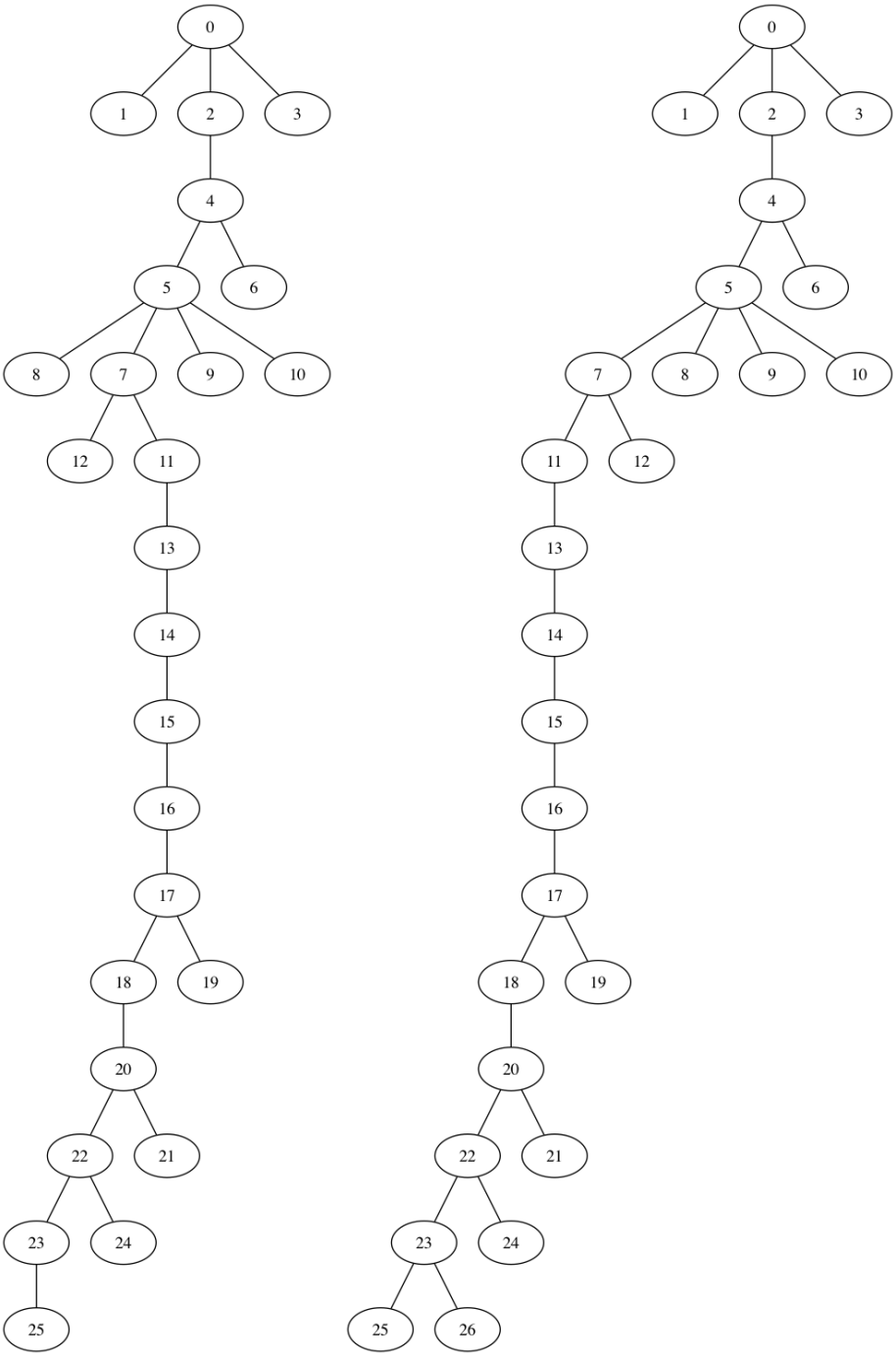

Figure 1: 20 peers, fast connection: high CPU power (left), low CPU power (right)

