Blockchain Arena: Simulation of a P2P Cryptocurrency Network

Project 1 Report

Tanay Jha
Roll No. 24B1040

June 25, 2025

Contents

1 Introduction

This report details the design, implementation, and analysis of a discrete-event simulator for a Proof-of-Work (PoW) based P2P cryptocurrency network. The simulator, built in Python, models key aspects of a blockchain system, including network topology, peer heterogeneity (in terms of network speed and CPU power), transaction propagation, PoW mining, and fork resolution.

The primary objective of this project is to use the simulator as a digital laboratory to study the emergent properties of a decentralized network under various conditions. Through a series of controlled experiments, we investigate the impact of network parameters on system stability, economic efficiency, and the distribution of wealth and power among miners.

2 Theoretical Justifications

T his section provides theoretical justification for certain modeling choices made in the simulation.

2.1 Choice of Exponential Distribution for Inter-arrival Times

The project requires that both transaction generation times (mean T_{tx}) and block mining times (mean I/h_k) follow an exponential distribution. This choice is theoretically sound for several reasons:

- **Memoryless Property:** The exponential distribution is the only continuous probability distribution that is "memoryless." This means that the probability of an event occurring in the future is independent of how much time has already passed. In the context of PoW mining, this is a perfect model. A miner's chance of finding a block in the next second is the same regardless of whether they have been mining for one second or one hour. Each hash attempt is an independent trial.
- **Poisson Process:** Events that occur independently and at a constant average rate are well-described by a Poisson process. The time between successive events in a Poisson process is exponentially distributed. If we assume that users across the network generate transactions independently, then the stream of transactions from any single peer can be modeled as a Poisson process, making the exponential distribution the correct choice for inter-arrival times.

2.2 Relationship Between Queuing Delay and Link Speed

The simulation models the queuing delay (d_{ij}) at a node i with a mean of 96 kbits$/c_{ij}$, where c_{ij} is the link speed. This inverse relationship is fundamental to queuing theory.

- **Intuitive Explanation:** A queue forms at a router (or peer) when data arrives faster than it can be sent out. The link speed, c_{ij}, determines the rate at which data can be "serviced" or sent out. A higher link speed (a larger c_{ij}) means the node can clear its outgoing message queue much faster. Consequently, messages spend less time waiting in the queue, leading to a lower average queuing delay.
- **Queuing Models:** This model is a simplification of more complex queuing systems like M/M/1 queues, but it correctly captures the core principle: service rate (analogous to c_{ij}) is inversely proportional to waiting time. A faster link provides a higher service rate, thus reducing delays.

3 Experimental Analysis and Findings

This section presents the results from a series of simulation runs designed to explore the properties of the blockchain network.

3.1 The Impact of Block Inter-arrival Time (I)

The inter-arrival time, I, is a critical parameter for network stability.

3.1.1 Scenario 1: Low Inter-arrival Time (Unstable Network)

With a low I (e.g., $I = 8$), the network becomes highly unstable. The block generation time is too short relative to the time it takes for blocks to propagate across the network, especially with a high fraction of slow peers.

As seen in Figure 1, this leads to a "bushy" tree with many long, competing forks. The Miner Efficiency chart (Figure 2) confirms this, showing a very high proportion of stale blocks for all miners, indicating a huge amount of wasted work.

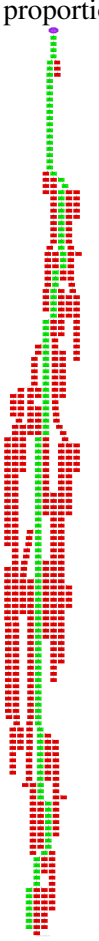

Figure 1: Blockchain tree with low I, showing extreme forking and instability.

