CoralChain

Author: **Xinjian Zhang** (MSc Student)
Supervisor: **Mubashar Iqbal, PhD**

CoralChain – A Visual Gadget Collection for Exploring Blockchain Fundamentals

INTRODUCTION

CoralChain is a lightweight, web-based blockchain simulation framework built with Ruby and Sinatra. It brings together a collection of visual, interactive gadgets designed to illustrate key concepts such as blockchain structure, multi-node synchronization, and consensus mechanism comparison. This project offers an approachable and hands-on way to explore the fundamental behaviors of blockchain systems in simplified, illustrative scenarios.

MAIN TECH

Ruby + Sinatra + Chart.js

METHODS

Blockchain Structure

- Each block contains: index, timestamp, data, previous_hash, nonce, and SHA-256 hash
- hash is computed as: SHA256(index + timestamp + data + previous_hash + nonce)
- Chain validity requires contiguous hash linkage and deterministic rehash verification
- Block acceptance depends on consensus-specific hash constraints (e.g., prefix match)

Multi-Node Architecture

- Each node runs an isolated chain instance with independent state
- Inter-node synchronization is resolved via Longest Valid Chain policy
- Chain divergence is detected through indexed block comparison
- Byzantine node introduces tampered chains to simulate adversarial propagation and fork injection

Consensus Mechanism Simulation

- **Proof of Work (PoW)**: Iteratively increment nonce to find a hash matching N-bit zero prefix (e.g., "00000...")
- **Proof of Authority (PoA)**: Emulates validator role via fixed-round randomized hash computations
- **Proof of Stake (PoS)**: Generates stake-weighted nonce values to influence hash output over multiple iterations