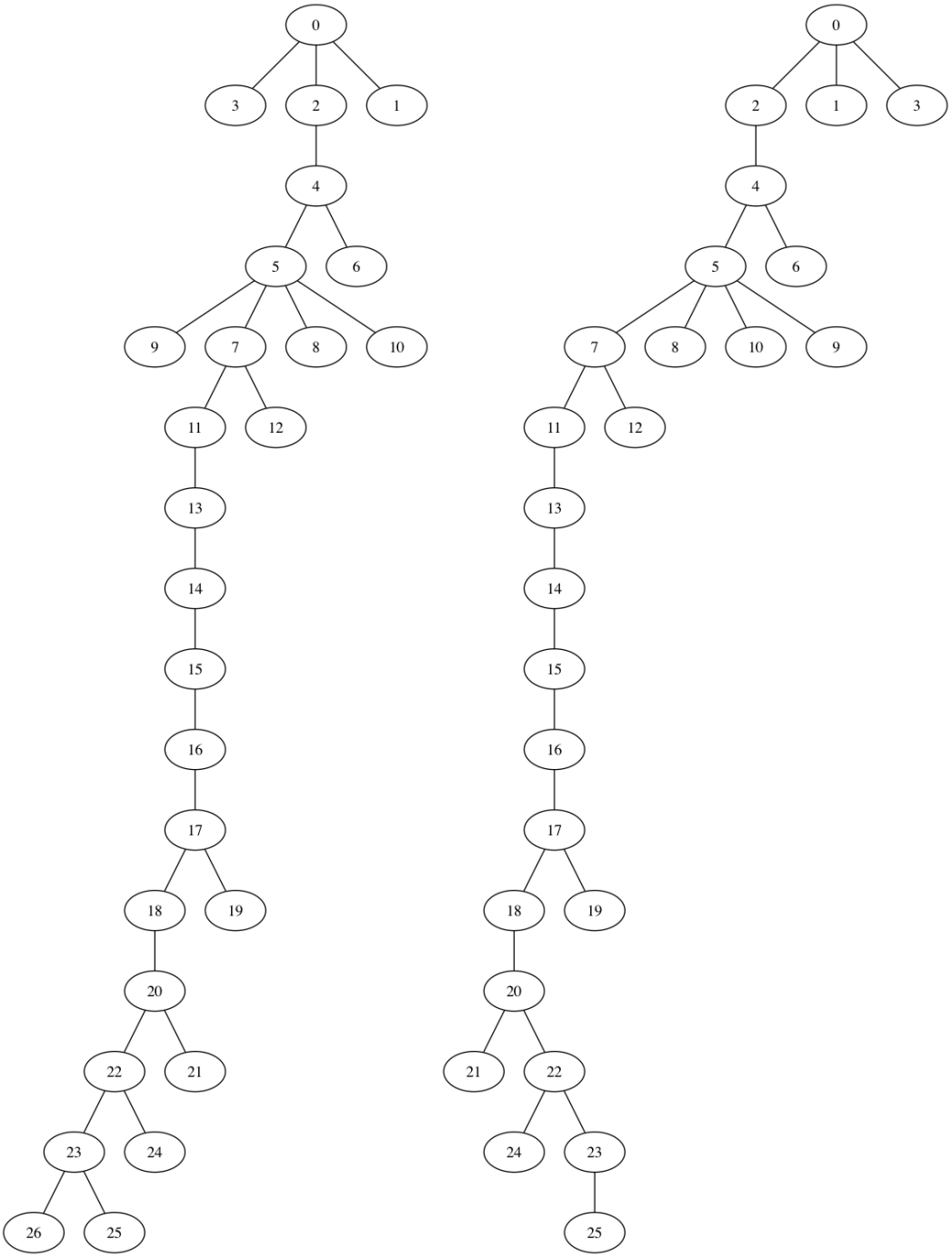

Figure 2: 20 peers, slow connection: high CPU power (left), low CPU power (right)

