CS 765 Assignment 1

Adithya Bhaskar (190050005) Danish Angural (190050028) Dev Desai (190020038)

February 14, 2023

1 Answers to questions raised in the PS

(a) *What are the theoretical reasons of choosing the exponential distribution? [ref. (2) in PS]*
The main reason for choosing the exponential distribution is its **memorylessness**. In particular, the probability of miners mining a block in a small interval δt of time is independent of them mining/not mining a block in the past. Therefore, the mining process is (a continuous version of) tossing a number of (biased) coins until we get heads. The time until success is modeled in the discrete case by a geometric distribution and in the continuous case by an exponential distribution.

(b) *Why is the mean of d_{ij} inversely related to c_{ij}? [ref. (5) in PS]*
The purpose of d_{ij} is to model the queuing delay directly proportional to the total size of transactions awaiting their turn/currently being forwarded (from i to j). When link speeds are faster, fewer messages are stuck in the queue and this number is smaller. When they are slower, more transactions start queuing up - increasing d_{ij}. Hence we model d_{ij} to be inversely proportional to the link speed c_{ij}.

(c) *...explanation for the choice of a particular mean. [ref. (7) in PS]*
We choose the inter-arrival time to be $H = 10$ min $= 600$ sec $= 600000$ ticks (a tick is the unit of time in our simulations), as this is the average inter-arrival time in Bitcoin. We also choose the average inter-arrival time for a transaction to be 60 sec for each node. 100 nodes generate this way, every 10 minutes, roughly 1000 transactions. This is good as it allows also to simulate the case of transactions not making it into blocks and trying to get into later blocks (999 transactions apart from the coinbase transaction can fit into a block).

2 Analysis of the blockchain tree

In this section, we analyze the blocktree (blockchain along with the orphaned branches) both from the perspective of an individual node and on aggregate. As there are a lot of experiment settings and corresponding results, we first present the **simulation settings** for each experiment, then present the observed **measurements**, and finally provide **commentary** on our observations.

2.1 Simulation settings

All scenarios we simulate were run for a simulated 10^5 seconds, *i.e.*, 10^8 ticks (a tick is the unit of time used by our simulator). As in the problem statement, n denotes the number of nodes, while z_0 and z_1 denote the percentage of nodes that are slow and low-CPU, respectively. In addition, T_{tx} and T_k are the mean transaction inter-arrival time (for each node) and the mean block inter-arrival time (globally). Further $T_{d_{ij}}$ is the numerator in the expression for the mean of d_{ij}, specified in the Problem Statement as 96 KB by default. In the simulations we run we take $z \equiv z_0 = z_1$ (although z_0 and z_1 may be set to different values in the file `config.py`). Our simulations include the runs shown in Table 1. Note that we take 1 KB = 1024 B (instead of 1000), etc. Further, 1 Kb = 1024 bits.

Name	n	z (%)	T_{tx} (s)	T_k (min)	$T_{d_{ij}}$
`base`	100	30.0	60	10	96 Kb
`n25`	**25**	30.0	60	10	96 Kb
`z60`	100	**60.0**	60	10	96 Kb
`tx30`	100	30.0	**30**	10	96 Kb
`blk3`	100	30.0	60	**3**	96 Kb
`d5`	100	30.0	60	10	**5 Gb**

Table 1: The settings for our simulations.

2.2 Measurements

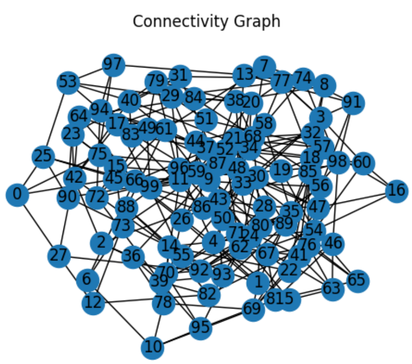
We list three main metrics for each run. First, P denotes the percentage of blocks mined by a node n that made it into its final view of the longest chain. R denotes the percentage of blocks in n's view of the longest chain that belong to it. These may not add up to 100% as the view of the blocktree may vary slightly between nodes. Finally, T is the average time elapsed between the mining of a block by another node and n hearing of it. The subscripts s and f denote slow and fast nodes respectively. Similarly, l and h denote low and high CPU nodes, respectively. All metrics are averaged across corresponding nodes in the system and are reported in the form of `mean` $\pm$ `std` (note that sometimes `std` may be larger than `mean` – this is not a contradiction). We report these metrics along with the time taken for simulation for 5 runs in each setting. In addition, we comment on the length and number of orphaned branches. In the included blocktree illustrations (of a randomly chosen node), blocks on the longest chain are colored green if they were mined by another node, and purple if they were mined by the selected one. Similarly, ones off the main chain are orange if they were mined by someone else, and red otherwise.

