

BLUEPRINT PROYEK: rnr-core (Proof of Sequential Sorting Race)

Proyek ini adalah blockchain berkinerja tinggi yang mengganti konsensus brute-force (PoW) dengan kompetisi algoritma pengurutan data. Sistem ini dirancang untuk mencapai skalabilitas global dengan tetap memperhatikan keterbatasan perangkat keras.

1. Arsitektur Konsensus: PoSSR (Proof of Sequential Sorting Race)

Dalam sistem ini, "Mining" adalah balapan (race) untuk mengurutkan 100 MB data transaksi. Seperti pada visualisasi video yang Anda sertakan, efisiensi setiap algoritma berbeda-beda (Quick Sort, Merge Sort, Radix, dsb).

A. Lotere Algoritma (The Variable Challenge)

Setiap blok (1 menit sekali), jaringan melalui VRF (Verifiable Random Function) menentukan Lotre Algoritma.

- * Contoh: Blok #101 mewajibkan penggunaan Heap Sort. Blok #102 mewajibkan Shell Sort.
- * Tujuan: Mencegah optimasi satu jalur pada hardware dan memaksa node memiliki implementasi kode yang fleksibel dan efisien.

B. Spesifikasi Balapan (The Race)

- * Workload per Node: 100 MB data mentah (Mempool Shard).
- * Target Pemenang: 10 Node tercepat secara global (Top 10 Finishers).
- * Block Composition: 10 Pemenang \times 100 MB = 1 GB per Blok.

2. Kalkulasi Performa & Skalabilitas

Parameter Nilai / Kapasitas
--- ---
Waktu Blok 60 Detik (1 Menit)
Kapasitas Blok 1 GB (1.024 MB)
Workload per Node 100 MB
Estimasi Transaksi/Blok ~2.147.483 Tx (asumsi 500 byte/tx)
Throughput (TPS) 35.791 TPS
Siklus Pruning Setiap 25 Blok (approx 25 Menit)
Data Aktif (Live) 25 GB (untuk 25 blok terakhir)

3. Logika Pemrosesan Data (Algoritma)

Node harus melakukan tiga tahap komputasi dalam satu menit:

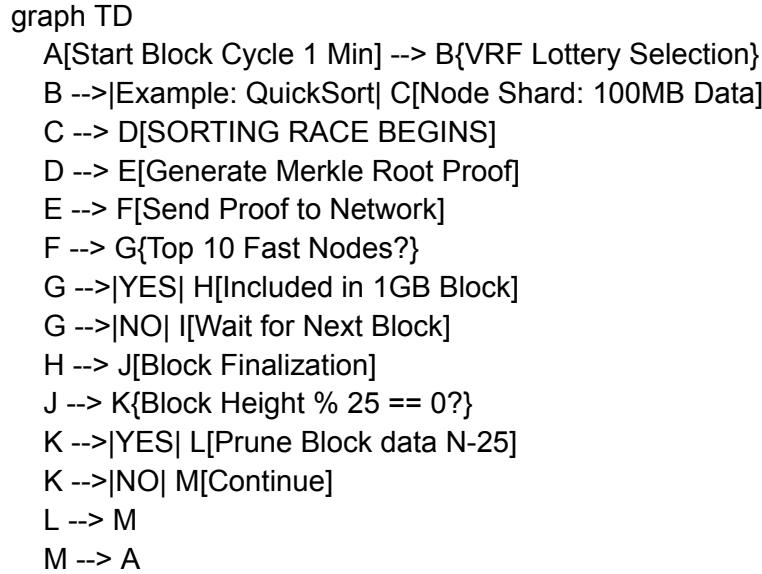
- * Ingestion: Mengambil 100 MB transaksi dari mempool yang sesuai dengan shard-nya.
 - * Sorting Race: Mengurutkan transaksi berdasarkan kunci unik (Hash Tx + Seed).
 - * Sifat: CPU-Intensive & Memory-Latency Bound.
- * Merkle Proof: Menghasilkan Merkle Root dari hasil sorting sebagai bukti kerja yang akan dikirim ke jaringan.

4. Sistem Pruning (Manajemen Penyimpanan)

Untuk menjaga agar perangkat (terutama Android atau VPS murah) tidak kehabisan storage:

- * Pruning Window: 25 Blok.
- * Metode: Setiap kali blok ke-N divalidasi, data transaksi mentah pada blok ke-(N-25) dihapus dari disk.
- * Integritas: Hanya State Root (saldo akhir) dan Header yang disimpan secara permanen. Ini memastikan sejarah tetap aman secara kriptografi tanpa beban penyimpanan yang membengkak (Storage-efficient).

5. Diagram Alur Sistem (Visual Blueprint)



6. Struktur Kode Dasar (Golang)

Berikut adalah struktur data utama untuk implementasi rnr-core:

```
package main
```

```
import (
    "crypto/sha256"
    "sort"
    "time"
)

// Shard 100MB untuk setiap node
type MempoolShard struct {
    Transactions []Transaction // Total ~200k transaksi
    Algorithm    string        // Hasil lotre (QuickSort, HeapSort, etc)
}

// Bukti Kerja yang dikirim ke jaringan
type SortingProof struct {
    NodeID      string
    MerkleRoot  string
    Duration    time.Duration
    Timestamp   int64
}

// Mining Logic
func PerformMining(shard MempoolShard) SortingProof {
    start := time.Now()
```

```

// Eksekusi Sorting berdasarkan lotre
// Sesuai visualisasi: https://youtu.be/kPRA0W1kECg
sort.Slice(shard.Transactions, func(i, j int) bool {
    return shard.Transactions[i].Hash < shard.Transactions[j].Hash
})

duration := time.Since(start)
root := GenerateMerkle(shard.Transactions)

return SortingProof{
    MerkleRoot: root,
    Duration: duration,
    Timestamp: time.Now().Unix(),
}
}

```

7. Keunggulan Strategis rnr-core

- * Keamanan (Zcash & BTC Style): Menggunakan hashing berlapis dan bukti matematika yang sulit dipalsukan tetapi cepat diverifikasi.
 - * Kecepatan (Solana Style): Menggunakan arsitektur paralel dan optimasi memori untuk mencapai puluhan ribu TPS.
 - * Adil & Terdistribusi: Siapa pun dengan kode algoritma yang paling efisien bisa menang, tidak hanya mereka yang punya modal besar untuk listrik.
- Blueprint ini menjadi fondasi bagi rnr-core untuk menjadi blockchain masa depan yang menggabungkan kecerdasan algoritma dengan efisiensi sumber daya.