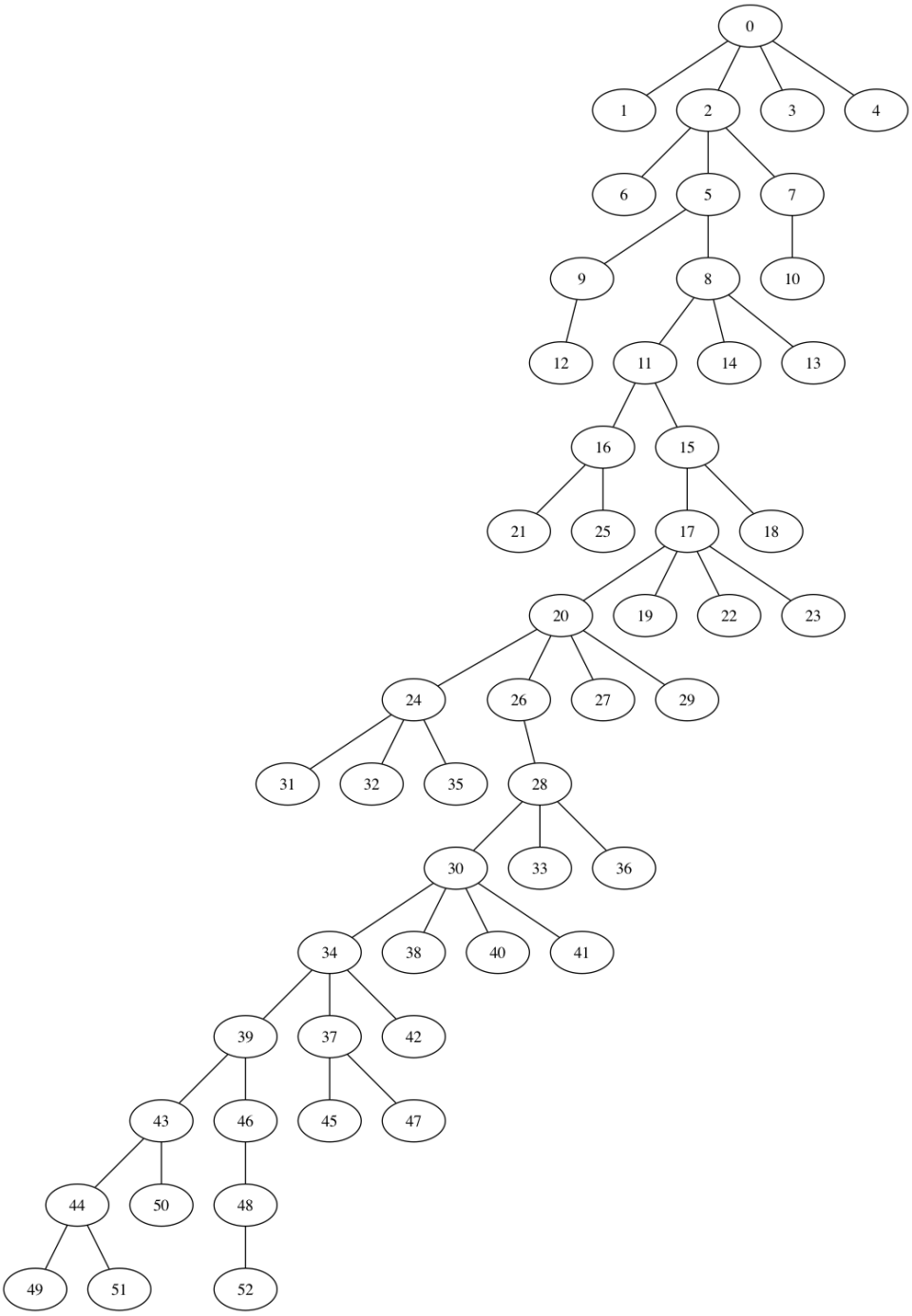

Figure 3: 40 peers, fast connection, high CPU power

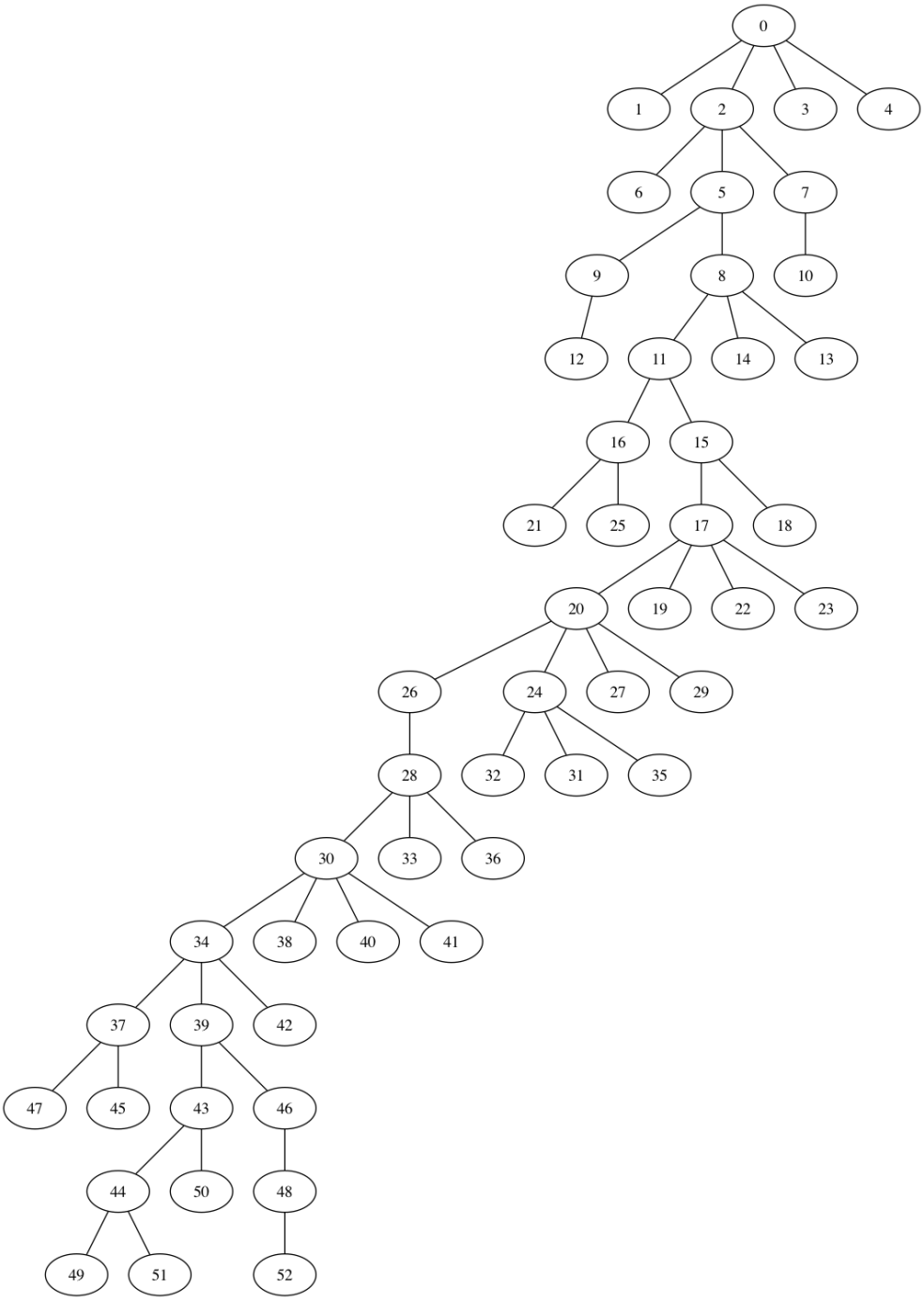