(a) **Metrics for** `base`
The metrics are shown in Table 2.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	2242.30 ± 0.00	2130.04 ± 0.00	2341.12 ± 0.00	2569.72 ± 0.00	2413.90 ± 0.00	2339.42 ± 149.57
P_{fh} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{sh} (%)	100.00 ± 0.00	99.05 ± 4.26	100.00 ± 0.00	100.00 ± 0.00	98.95 ± 4.47	99.60 ± 2.80
P_{sl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
R_{fh} (%)	1.40 ± 0.88	1.34 ± 0.84	1.44 ± 0.95	1.53 ± 0.76	1.33 ± 1.09	1.41 ± 0.91
R_{fl} (%)	0.14 ± 0.25	0.22 ± 0.40	0.22 ± 0.51	0.09 ± 0.21	0.07 ± 0.20	0.15 ± 0.34
R_{sh} (%)	1.28 ± 0.74	1.32 ± 0.89	1.12 ± 0.91	1.07 ± 0.53	1.47 ± 0.77	1.25 ± 0.79
R_{sl} (%)	0.13 ± 0.25	0.19 ± 0.43	0.07 ± 0.21	0.19 ± 0.27	0.20 ± 0.30	0.16 ± 0.30
T_{fh} (s)	1.35 ± 0.76	1.43 ± 0.75	1.33 ± 0.79	1.33 ± 0.77	1.44 ± 0.78	1.38 ± 0.77
T_{fl} (s)	1.32 ± 0.77	1.42 ± 0.76	1.25 ± 0.77	1.33 ± 0.76	1.43 ± 0.78	1.35 ± 0.77
T_{sh} (s)	2.68 ± 0.93	2.73 ± 0.93	2.70 ± 0.91	2.65 ± 0.95	2.70 ± 0.98	2.69 ± 0.94
T_{sl} (s)	2.71 ± 0.93	2.67 ± 0.89	2.71 ± 0.90	2.60 ± 0.95	2.70 ± 0.98	2.68 ± 0.93

Table 2: Metrics for the `base` setting.

An example connectivity graph (from Run 1) is shown in Figure 1a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 1b.



(a) Example connectivity graph for the `base` setting.

(b) Example final blockchain tree for the `base` setting.

Figure 1: The blocktree's image has also been included in the submission where it can be zoomed into.

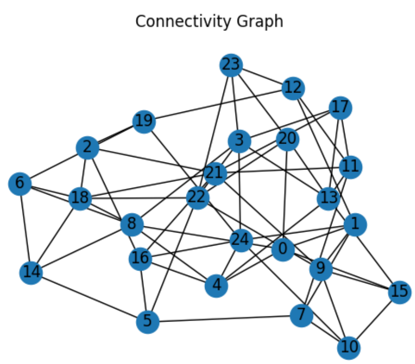
(b) **Metrics for** `n25`

The metrics are shown in Table 3.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	104.48 ± 0.00	112.88 ± 0.00	99.26 ± 0.00	100.82 ± 0.00	109.94 ± 0.00	105.48 ± 5.22
P_{fh} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{sh} (%)	98.46 ± 3.08	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	99.69 ± 1.51
P_{sl} (%)	100.00 ± 0.00	N/A	100.00 ± 0.00	100.00 ± 0.00	N/A	100.00 ± 0.00
R_{fh} (%)	5.18 ± 1.40	5.15 ± 1.35	5.74 ± 1.45	5.46 ± 1.40	5.44 ± 2.19	5.39 ± 1.61
R_{fl} (%)	0.38 ± 0.50	0.42 ± 0.46	0.45 ± 0.42	1.02 ± 0.56	0.70 ± 0.41	0.59 ± 0.53
R_{sh} (%)	5.66 ± 1.43	6.42 ± 1.15	4.29 ± 1.41	4.55 ± 1.72	4.99 ± 2.36	5.18 ± 1.84
R_{sl} (%)	0.94 ± 0.31	0.00 ± 0.00	0.56 ± 0.00	0.28 ± 0.28	0.00 ± 0.00	0.36 ± 0.41
T_{fh} (s)	0.71 ± 0.40	0.52 ± 0.31	0.64 ± 0.32	0.56 ± 0.35	0.59 ± 0.35	0.60 ± 0.35
T_{fl} (s)	0.61 ± 0.43	0.57 ± 0.30	0.59 ± 0.31	0.57 ± 0.35	0.59 ± 0.34	0.59 ± 0.35
T_{sh} (s)	1.02 ± 0.63	0.84 ± 0.50	0.93 ± 0.52	0.89 ± 0.50	0.88 ± 0.54	0.91 ± 0.54
T_{sl} (s)	0.94 ± 0.66	0.84 ± 0.50	0.84 ± 0.54	0.80 ± 0.48	0.82 ± 0.54	0.85 ± 0.55

