Blockchain Arena: Simulating Mining Wars and Network Attacks

Project 1 Report - Simulation of a P2P Cryptocurrency Network

June 2025

Objective

The goal of this project is to develop a discrete-event simulator that models a peer-to-peer (P2P) cryptocurrency network. The simulator implements network formation, transaction propagation, block creation and propagation, and consensus through longest chain protocol.

Design Components

1. Peer-to-Peer Network (Network Class)

- Nodes are classified as `fast` or `slow`, and as `high CPU` or `low CPU` based on parameters z_0 and z_1.
- Connections per peer are randomly selected in [3, 6].
- Network is undirected and connected.
- Each node is an instance of the `Peer` class.

2. Transaction Model (Transaction Class)

- Format: "TxnID: IDx pays IDy C coins".
- Size fixed at 1KB.
- Generated randomly using exponential distribution (mean T_{tx}).
- **Why exponential?** Because it models memoryless inter-arrival time, mimicking Poisson arrivals commonly assumed in decentralized systems.

3. Latency Model

- Latency $L_{ij} = \rho_{ij} + \frac{|m|}{c_{ij}} + d_{ij}$.
- ρ_{ij} sampled from $\mathcal{U}(10ms, 500ms)$.
- $|m|$ is message size in bits.
- c_{ij} is 100 Mbps if both peers are fast, else 5 Mbps.

- $d_{ij} \sim$ Exponential($\frac{c_{ij}}{96k}$).
- **Why inversely proportional to c_{ij}?** Because faster links reduce queuing times — typical in queuing theory.

4. Loop-less Forwarding

- Transactions are forwarded only to peers from which they were not received and not already sent to.
- This avoids infinite propagation loops.

5. Blockchain and PoW Simulation

- Each peer maintains a `BlockchainTree` of received blocks.
- Genesis block is inserted at $t = 0$.
- Blocks include valid transactions only and add coinbase transaction for miner (+50 coins).
- Block is mined after $T_k \sim$ Exponential(I/h_k) where h_k is peer's hash power.
- **Why exponential T_k?** Poisson process modeling random block discovery.

6. Discrete-Event Simulation

- Central event queue handled by `Simulator` class.
- Event types: `Gen_txn`, `Rec_txn`, `Rec_block`, `Gen_block`.
- Event ordering ensures causal simulation.

Files and Submission

- Code: `blockchain_simulator.py`
- README: Setup and execution instructions.
- Report: This LaTeX document (`report.tex`)