Figure 4: 40 peers, fast connection, low CPU power

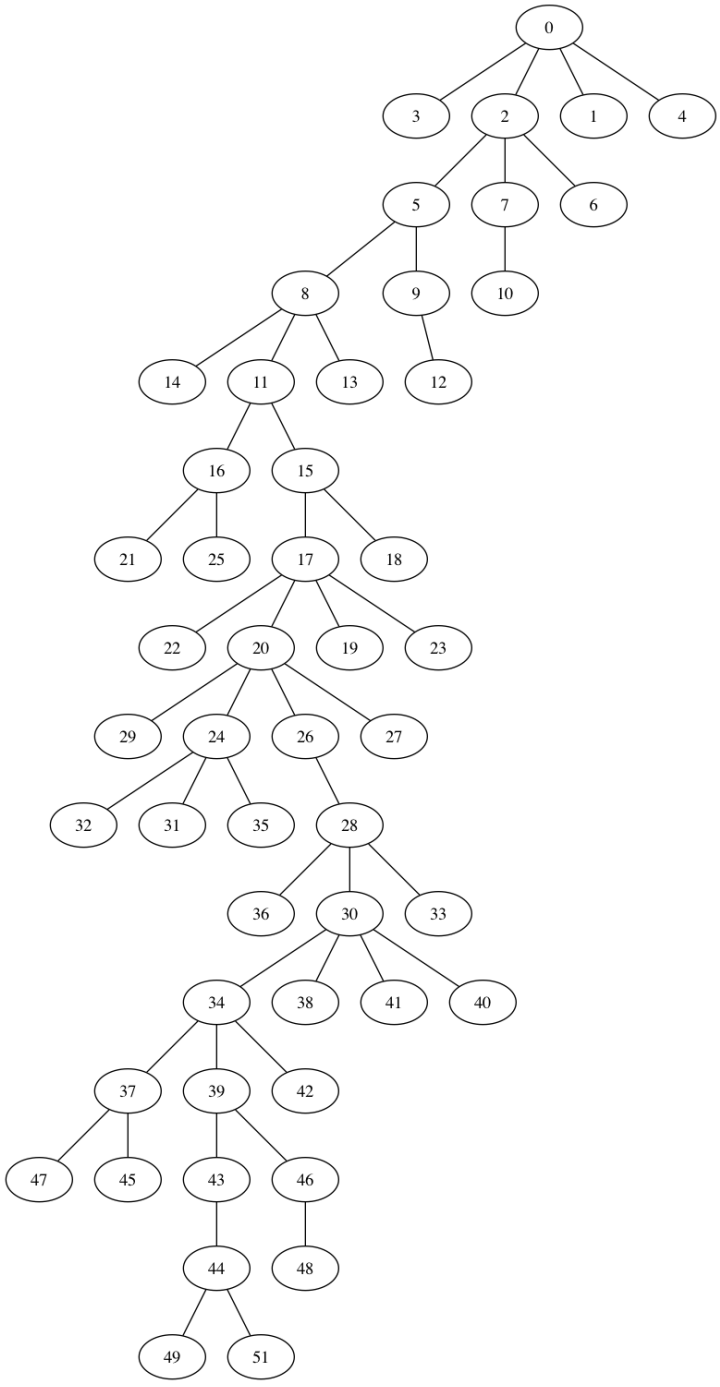

Figure 5: 40 peers, slow connection, high CPU power