Table 3: Metrics for the `n25` setting.

An example connectivity graph (from Run 1) is shown in Figure 2a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 2b.



(a) Example connectivity graph for the `n25` setting.

(b) Example final blockchain tree for the `n25` setting.

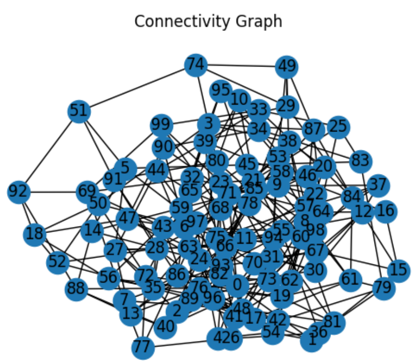
Figure 2: The blocktree's image has also been included in the submission where it can be zoomed into.

(c) **Metrics for** `z60`
The metrics are shown in Table 4.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	2179.30 ± 0.00	2404.00 ± 0.00	2294.54 ± 0.00	2159.54 ± 0.00	2138.10 ± 0.00	2235.10 ± 100.29
P_{fh} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	97.92 ± 6.91	99.58 ± 3.20
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	95.00 ± 15.00	100.00 ± 0.00	100.00 ± 0.00	99.00 ± 7.00
P_{sh} (%)	100.00 ± 0.00	98.61 ± 6.66	98.15 ± 6.55	100.00 ± 0.00	100.00 ± 0.00	99.35 ± 4.25
P_{sl} (%)	91.67 ± 27.64	100.00 ± 0.00	100.00 ± 0.00	92.86 ± 25.75	100.00 ± 0.00	96.90 ± 17.32
R_{fh} (%)	2.16 ± 1.03	1.99 ± 1.08	1.44 ± 1.21	2.11 ± 0.91	1.84 ± 0.46	1.91 ± 1.01
R_{fl} (%)	0.25 ± 0.35	0.26 ± 0.46	0.24 ± 0.36	0.20 ± 0.38	0.31 ± 0.42	0.25 ± 0.40
R_{sh} (%)	2.11 ± 1.02	2.25 ± 1.11	2.51 ± 1.19	2.12 ± 1.17	2.18 ± 1.21	2.23 ± 1.15
R_{sl} (%)	0.23 ± 0.38	0.21 ± 0.33	0.16 ± 0.24	0.28 ± 0.35	0.24 ± 0.38	0.22 ± 0.34
T_{fh} (s)	2.21 ± 1.08	2.74 ± 1.38	2.77 ± 1.40	3.09 ± 1.46	3.17 ± 1.65	2.80 ± 1.44
T_{fl} (s)	2.27 ± 1.09	2.92 ± 1.54	2.63 ± 1.29	2.88 ± 1.41	3.07 ± 1.48	2.75 ± 1.40
T_{sh} (s)	3.25 ± 1.28	3.68 ± 1.46	3.64 ± 1.51	3.95 ± 1.34	3.85 ± 1.45	3.67 ± 1.43
T_{sl} (s)	3.33 ± 1.36	3.74 ± 1.45	3.54 ± 1.47	3.99 ± 1.38	3.85 ± 1.44	3.69 ± 1.44

Table 4: Metrics for the `z60` setting.

An example connectivity graph (from Run 1) is shown in Figure 3a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 3b.



(a) Example connectivity graph for the `z60` setting.

(b) Example final blockchain tree for the `z60` setting.

Figure 3: The blocktree's image has also been included in the submission where it can be zoomed into.