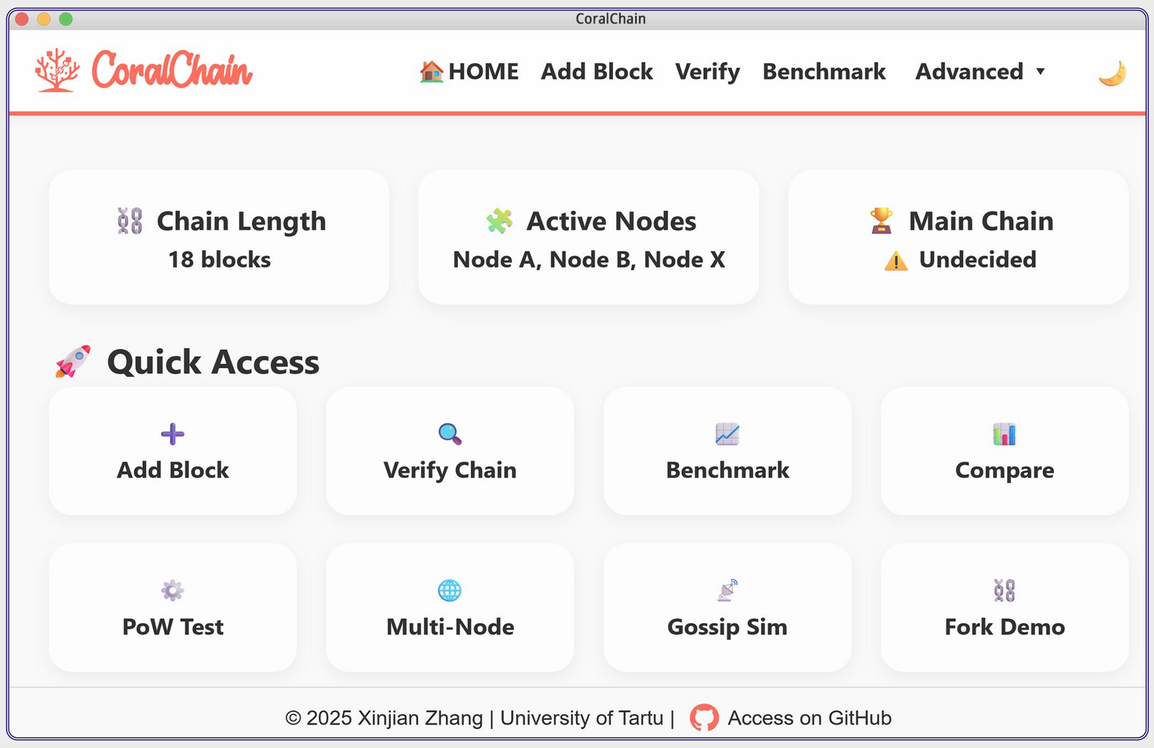

Fig 1. Main Interface of CoralChain

KEY FEATURES

1 Block Operations & Verification

- Create blocks with input data and chosen consensus method
- Hash and verify chains using SHA-256 linkage
- Tamper blocks to trigger chain-wide invalidation
- Show mismatched hashes and real-time verification results

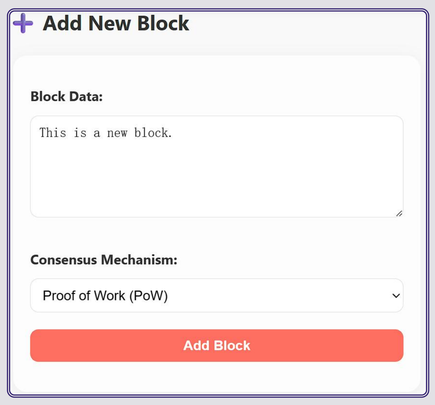

Fig 2. Block Creation

Fig 3. Blockchain Integrity Check

2 Multi-Node Interaction & Synchronization

Fig 4. Chain Synchronization Between Nodes A & B

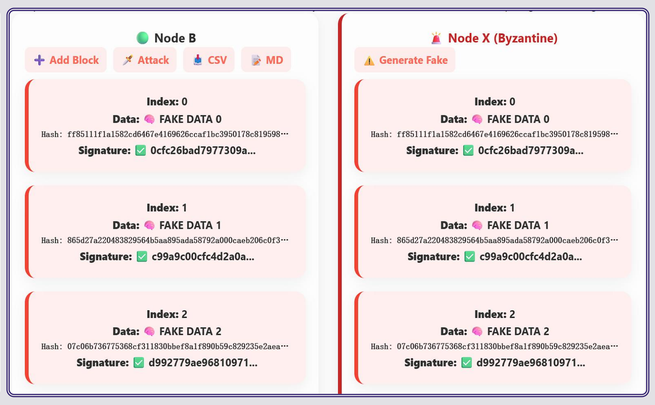

Fig 5. Byzantine Fake Chain Injection (Node X -> B)

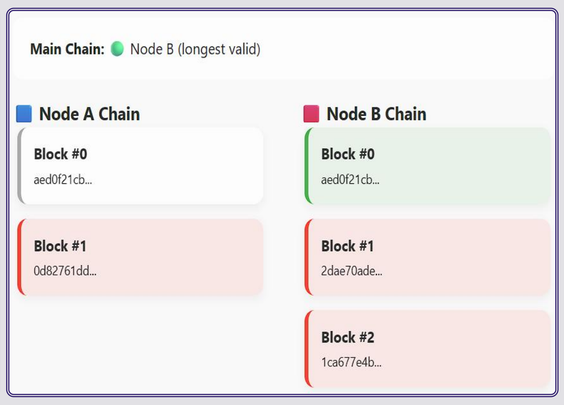

Fig 6. Chain Fork Between Nodes A & B

3 Consensus & Network Simulation

- Switch between PoW, PoS, and PoA during runtime
- Simulate gossip-based message spread with delay and faults
- Track block time and performance across algorithms
- Export chain data and results as CSV