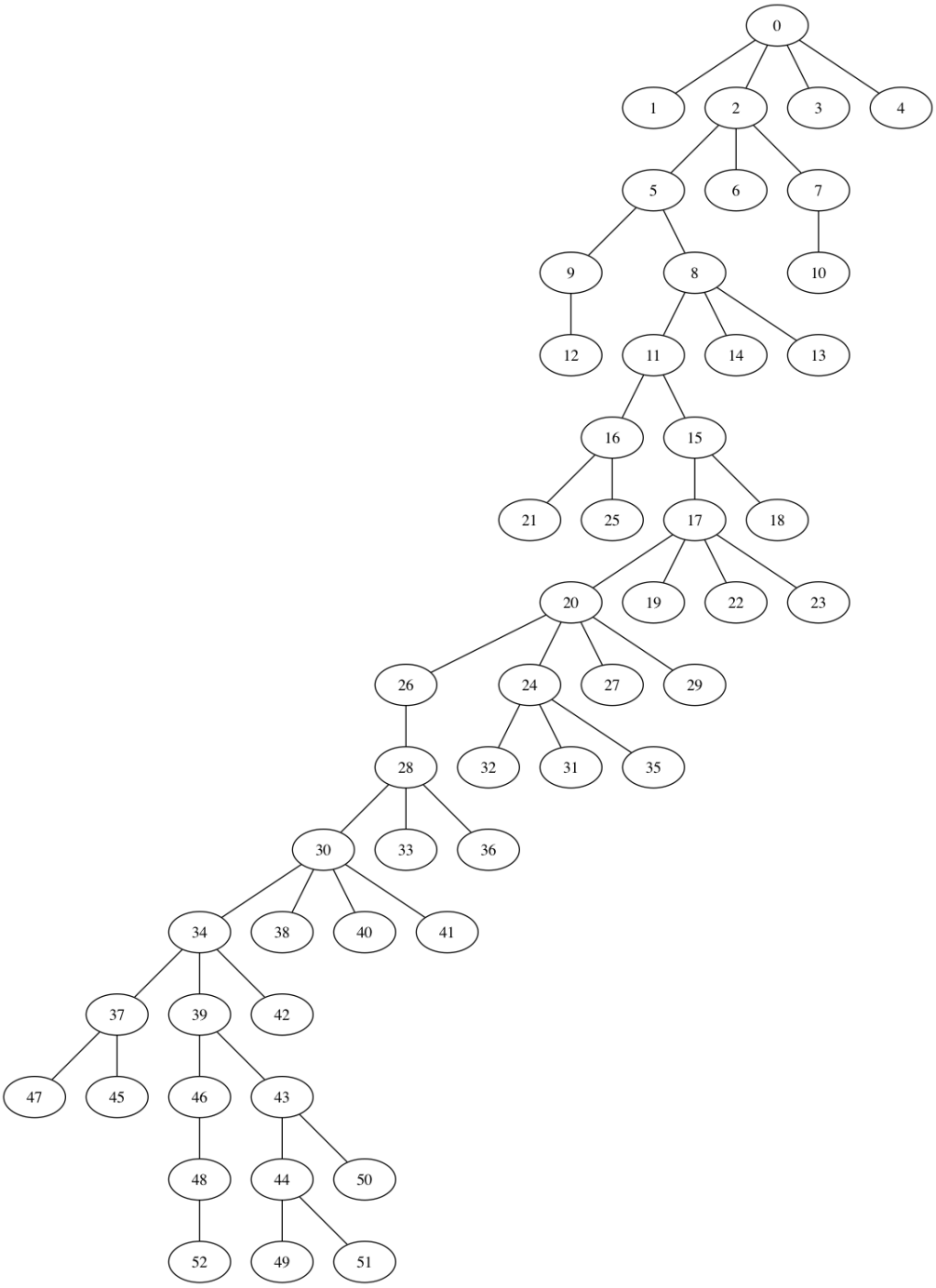

Figure 6: 40 peers, slow connection, low CPU power

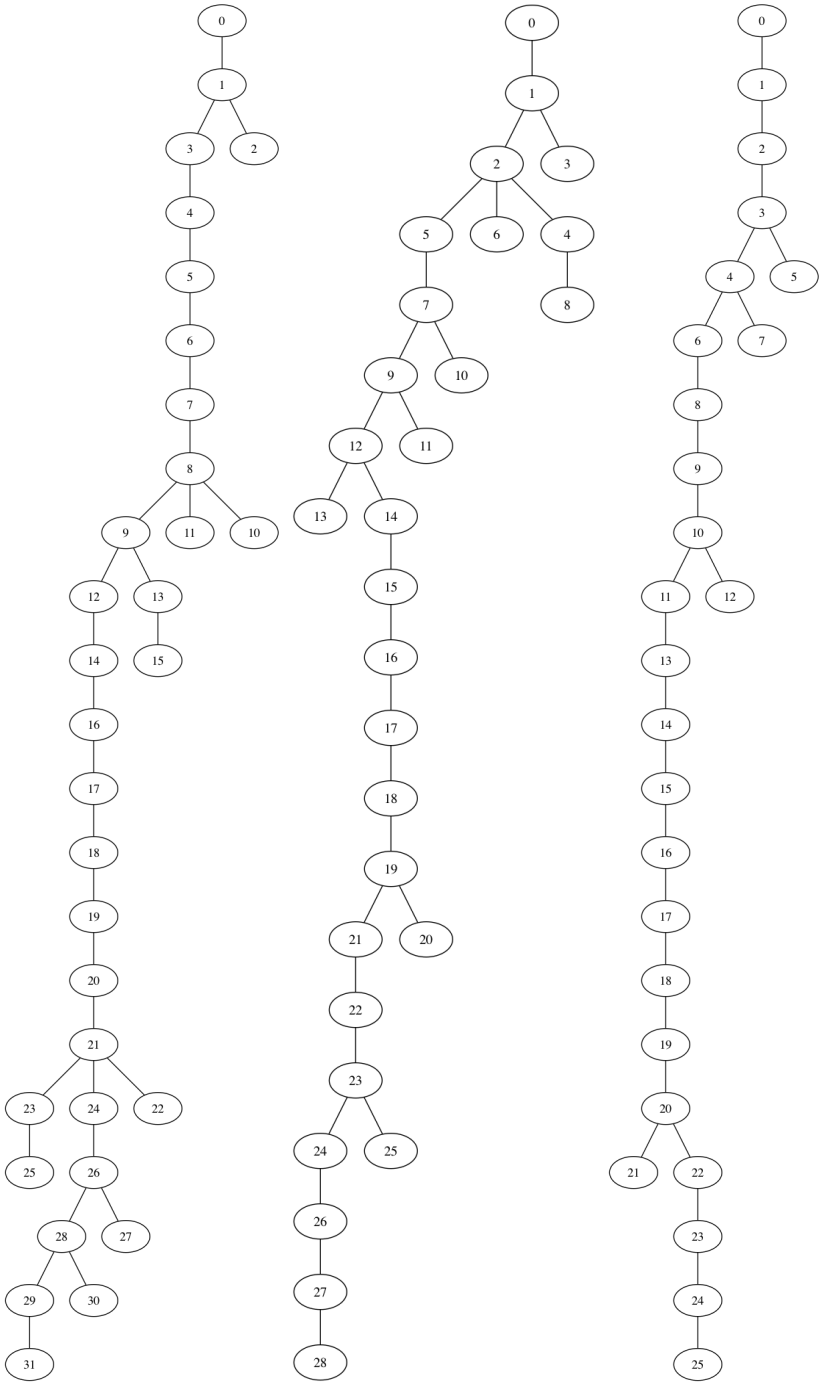