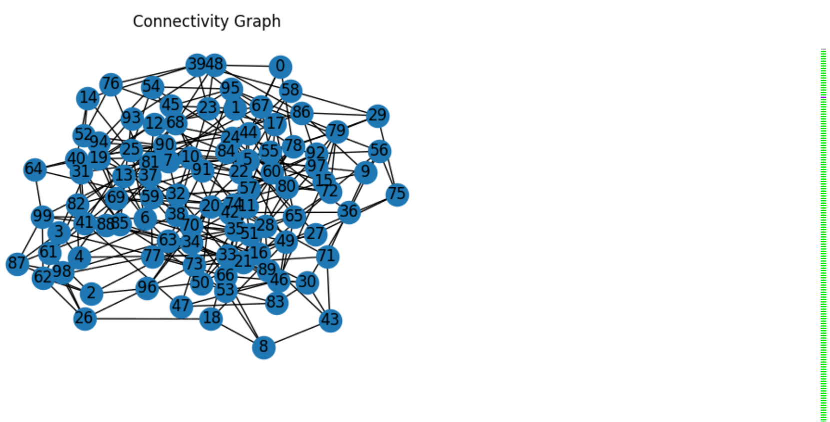
(d) **Metrics for** `tx30`

The metrics are shown in Table 5.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	5289.46 ± 0.00	5499.05 ± 0.00	5417.49 ± 0.00	5446.80 ± 0.00	6337.37 ± 0.00	5598.03 ± 376.06
P_{fh} (%)	100.00 ± 0.00	100.00 ± 0.00	98.89 ± 7.37	100.00 ± 0.00	100.00 ± 0.00	99.78 ± 3.33
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{sh} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	98.17 ± 5.53	100.00 ± 0.00	99.63 ± 2.58
P_{sl} (%)	100.00 ± 0.00	N/A	100.00 ± 0.00	N/A	100.00 ± 0.00	100.00 ± 0.00
R_{fh} (%)	1.39 ± 0.98	1.38 ± 0.92	1.36 ± 0.99	1.39 ± 0.90	1.26 ± 0.82	1.36 ± 0.92
R_{fl} (%)	0.12 ± 0.25	0.15 ± 0.27	0.07 ± 0.19	0.15 ± 0.26	0.18 ± 0.32	0.13 ± 0.26
R_{sh} (%)	1.33 ± 0.66	1.39 ± 0.94	1.36 ± 0.88	1.35 ± 0.73	1.56 ± 1.15	1.40 ± 0.89
R_{sl} (%)	0.08 ± 0.21	0.00 ± 0.00	0.26 ± 0.30	0.00 ± 0.00	0.07 ± 0.19	0.08 ± 0.21
T_{fh} (s)	1.45 ± 0.80	1.31 ± 0.74	1.30 ± 0.77	1.48 ± 0.90	1.65 ± 0.99	1.44 ± 0.85
T_{fl} (s)	1.40 ± 0.81	1.29 ± 0.74	1.38 ± 0.77	1.41 ± 0.81	1.65 ± 0.92	1.43 ± 0.82
T_{sh} (s)	2.87 ± 0.86	2.67 ± 0.80	2.76 ± 0.82	2.95 ± 0.86	3.02 ± 0.99	2.85 ± 0.88
T_{sl} (s)	2.92 ± 0.86	2.74 ± 0.82	2.80 ± 0.83	2.89 ± 0.87	2.97 ± 0.96	2.86 ± 0.87

Table 5: Metrics for the `tx30` setting.

An example connectivity graph (from Run 1) is shown in Figure 4a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 4b.



(a) Example connectivity graph for the `tx30` setting. (b) Example final blockchain tree for the `tx30` setting.

Figure 4: The blocktree's image has also been included in the submission where it can be zoomed into.

