Towards Horizontal Scalable Information Propagation and MOCA Consensus in Massive-Scale Networks

Yilun Zhang

NKN Labs

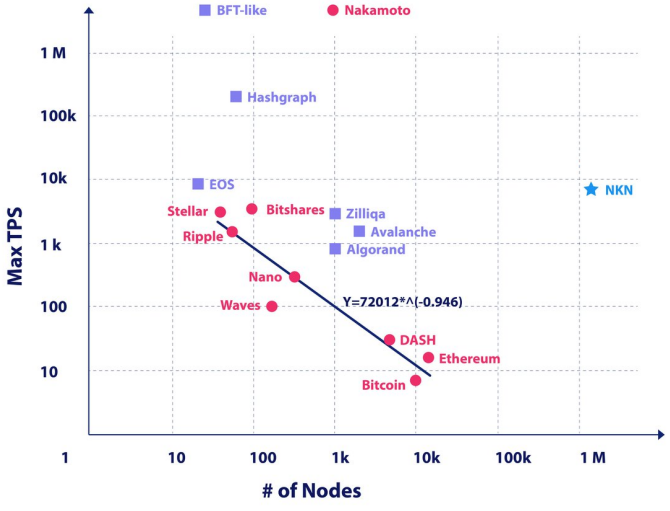
Oct 23, 2019

NKN

What Do We Mean By Scalability?

- Transaction per second (TPS)
  - Transaction throughput
- Network size
  - Decentralization
  - Robustness
  - Security
  - Aggregated capacity of off-chain service

Throughput - Scale Tradeoff



Challenges to Achieve High Scalability

- Necessary steps to produce a block:
  - Transaction broadcasting
  - Block broadcasting
  - Consensus
- Under the condition:
  - Number of nodes $N \sim$ or $>$ TPS
  - Nodes are sparsely connected, number of neighbors $M \ll N$

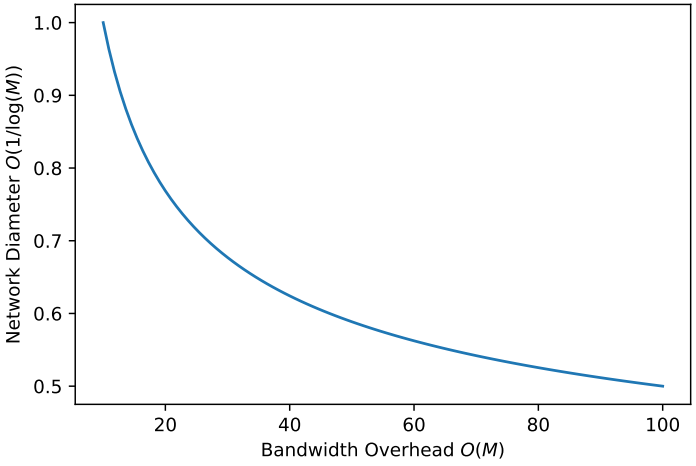
Gossip Protocol

- The "standard" way of broadcasting msg
- In a nutshell:
  - Each node sends msg hash/id to neighbors after receiving msg (I have)
  - Each node received msg hash but not msg itself requests the msg from one neighbor (I want)
- Each node will receive msg itself only once
- Each node will receive msg hash M times (overhead)
- Near optimal bandwidth usage when msg size $\gg$ hash size $\times M$

Transaction Broadcasting

- Transaction (txn) size is small:
  - Simple Bitcoin transfer txn size: 200+ bytes
  - Simple Ethereum transfer txn size: 100+ bytes
- Txn hash size: 32 bytes
- Other msg overhead: $\sim$10 bytes
- $M \sim 80$ in our current Mainnet
- Gossip overhead is huge: over 30$\times$
- One can use part of neighbors but does not change the conclusion

Latency - Throughput Tradeoff



NKN

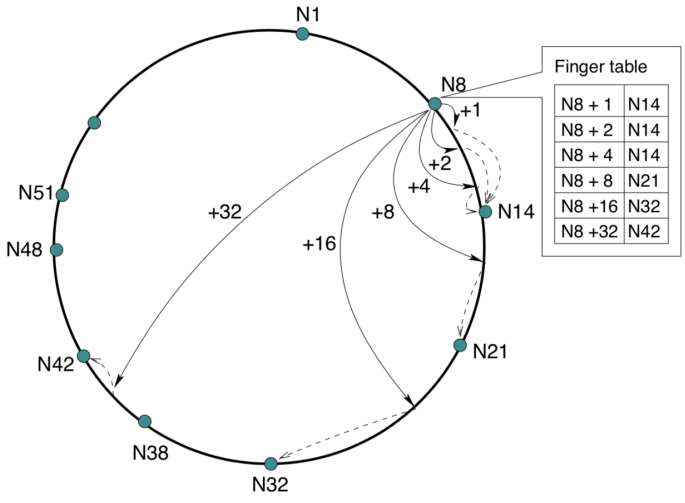
Can We Bundle Multiple Transactions Into One Message?

