Prism: Scaling Bitcoin by 10,000×

Lei Yang [1] Vivek Bagaria [2] Gerui Wang [3] Mohammad Alizadeh [1]
David Tse [2] Giulia Fanti [4] Pramod Viswanath [3]

[1]MIT CSAIL

[2]Stanford University

[3]University of Illinois Urbana-Champaign

[4]Carnegie Mellon University

The Stanford Blockchain Conference 2020

Bitcoin: high security, low throughput, and long latency

- Security: 50% adversary
- Throughput: 7 tps
- Confirmation Latency: hours

Prism: decoupling performance from security

- Bitcoin: throughput, latency coupled with security
- Prism: decouple performance from security

This talk

The Prism consensus protocol

System implementation

Evaluation results and findings

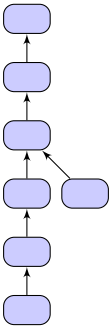
This talk

The Prism consensus protocol

System implementation

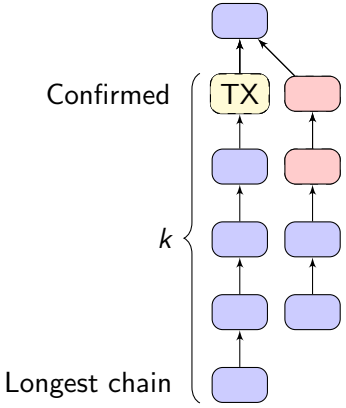
Evaluation results and findings

Bitcoin: the longest chain rule



Mining rate = 1 block per 10 min

Bitcoin: k-deep confirmation rule



30% adversary power

For 10^{-3} attack probability, wait 250 mins!

Nakamoto's Table

k	ϵ
0	1.0000000
5	0.1773523
10	0.0416605
15	0.0101008
20	0.0024804
25	0.0006132
30	0.0001522
35	0.0000379
40	0.0000095
45	0.0000024
50	0.0000006

Calculating the performance of Bitcoin

Throughput

Block size $\times$ Mining rate

Latency

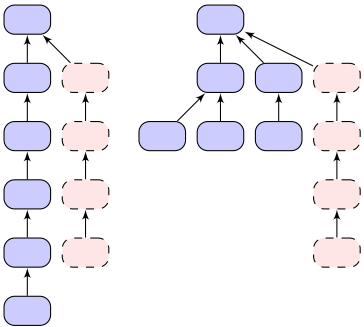
Confirmation depth (k) $\div$ Mining rate

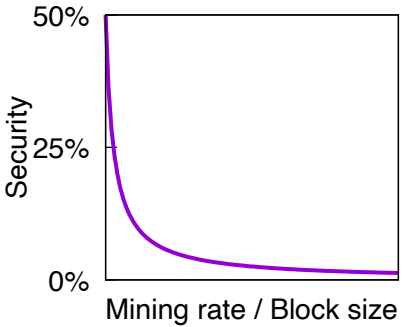
Increasing the mining rate or the block size causes forking

Forking happens when the next block is mined before the previous one is received.

- Mine faster $\rightarrow$ Shorter interval
- Bigger blocks $\rightarrow$ Slower propagation

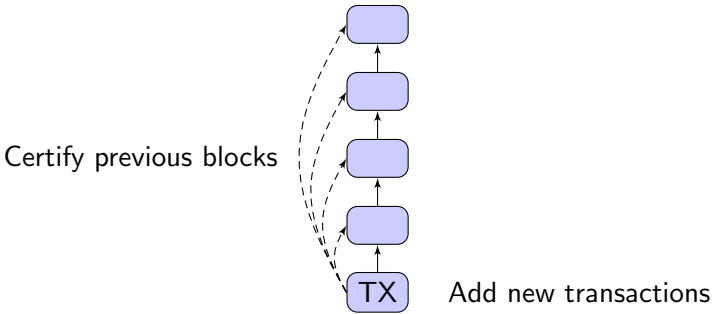
Forking compromises security



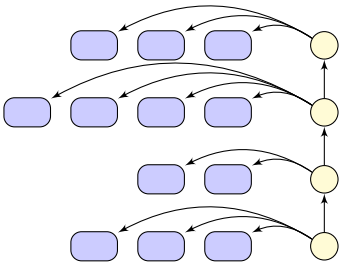


Bitcoin performance is coupled with security!