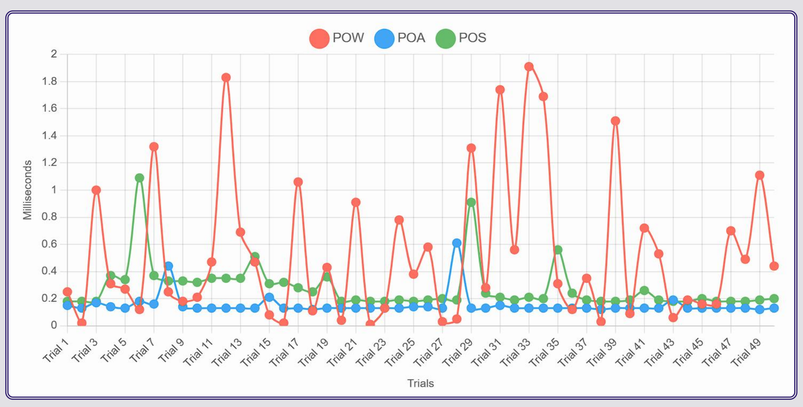

Fig 7. Consensus Benchmark Results (PoW, PoA, PoS, 50 Trials Each)

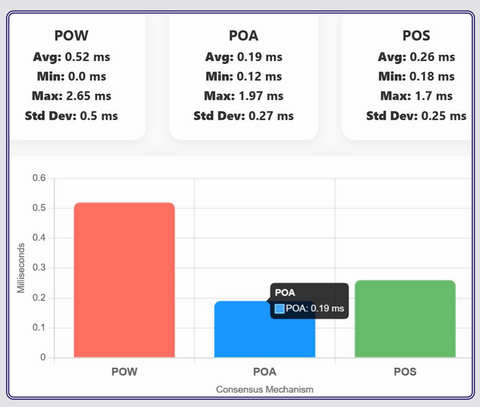

Fig 8. Consensus Performance Comp. (50 Trials Each)

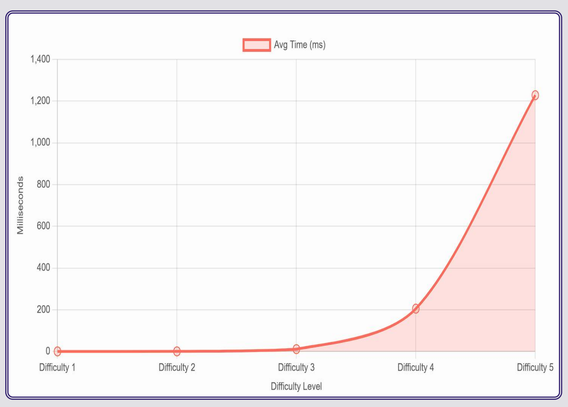

Fig 9. PoW Difficulty Experiment (5 Blocks, Levels 1–5)

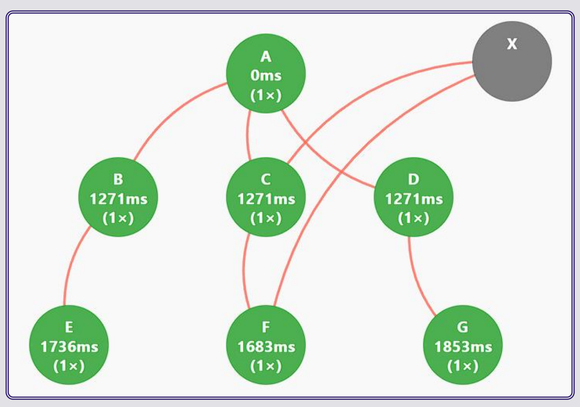

Fig 10. Gossip Propagation Sim. (RandDelay, 7 Nodes)

INSIGHTS

This project originated from an accidental spontaneous idea, without any specific goal in mind.
While its current functionality is limited, I hope it might offer a small spark of interest for anyone curious about blockchain.

SCAN HERE

Website

GitHub Repo

UNIVERSITY OF TARTU
Institute of Computer Science
1632

CHESS
Cyber-security Excellence Hub in Estonia and South Moravia
https://chess-eu.cs.ut.ee/

Visit the Website for More Details
Available in English and Estonian