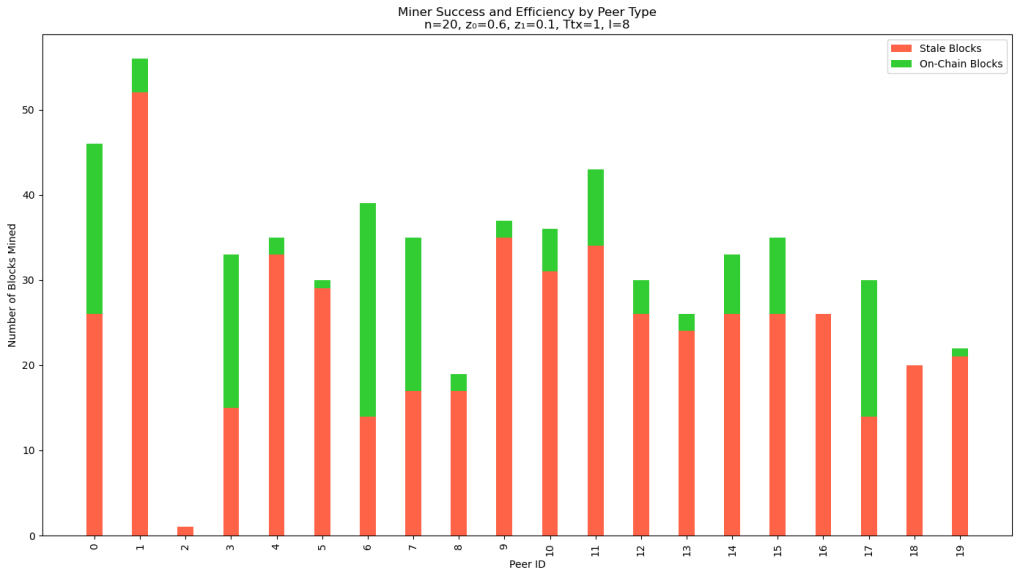

Figure 2: Miner efficiency with low I. The high ratio of stale (red) to on-chain (green) blocks shows massive inefficiency.

3.1.2 Scenario 2: High Inter-arrival Time (Stable Network)

Conversely, a high I (e.g., $I = 50$ or $I = 100$) creates a stable environment. This gives new blocks ample time to propagate, allowing the network to reach consensus before the next block is likely found. This leads to a highly linear blockchain (Figure 3) and greater mining efficiency, as visible in Figure 4.

Figure 3: Blockchain tree with high I, showing a near-linear chain and strong consensus.

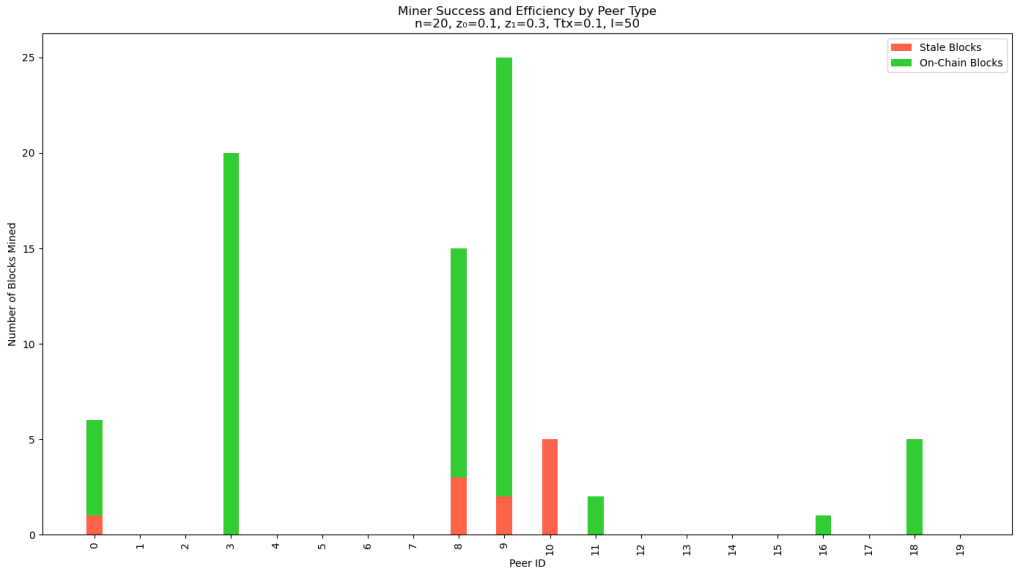

Figure 4: Miner efficiency with high I. The high ratio of on-chain (green) to stale (red) blocks shows good efficiency.

3.2 The Role of Hashing Power and Network Speed

Our experiments reveal a clear hierarchy of factors determining a miner's success.

- **Hashing Power is Primary:** Consistently, peers with 'high_power' CPUs accumulate the vast majority of the wealth. In scenarios with little competition (e.g., only 2 high-power peers), this leads to a near-total monopoly, as one miner establishes a positive feedback loop and wins nearly every block (Figure 5).
- **Network Speed is Secondary but Significant:** Network speed acts as a competitive advantage. However, its impact is nuanced. A 'slow_net' peer with a favorable, central position in the network topology can outperform a 'fast_net' peer on the network's periphery. This shows that total latency, not just link speed, is what matters. Figure 6 shows a scenario where both slow and fast net peers were among the top earners.