Deconstructing a block: proposing and voting



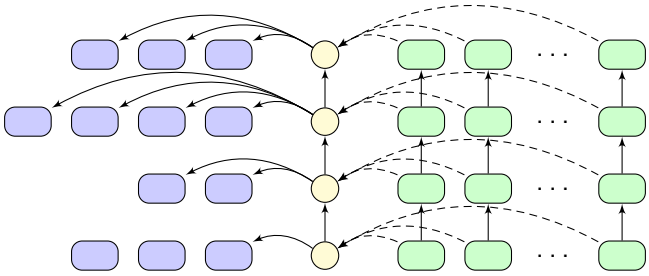
Scale proposing with lots of transaction blocks



Forking does not matter among transaction blocks!

Throughput ☺, latency ☹

Scale voting with lots of voter chains



- 1 voter chain: 25-deep ☹
- 1000 voter chains: 2-deep ☺

Prism is provably fast and secure

Adversary power $\beta < 0.5$
Network bandwidth C, network latency D, confirmation reliability ϵ, m voter chains

- Security: consistency and liveness
- Throughput: $(1-\beta)C$
- Confirmation Latency: $O(D) + O\left(\frac{-D\log(\epsilon)}{m}\right)$

Understanding the real-world performance of Prism

- Protocol more complex than Bitcoin
- Theory based on simplified assumptions
- Unknown constants in confirmation latency
- Other bottlenecks besides network

This talk

The Prism consensus protocol

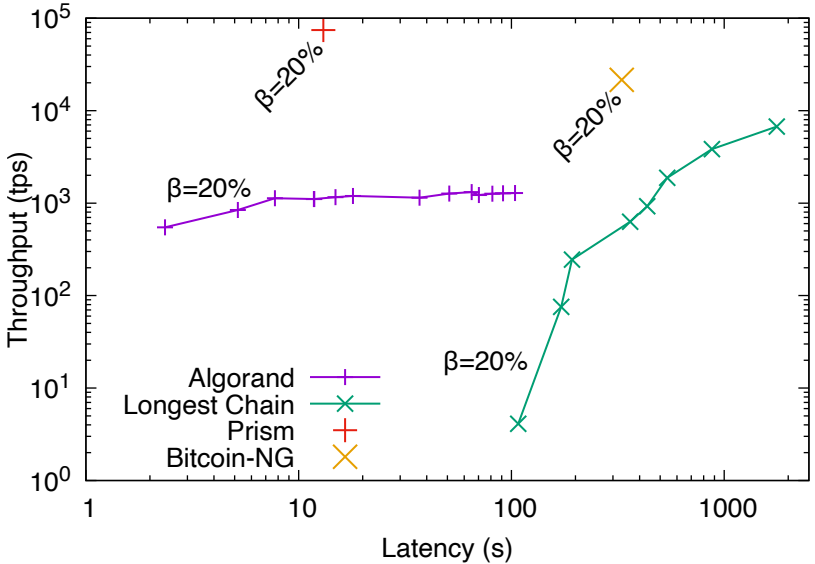
System implementation

Evaluation results and findings

Implementing Prism in Rust

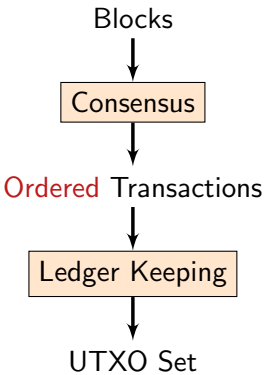
- 10,000 lines of Rust
- Unspent Transaction Output (UTXO) model
- Pay-to-public-key transactions
- Code available at t.leiy.me/prism-code

Prism: ∼70,000 tps and ∼10s latency



Blockchain client: consensus and ledger keeping

- Consensus: keep track of the blockchains and mine
- Ledger keeping: process transactions and calculate the UTXO set