- Txn needs to be broadcasted ASAP to minimize confirmation time
- Bundle multiple txn into one msg to reduce Gossip overhead in small network $N \ll$ TPS
- Not realistic in large network $N \gg$ TPS if txn are evenly distributed (which we do want to see)

Bottleneck of TPS

- Bandwidth usage of txn broadcasting using the previous numbers:
  - $TPS = 1,000$: 3+ MB/s (byte, not bit) upload and download (possible in lab environment, not practical in public blockchain)
  - $TPS = 10,000$: 30+ MB/s (byte, not bit) upload and download (hard to achieve even in lab environment unless geolocational clustered)
- Bottleneck of TPS in large permissionless public blockchain
- Other bottleneck not related to scalability: signature verification, database

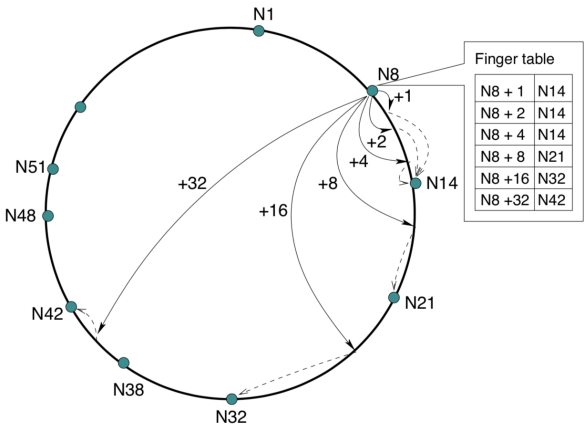
Efficient Broadcasting Based on Chord DHT



NKN

Efficient Broadcasting Based on Chord DHT

- Forward table: msg from ith finger table is forwarded to $1 \sim (i-1)$th finger table

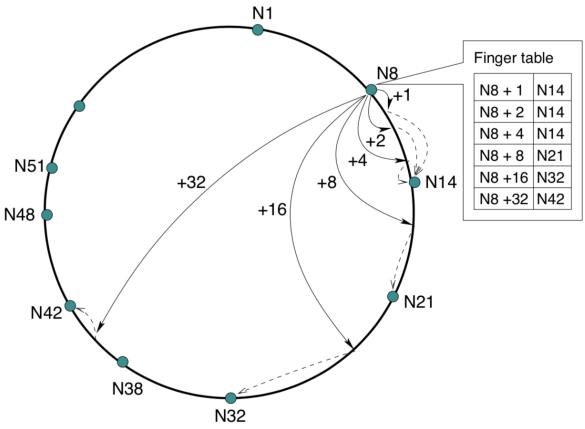


Efficient Broadcasting Based on Chord DHT

- Construct a spanning tree from any node using Chord finger table
- Each node receives msg once
- Each node sends msg once on average, at most $O(\log N)$ times
- Zero bandwidth overhead
- Time to finish propogation: $O(\log N)$
- Challenges:
  - Original Chord DHT is not robust (cannot tolerate $O(N)$ simultaneous node failure)
  - Delivery is not reliable with failed/malicious nodes

Extending the Finger Table

- Change each finger table item to an array of K nodes
- K determines redundancy and fault tolerence



NKN

Transaction Broadcasting Can be Unreliable

- Less robust than Gossip
- Transaction broadcasting does not need to be reliable: partial delivery of txn only increases confirmation time
- Ideal for txn broadcasting

Bandwidth Usage Recompute

- Bandwidth usage of txn broadcasting assuming 150 bytes per txn:
  - $TPS = 1,000$: 150 KB/s (byte, not bit) upload and download (doable in public blockchain)
  - $TPS = 10,000$: 1.5 MB/s (byte, not bit) upload and download (hardable but still feasible in public blockchain)
- Bandwidth usage grows with $O(\log N)$
- Boost TPS by 20$\times$ given that bandwidth is the bottleneck
- Pure network layer, blockchain agnostic

Block Broadcasting

- Block is large, Gossip is considered to be good enough
- Most part of a block is just transactions, leading to 2x bandwidth usage
- How can we reduce redundency in block propogation?
  - A sends B block header + transaction list (short hash)
  - B sends A missing transaction list (short hash)
  - A sends B missing transactions
- Bandwidth usage: $\sim 5\%$ of transaction broadcasting

BFT-like Consensus

- Why choose BFT-like consensus?
  - Near-instant finality
  - Energy efficient and provide addition service
  - Less proposal $\rightarrow$ higher throughput
- However, not scalable in terms of network size

Why BFT is Not Scalable?

