StreamChain: Do Blockchains Need Blocks?

Zsolt István, Alessandro Sorniotti, Marko Vukolić

IMDEA Software      IBM Research Zürich
StreamChain in a nutshell

• **Goal**: Low latency *and* high throughput operation in permissioned ledgers for wider adoption (without changing security or reliability properties)

• **Idea**: Revisit core design decisions → turn block-based processing into streaming processing

• **Enables**: New opportunities for blockchains, ability to benefit from recent hardware trends
The lineage of permissioned ledgers

• Public ledgers (blockchains)
  • Geo-distribution → no way around communication latency, gossip to keep everyone up to date
  • Proof-of-work → amortize cost by packaging up many TXs in blocks

• Permissioned ledgers
  • Compelling non geo-distributed use-cases
    • Low latency, high bandwidth, gossip not necessary
  • No proof-of-work

Pain point: When executed inside the same datacenter, permissioned ledgers still take hundreds of milliseconds for transaction finality!
The source of high latency

a) “Block” behavior

b) Streaming behavior
StreamChain – Design principles

• Process transactions system-wide as they arrive
  • Reduces latency without impacting throughput

• Use batching to hide the cost of high-latency operations (disk accesses)
  • Logical “blocking” of transactions and batching are decoupled

• Use multi-core parallelism to speed up cryptographic operations
  • Streaming doesn’t change the cost of these...
Hyperledger Fabric 101

- Open source platform for building applications on top of a permissioned ledger
  - Smart contracts as “chain code” written in various languages
  - Customizable behavior
  - Separates ordering of transactions into dedicated service – pluggable implementations for BFT
Executing transactions in Fabric

- Has an EOV model to save resources, provide confidentiality
  - **Execute**: Choice of endorsers depends on a user’s endorsement policy and produce R/W set of the TX
  - **Order**: Orderer orders the transactions (R/W sets signed by endorsers), signs blocks
  - **Validate**: Nodes apply R/W set if endorsement is valid and compatible with state
Life after Ordering in Fabric

• Fabric can have *failed* transactions due to R/W set conflicts
  • Client have to retry transaction
  • (Or use a suitable programming model)

• The less latency between execution and validation, the less chance of failing TX
  • StreamChain brings this additional benefit in Fabric
Sketch of StreamChain in Fabric

Endorsement of chain codes

Pipelined execution of Validate step

TXs from Orderer

Sign. Valid.

Streaming

P/W Set Validation

S

Write to Ledger

L

Batching

Peer
Our Proof of Concept

• Modifies Fabric v1.0 code to simulate behavior
• Streaming by making blocks with 1 TX and null signatures from CFT ordering service
  • Still relies on TLS connections
  • Cost of Orderer signature checking per block is negligible compared to TX signatures
• Implemented parallel signature checks on TXs in the peers
• Simulating amortized cost of disk access using RAMdisk
Does this work with ordering service failures?

- For CFT: Connections to ordering nodes set up via TLS
  - Can rely on single ordering node until crash

- For BFT: If each node connects to $t+1$ ordering nodes: data can be streamed from one, hashes from the others
  - High bandwidth requirement, many connections
Does this work with a BFT ordering service?

• If connecting to only one ordering node, transactions cannot be recorded to ledger as they arrive
  • Multi-signature required periodically

• Can speculate on state in the meantime – explained in the paper
  • Make transaction outcome immediately visible to execution logic
  • If signature is wrong, remove temporary state

• May waste work but no data corruption possible on ledger
Evaluation

• Ran **StreamChain** in the IBM Cloud (9 machines)
  • Intel Xeon E5-2683 @ 2GHz
  • SSD storage
  • 1Gbps network

• Compared to **Fabric (Fabcoin)** [Eurosys18]
  • UTXO application
  • ~4000TX/s, ~350ms end-to-end latency
  • (Related work has similar orders of magnitude)
Latency
Throughput vs Latency

Throughput bound by R/W set check and ledger commit.

Fabric (Fabcoin)
StreamChain P.o.C.
Future expectation
Thoughts on the future

• Permissioned ledger adoption could hinge on performance
  • Revisit assumptions: streaming processing is a realistic option
  • Proof-of-concept using Hyperledger Fabric

• StreamChain exposes new bottlenecks $\rightarrow$ New research challenges
  • Ordering service optimizations
  • Smart contract execution

Birds of a Feather Session tomorrow: Consensus and coordination using modern hardware