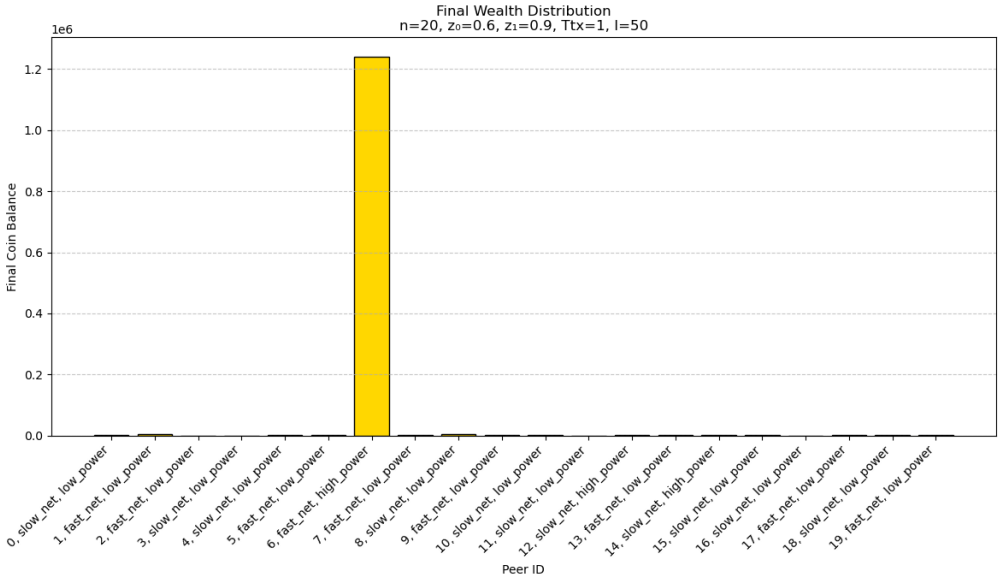

Figure 5: Wealth distribution in a non-competitive market, showing a monopoly by one high-power peer.

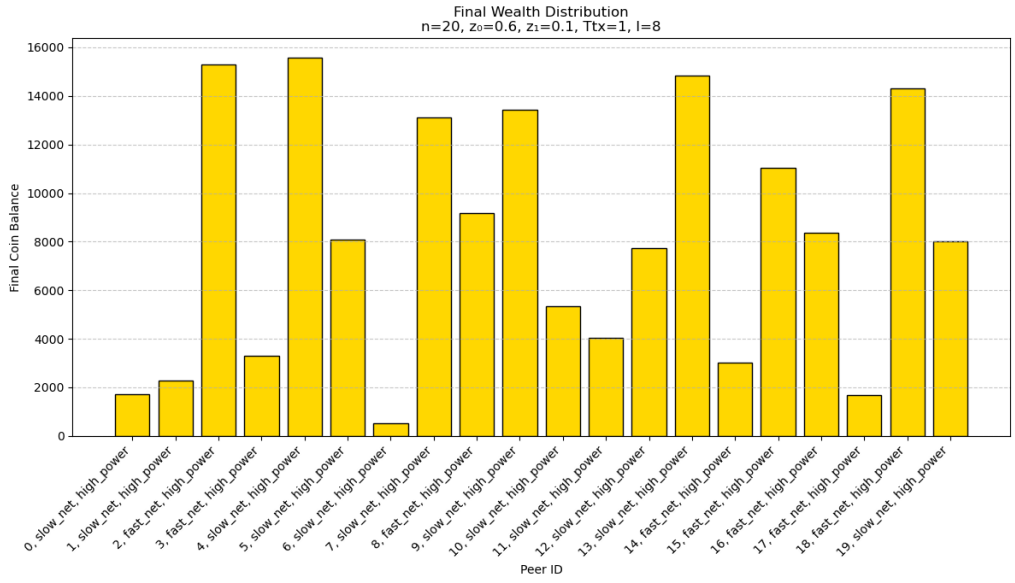

Figure 6: Wealth distribution in a competitive market. Top earners include both 'fast_net' and 'slow_net' peers.

3.3 Simulator Performance and Scalability

A key observation during this project was the performance of the simulator itself.

- The runtime per iteration slows down drastically as the simulation progresses. This is due to the naive 'calculate_balance' algorithm, which has a complexity of $O(L)$, where L is the length of the blockchain. This means that every time a new block is to be verified for it's transactions, the simulator goes through the complete blockchain for calculating balances.
- To manage this, a state-caching mechanism (e.g., storing a balance snapshot for each block) would be necessary for larger-scale simulations. This represents a classic time-memory trade-off. Since the memory on my computer is quite limited at 8GB, I went for a longer runtime but lower memory needs algorithm.
- Due to these performance considerations, the number of peers was limited to 20-40 to allow for experiments to complete in a reasonable timeframe. This is lower than the 50-100 mentioned in the previous network building assignment.

4 Conclusion

This project successfully developed a robust simulator for a P2P cryptocurrency network. The key takeaways from our experiments are:

1. Network stability is a direct function of the ratio between block time (I) and propagation delay. Low ratios lead to instability and inefficiency.
2. Hashing power is the primary determinant of success, but in competitive environments, secondary factors like network topology and speed can be decisive.
3. The positive feedback loop, where a winning miner gets a head-start on the next block, is a powerful centralizing force, especially in stable networks with low competition.
4. Designing a scalable blockchain simulator requires careful consideration of algorithmic complexity, particularly for state validation functions.

The simulator has proven to be an effective tool for exploring these complex, emergent properties and provides a solid foundation for future experiments, such as modeling network attacks or transaction fee markets.

Note regarding generative AI: I only used genAI tools for a visualisation function since I do not have much knowledge of graphviz. I tried using matplotlib for the same tasks but the properties of the blockchain weren't immediately obvious from the resulting graph, compared to the current red-green chain method which is very clear.