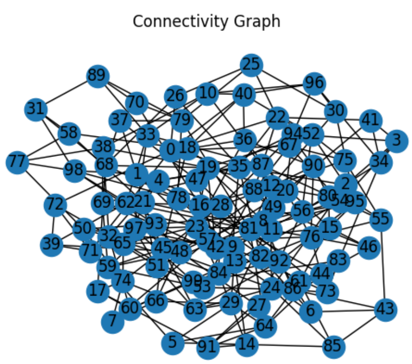
(e) **Metrics for `blk3`**

The metrics are shown in Table 6.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	4268.86 ± 0.00	4727.06 ± 0.00	4004.41 ± 0.00	4280.77 ± 0.00	4096.45 ± 0.00	4275.51 ± 248.81
P_{fh} (%)	99.50 ± 3.50	99.57 ± 2.89	100.00 ± 0.00	99.47 ± 2.63	100.00 ± 0.00	99.71 ± 2.36
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
P_{sh} (%)	99.50 ± 2.18	97.78 ± 8.37	100.00 ± 0.00	100.00 ± 0.00	98.64 ± 4.55	99.18 ± 4.45
P_{sl} (%)	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00
R_{fh} (%)	1.31 ± 0.46	1.34 ± 0.42	1.33 ± 0.47	1.39 ± 0.48	1.36 ± 0.44	1.35 ± 0.46
R_{fl} (%)	0.14 ± 0.20	0.14 ± 0.12	0.11 ± 0.15	0.08 ± 0.12	0.15 ± 0.15	0.12 ± 0.15
R_{sh} (%)	1.54 ± 0.50	1.43 ± 0.42	1.47 ± 0.50	1.36 ± 0.44	1.38 ± 0.45	1.44 ± 0.47
R_{sl} (%)	0.05 ± 0.08	0.13 ± 0.12	0.14 ± 0.27	0.14 ± 0.18	0.14 ± 0.13	0.12 ± 0.17
T_{fh} (s)	0.99 ± 0.47	0.89 ± 0.53	0.91 ± 0.48	0.86 ± 0.45	0.88 ± 0.44	0.90 ± 0.48
T_{fl} (s)	0.99 ± 0.48	0.87 ± 0.53	0.89 ± 0.47	0.85 ± 0.44	0.85 ± 0.42	0.89 ± 0.47
T_{sh} (s)	1.34 ± 0.73	1.35 ± 0.81	1.38 ± 0.77	1.29 ± 0.73	1.28 ± 0.68	1.33 ± 0.75
T_{sl} (s)	1.43 ± 0.78	1.36 ± 0.82	1.43 ± 0.78	1.26 ± 0.72	1.23 ± 0.69	1.34 ± 0.76

Table 6: Metrics for the `blk3` setting.

An example connectivity graph (from Run 1) is shown in Figure 5a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 5b.



(a) Example connectivity graph for the `blk3` setting.

(b) Example final blockchain tree for the `blk3` setting.

Figure 5: The blocktree's image has also been included in the submission where it can be zoomed into.

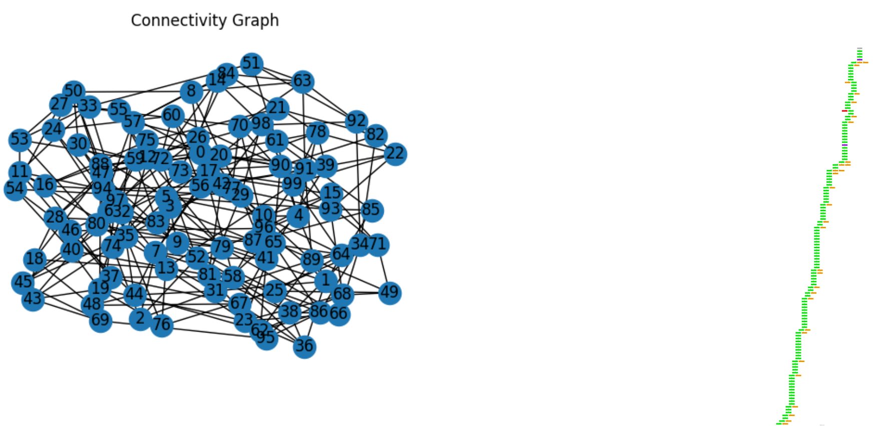
(f) **Metrics for** `d5`

The metrics are shown in Table 7.