Figure 7: Comparing transaction mean: mean=0.05sec (left), mean=0.1sec (center), mean=0.2sec (right)

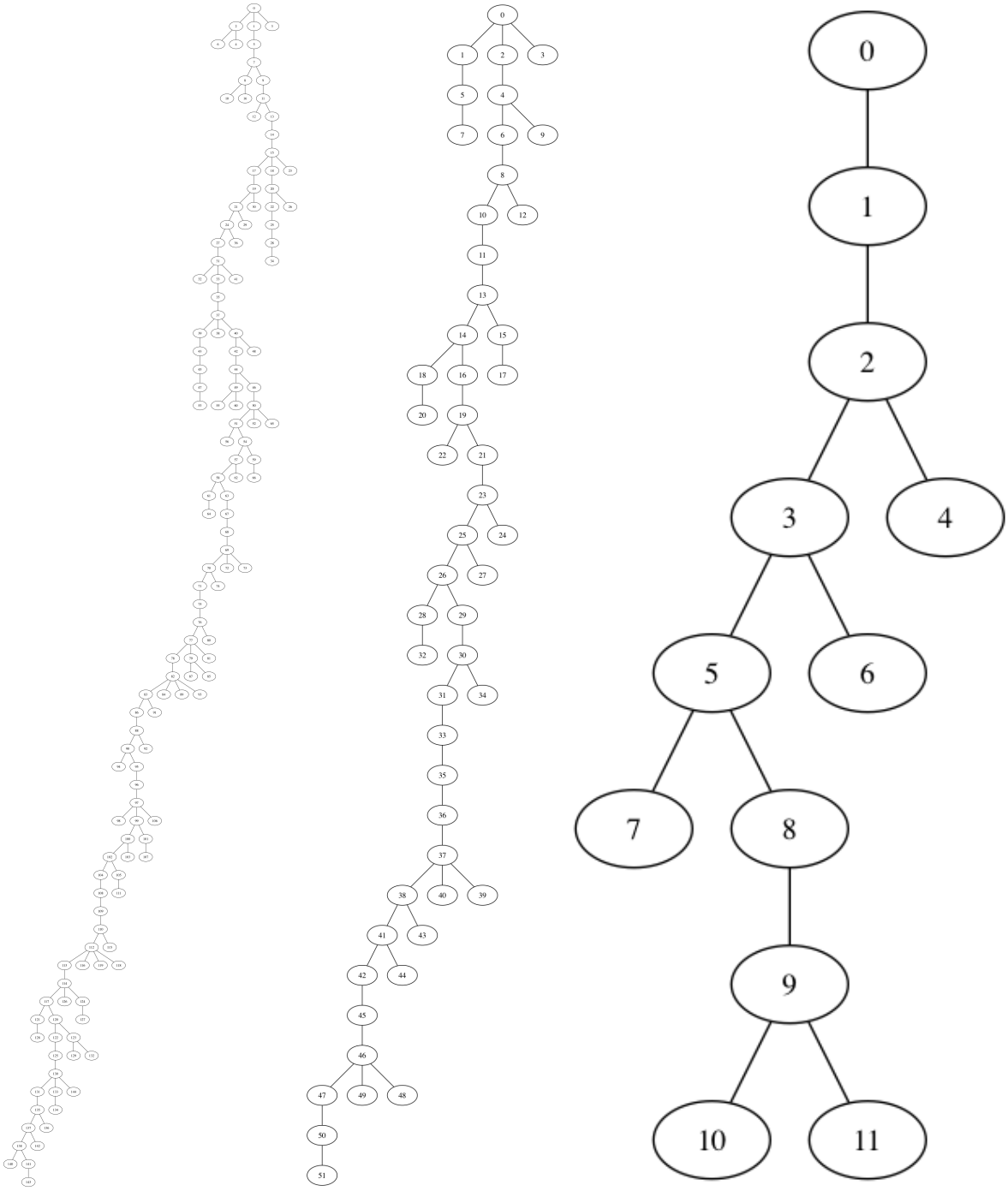

Figure 8: mean of the exponential distribution for T_k: mean=50sec (left), mean=150sec (center), mean=400sec (right)

