A Decentralized Blockchain with High Throughput and Fast Confirmation

Chenxing Li*, Peilun Li*, Dong Zhou, Zhe Yang†, Ming Wu†,
Guang Yang†, Wei Xu, Fan Long‡‡, Andrew Chi-Chih Yao

Tsinghua University †Conflux Foundation ‡University of Toronto

*The first two authors contributed equally.
An Ideal Blockchain System

- **Robustness**
  - *Safety* against double spending attacks
  - *Liveness* against denial of service attacks

- **Performance**
  - High *throughput*
  - Fast *confirmation*

- **Decentralization**
  - *Scale* to large amount of participants
  - *Permissionless* to join and leave
Blockchain Performance Problem

- Transactions per Second:
  - Bitcoin: ~7
  - Ethereum: ~30
  - PayPal: ~200
  - VISA: ~3000

- Confirmation Latency:
  - Bitcoin: 1 hour
  - Ethereum: 7~10 minutes
  - PayPal: Few seconds
  - VISA: Few seconds

Undesirable user experience, long processing delay, and skyrocketing transaction fees!
Bitcoin and Ethereum Background

The Blockchain: A record of transactions

Secured by “Proof-of-Work”

Sybil Attack
Standard Nakamoto Consensus

• **Longest-chain**: all participants agree on the longest chain as the valid transaction history
  ▪ Security assumption is >50% computation power owned by honest nodes.

• **Slow/small** block generation
  ▪ Bitcoin: 1MB block per 10 minutes
  ▪ Ethereum: ~100KB block per 15 seconds
What if we run Nakamoto Consensus with large blocks or fast generation?
• Mining are concurrent and block broadcast has delay
• Larger block size/faster block-gen rate -> more forks

• Forks waste network/processing resources
• Downgrade safety
Longest Chain is **Not Safe** with Fast Generation

Suppose the Longest Chain has 10% of blocks

Attacker with more than 10% computation power will be able to revert the longest chain
GHOST Consensus

• Proposed by Sompolinsky et. al., ICFCDS’15, adopted partially by Ethereum

• Heaviest subtree rule

• Start from the Genesis block

• Iteratively advance to the child block with the largest subtree
1. Start from the Genesis block
2. Iteratively advance to the child block with the largest subtree

**GHOST Rule:**

All the blocks including forked blocks contribute to the chain selection
GHOST is Weak to Liveness Attacks

- When the block generation is much faster than $d$, an attacker with little computation power can stall the consensus forever!
One Fix: Structured GHOST Approach

• Only \(1/h\) of blocks have weights for chain selection
  ▪ Remaining blocks only contribute transactions

• Secure against liveness attacks if \(h\) is large enough
  ▪ Because concurrent generation of weighted blocks is rare

• Cons: Slow confirmation!
  ▪ Need to wait for enough weighted blocks being generated to confirm
Greedy Heaviest Adaptive SubTree (GHAST)

• Assign different weights to generated blocks
• Select **pivot chain** using heaviest subtree rule and decide total order of all blocks based on the pivot chain.
• In normal scenarios, assign **equal** weights to all blocks
  ▪ Operate like GHOST
  ▪ Achieve near optimal throughput and confirmation latency
• When attack happens, assign **high weights** to a **small subset** of blocks
  ▪ Operate like structured GHOST
  ▪ Slow confirmation to ensure consensus progress
How to make honest participants automatically switch between two scenarios?
Conflux operates with a **Tree-Graph** structure.

Each block has one **parent edge**.
E admits that D is generated before E

Each block may have multiple reference edges

Reference edges simply denote happens-before relationships
When generating a new block:

1. Select the last block in the pivot chain as the parent
2. Create reference edges to all other blocks without incoming edges
Edges in the Tree-Graph capture the history blockchain state for each generated block

The past-subgraph of a block → All blocks that the block generator saw
Determine Weights from Past Sub-graphs

Is the past sub-graph stable enough?

Yes: Assign weight 1
No: Assign weight $h$ for $1/h$ blocks, 0 for other blocks

$\mathbb{f}(\text{generated block}) = \text{Weight of the generated block}$

- All honest participants will agree on the weights
- Even with the presence of attackers!
Determine Sub-graph Stability

• **Rationale**: For any pivot chain block $A$ that is generated long enough, one of its child $A'$ must become **dominant**

• Most future blocks after $A$ should accumulate under the subtree of $A'$
Trusted Block Generation Time: TimerChain

• TimerChain: a blockchain embedded in TreeGraph with longest-chain rule and low generation rate.
  ▪ A small subset of blocks (Timer Blocks) have weights, like structured GHOST
• Block generation time: the height of the latest Timer Block in its past.

Past of A

GenerationTime(A) = TimerChainHeight(C)
Conflux Ordering Algorithm

• **Key Idea:** deterministically define a block total order of a Tree-Graph based on a chain

• First use GHAST to agree on a *pivot chain* of blocks

• Then extend the agreed pivot chain into a total order of all blocks in the Tree-Graph
1. Each pivot chain block forms one epoch

2. An off-chain block belongs to the first epoch whose corresponding pivot chain block happens after it.

D belongs to the epoch of E, because D happens before E but does not happen before C.
1. Order based on epoch first
2. Topologically sort blocks in each epoch
3. Break ties based on block id

Block Total Order: **Genesis, A, B, C, D, F, E, G, J, I, H, K**
Implementation & Optimizations

• Implemented in Rust with a modified EVM to handle smart contract transactions.

• Several key optimizations:
  • Link-cut tree and lazy validation
    ▪ Efficiently maintain weights in Tree-Graph
  • Deferred execution
    ▪ Avoid redundant execution rollbacks
Evaluation
Experimental Environment

- Run up to 12k Conflux full nodes on Amazon EC2 m5.2xlarge VMs
- Limit the bandwidth of each full node to 20Mbps
- Simulate network latency between full nodes

- Measure the achieved throughput and confirmation latency
  - Consider a block confirmed if its confidence is the same as waiting for 6 Bitcoin blocks
Throughput, Latency, and Scalability

- 300K block size and 4 block per second.
- Conflux achieves 9.6Mbps throughput.
- Up to 32X GHOST throughput.
- Confirm transactions on avg. 51.5 seconds.
- Scales to 12k full nodes.

Run up to 15 full nodes per EC2 VM and disabled transaction executions.
Conclusion

• Conflux achieves both high throughput and fast confirmation.

• Conflux is safe against both double spending and liveness attacks.

• Conflux achieves this with a novel consensus protocol GHAST, which assigns different weights to blocks adaptively and automatically.

• With 12K nodes, Conflux can reach 9.6Mbps throughput and confirm blocks within one minute.
Thanks!

Presenter Email: lpl15@mails.tsinghua.edu.cn
Conflux Website: https://www.conflux-chain.org/