Metric	Run 1	Run 2	Run 3	Run 4	Run 5	Aggregate
Time(s)	3304.99 ± 0.00	3202.52 ± 0.00	3415.09 ± 0.00	3415.09 ± 0.00	3296.41 ± 0.00	3058.32 ± 124.17
P_{fh} (%)	93.14 ± 14.24	88.55 ± 24.63	82.54 ± 29.17	86.70 ± 25.64	91.32 ± 17.26	88.45 ± 23.17
P_{fl} (%)	100.00 ± 0.00	100.00 ± 0.00	93.75 ± 16.54	100.00 ± 0.00	83.33 ± 23.57	95.42 ± 14.43
P_{sh} (%)	50.42 ± 30.79	43.65 ± 34.71	55.64 ± 34.83	43.77 ± 39.91	44.98 ± 33.93	47.69 ± 35.27
P_{sl} (%)	100.00 ± 0.00	0.00 ± 0.00	50.00 ± 50.00	66.67 ± 0.00	0.00 ± 0.00	43.33 ± 44.85
R_{fh} (%)	1.70 ± 0.99	1.55 ± 1.27	1.40 ± 1.06	1.70 ± 1.07	1.60 ± 0.94	1.59 ± 1.08
R_{fl} (%)	0.20 ± 0.33	0.28 ± 0.38	0.28 ± 0.35	0.14 ± 0.30	0.11 ± 0.28	0.20 ± 0.34
R_{sh} (%)	0.65 ± 0.46	0.80 ± 0.95	1.15 ± 0.94	0.60 ± 0.62	0.94 ± 0.75	0.83 ± 0.79
R_{sl} (%)	0.11 ± 0.26	0.00 ± 0.00	0.08 ± 0.23	0.19 ± 0.51	0.00 ± 0.00	0.08 ± 0.29
T_{fh} (s)	163.63 ± 182.57	174.25 ± 189.53	166.10 ± 167.49	167.92 ± 190.30	190.24 ± 199.47	172.43 ± 186.42
T_{fl} (s)	161.42 ± 181.94	174.74 ± 189.56	163.54 ± 167.81	158.37 ± 178.86	191.76 ± 198.45	169.97 ± 184.02
T_{sh} (s)	364.65 ± 264.74	373.36 ± 268.40	384.42 ± 262.68	389.44 ± 285.74	419.11 ± 288.22	386.20 ± 274.80
T_{sl} (s)	377.49 ± 265.57	391.50 ± 294.26	384.58 ± 261.13	374.30 ± 281.97	415.48 ± 295.78	388.67 ± 280.49

Table 7: Metrics for the `d5` setting.

An example connectivity graph (from Run 1) is shown in Figure 6a. The produced blockchain tree is illustrated from a bird's-eye-view in Figure 6b.



(a) Example connectivity graph for the `d5` setting.

(b) Example final blockchain tree for the `d5` setting.

Figure 6: The blocktree's image has also been included in the submission where it can be zoomed into.

2.3 Commentary on Observations

We make the following comments on our observations:

(a) **In all settings apart from** `d5`**, branches are extremely uncommon**. Note that[1] regular random graphs have a very low diameter (as low as 2 in the common case for our 100-node graphs). Therefore the time for a node to hear of the block mined by another node is of the order of seconds, while the mining time

[1] `https://mathoverflow.net/questions/365340/the-diameter-of-random-regular-graphs`

across nodes is on average 10 minutes, and so most nodes hear about other miners' blocks in time to avoid branches. However, in the d5 setting, the delay has been artificially ramped up, leading to rampant branching.

(b) **In all settings except d5, all types of nodes almost always manage to get their blocks into the main chain.** This follows directly from the previous bullet point. Even in the d5 setting, it is only the slow high-CPU nodes that see a major drop. This is because they both produce enough blocks to be statistically meaningful *and* suffer from low propagation times. Note that if a node is fast, it will probably hear messages fast through the "fast-only" paths connecting it to other fast nodes, but a minimum of one slow edge is unavoidable for slow nodes.

(c) **The number of blocks a node slips into the main chain depends on it being high or low CPU, but is not affected much by it being fast or slow**. As in the various experiments, we see that a node being fast or slow does not impact its contribution to the main chain except in d5. This is well explained by the first bullet point. In addition, although slow nodes may hear of blocks later potentially resetting more progress, they may also mine a block themselves in that small extra interval leading to them continuing their own chain. Therefore slow connections do not impact the proportion of one's own block to others' in the final chain.

(d) **Most branches are just a single block long**. As it is, branches are rare. The probability of a branch growing longer than one block before getting orphaned is vanishingly small, and therefore seldom happens.

(e) **Most simulation parameters do not affect the recall or precision of one's blocks in the final longest chain.** The results are very similar across experiments in this regard. The numbers are larger in n25, but that is because there are fewer nodes to contribute to the total against which the percentage is calculated. We also observe that in z60, the results improve slightly more than their counterpart. This is because in base, a slow node has $\frac{100.0\%\times 1}{30+70\times 10} \approx 0.137\%$ of the computation power and a fast node 1.37%. In the z60 setting the corresponding percentages are $\frac{100.0\%}{60+40\times 10} = 0.217\%$ and 2.17%, i.e. both slow and fast nodes have a greater share of the total computation power. This is possible because there are fewer high-CPU nodes, leading to a lower total. Subsequently, the numbers are a bit better in the z60 runs.

(f) **The time taken for a node to hear of another's block depends only on it being fast/slow, and is roughly double for the slower nodes**.