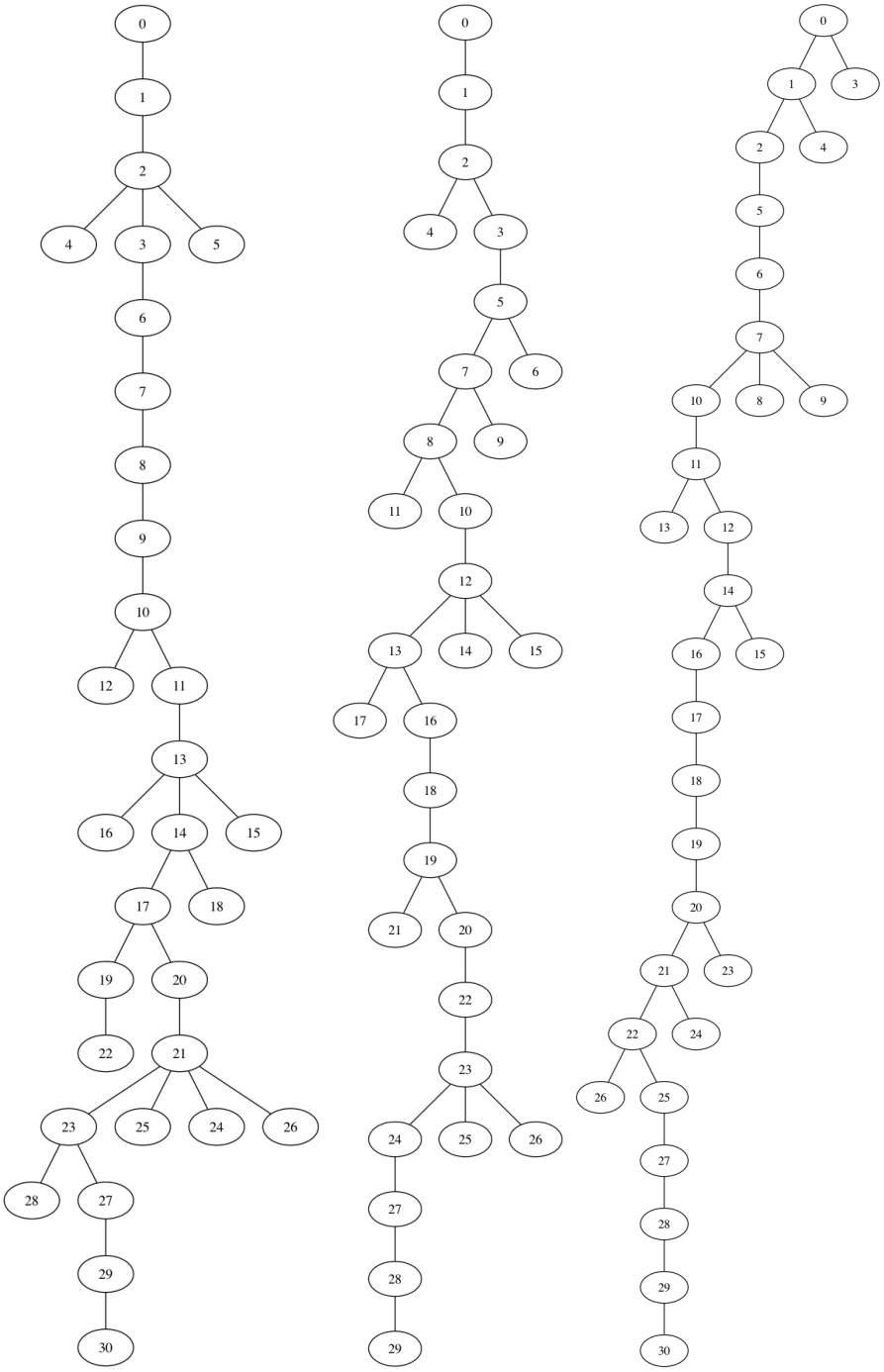

Figure 9: Comparing parameter z: z=0.2 (left), z=0.5 (center), z=0.9 (right)