Achieving high throughput

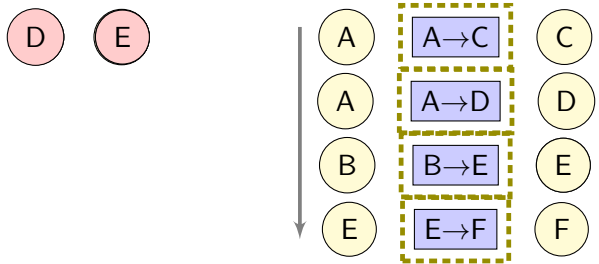
Lessons

- Bottlenecks: ledger keeping, SSD
- Must parallelize

Optimizations

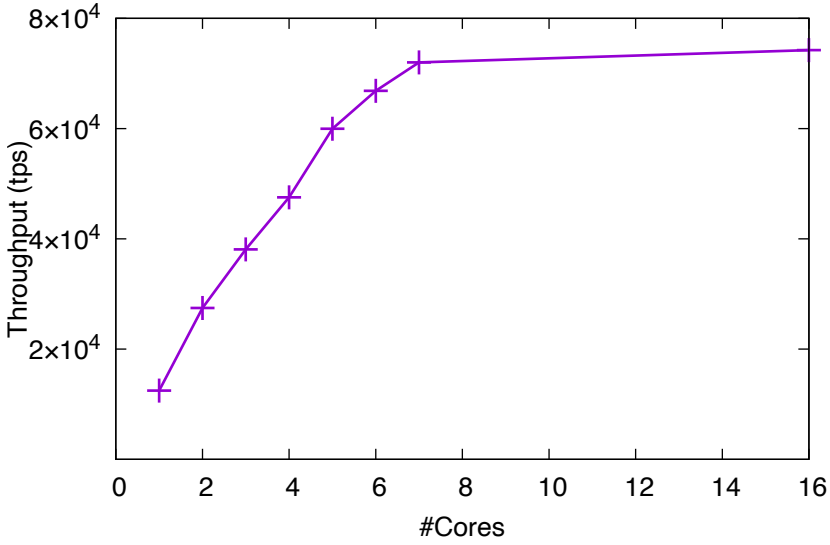
- "Scoreboarding"
- Async ledger keeping
- Functional design pattern
- No transaction broadcasting

Parallelize ledger keeping with scoreboarding



Scoreboard: A, C A, B, C, E A, D A, D, E, F

Parallelize ledger keeping with scoreboarding



Parallelize consensus with async ledger update

- Ledger updates are “infrequent”
- Don’t do it every new block

The Prism consensus protocol

System implementation

Evaluation results and findings

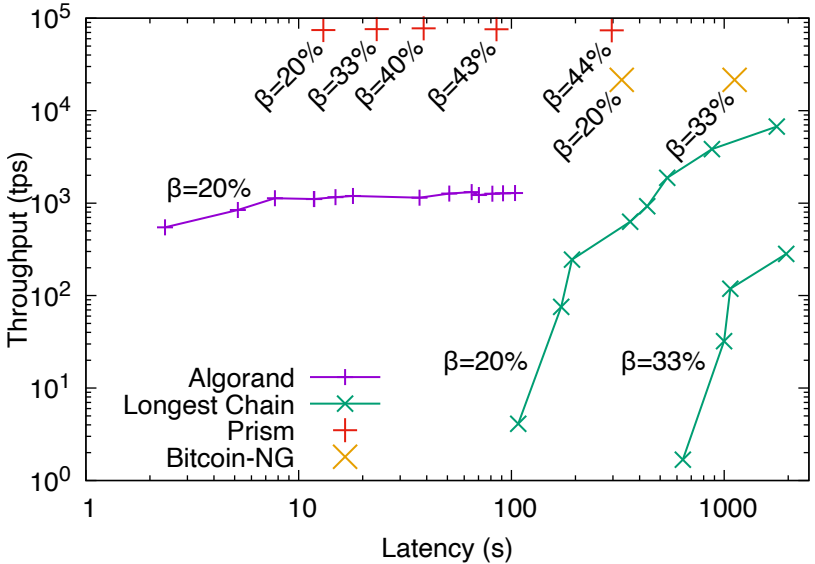
Testbed setup

- 100 - 1000 AWS EC2 instances
- Random 4-regular graph
- 120ms propagation delay
- 400 Mbps bandwidth

Experiments

- Comparison of throughput, latency with Algorand, Bitcoin-NG, Longest Chain
- Scalability to 1000 nodes
- CPU, bandwidth utilization
- Censorship, spamming, balancing attacks

Comparison with Algorand, Bitcoin-NG, Nakamoto



Our implementation is efficient

CPU

- Signature check: 22%
- RocksDB: 53%
- Data serialization: 7%

Bandwidth

- Transaction blocks: 99.5%
- Voter blocks: 0.4%
- Proposer blocks: 0.1%

Takeaways

- Deconstruct blockchain into proposing and voting
- Scale each part to the physical limit
- Must parallelize everying in the client
- Current bottleneck is ledger keeping and SSD
- Simple and natural extension of Natamoto

Resources

- Code: t.leiy.me/prism-code
- Theory Paper: Deconstructing the Blockchain to Approach Physical Limits
- System Paper: t.leiy.me/prism-paper
- Online Demo: t.leiy.me/prism-demo