- Consider PBFT with N fully connected consensus nodes
- Every node needs to send msg to every other node for each election
- Every node sends and receives $O(N)$ message per block
- Every node needs to maintain $N-1$ connections, can be reduced at the cost of higher message complexity

How To Make BFT-like Consensus Scalable

Bottleneck:

- $O(N)$ message complexity per node
- $O(N)$ total message size per node

Approach:

- Leader relay
- Aggregated signature
- Reducing N: elect a smaller group of consensus nodes efficiently
- Reducing $O(N)$: each node only communicates with a smaller group of other nodes (neighbors), and only collect neighbor votes (our solution)

Inspired By Cellular Automata

- Cellular Automata: a network of state machine, each node updates its state based on its local neighbors' states
- Collective behavior emerged from local interaction
- Global consensus is a special case
- New Kind of Network (NKN) is inspired by Stephen Wolfram's New Kind of Science

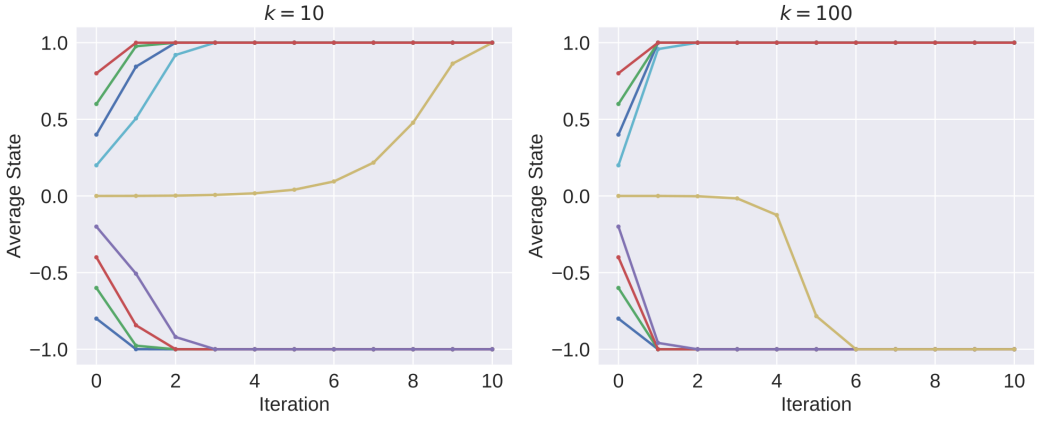
MOCA: Majority vOte Cellular Automata

- We proposed Majority vOte Cellular Automata (MOCA) consensus algorithm:
  - Each node selects a random set of neighbors
  - For a proposal (e.g. block), each node sends out his initial vote (e.g. accept/reject)
  - Each node listens for his neighbors' vote. Whenever majority of them have different vote, he changes his vote and sends the new vote to his neighbors.

Selecting Random Neighbors

- Requirement for neighbor selection:
  - Random: security + faster convergence (important!)
  - Verifiable: security
  - Sparse: scalability
- Our choice:
  - Unpredictable, uncontrollable, verifiable, delayed node ID generation using VRF
  - Chord DHT topology: verifiable and scalable ($O(\log N)$ neighbors)
  - Random initial condition: separate block propagation topology and voting topology
  - Random weight: hard to predict honest nodes' behavior

Convergence

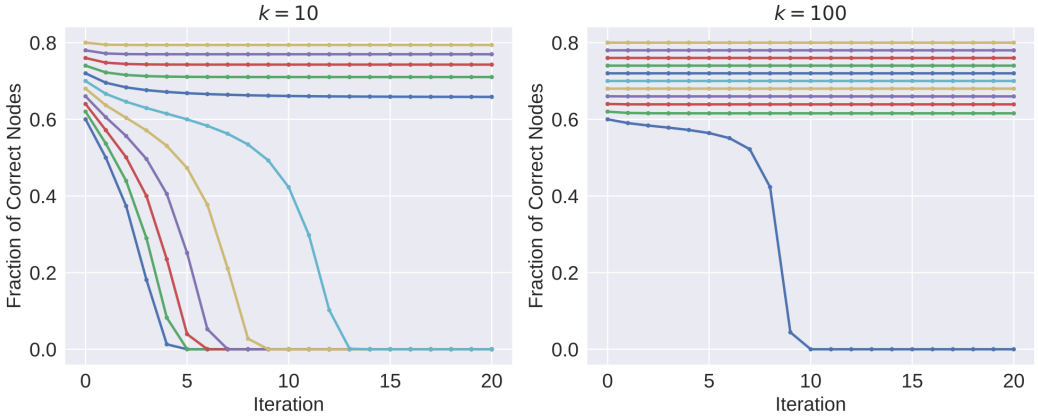


NKN

Scalability of MOCA Consensus

- Message complexity: $O(\log N)$ per node per block, horizontal scalable!
- Reach consensus in just a few seconds
- Our permissionless mainnet: 20k+ global nodes, almost ZERO resource comsumption (CPU, RAM, Network)
- To reach one MILLION nodes: 40% more resources comsumption
- To reach one BILLION nodes: 210% more resources comsumption

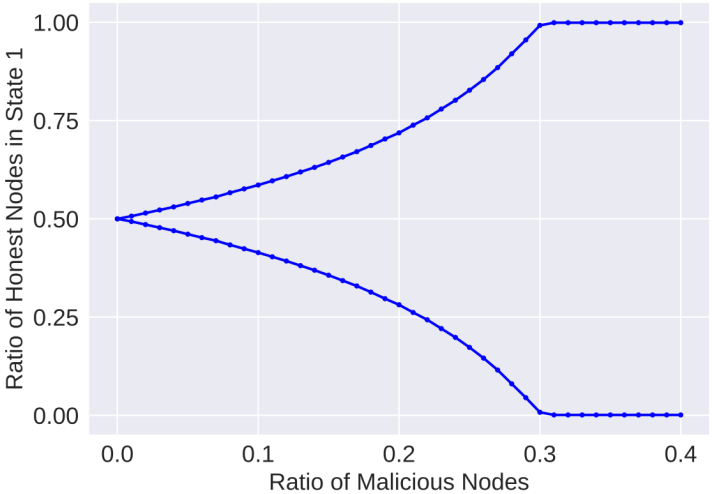
Fault Tolerance



NKN

Conditional BFT

What do we trade scalability for?

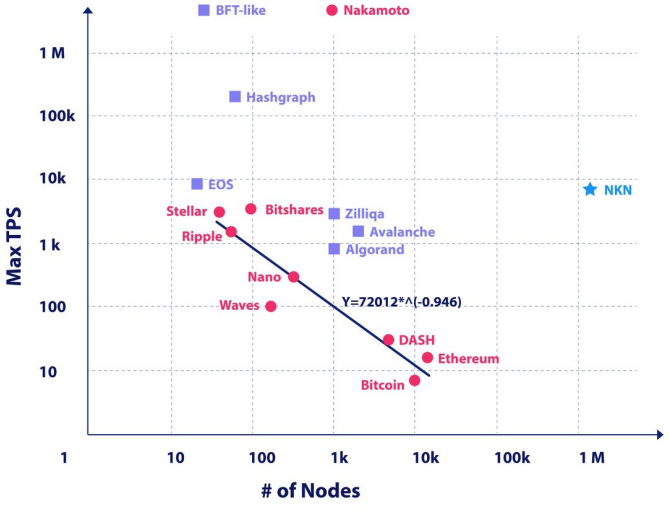


NKN

Rules For Selecting Block Proposer

- Proof of relay: select globally "luckiest" signature chain
- Unpredictable, Uncontrollable, Verifiable
- Verifiable random function (VRF): Uniqueness
- Optimization: constant size overhead ($\sim$ 32 bytes)

Pushing the Limit



NKN

Summary

- Efficient transaction broadcasting with optimal throughput and latency, boost TPS by 20× for public, large scale blockchain
- MOCA: conditional BFT consensus algorithm that is horizontal scalable to ANY number of nodes
- Everything has been implemented, open sourced, and tested with 20k+ global community nodes
  - Full node: https://github.com/nknorg/nkn
  - P2P network layer: https://github.com/nknorg/nnet
  - Client/wallet SDK: JavaScript, Go, Java...

Join us!

- **We are hiring!**
  - R&D intern/full-time
- Website: www.nkn.org
- Email: contact@nkn.org