Simulation analysis

leios-2025w23

Two experiments with pseudo-mainet

1. Transaction lifecycle (Rust only)
2. IB diffusion (comparison of Rust vs Haskell)

Pseudo-mainnet

This is the first cut at a realistic mainnet-scale topology for Leios, but it likely contain imperfections because several compromises were made during its construction, so as to smooth out inconsistencies in source data.

- Realistic stake distribution
- Realistic number of stake pools
- Two relays for each block producer
- Block producers only connected to their relays
- 10,000 nodes total
- Realistic latencies, generally consistent with the RIPE Atlas ping dataset
- Bandwidth consistent with the low end of what is generally available in cloud data centers
- Node connectivity generally consistent with measurements by the Cardano Foundation
- Geographic distribution (countries and autonomous systems) consistent with measurements by the Cardano Foundation

Finding: *Creating a much more realistic topology would require simulating the p2p algorithm itself at mainnet scale, so that the topology would be an emergent property of the simulation.*

Metric	Value
Total nodes	10000
Block producers	2657
Relay nodes	7343
Total connections	298756
Network diameter	6 hops
Average connections per node	29.88
Clustering coefficient	0.122
Average latency	77.0 ms
Maximum latency	636.8 ms
Stake-weighted latency	0.0 ms
Bidirectional connections	10800
Asymmetry ratio	92.77%

Transaction lifecycle experiment

- Rust simulator (since Haskell does not model transactions)
- 100-node topology
- 8 vCPUs / node
- 10 slot / stage
- 3 shard / group
- 10 groups
- 1.5 EB/stage
- 1, 3, 10, 30, 100, 300 tx/s*
- IB generation probability varies with TPS
- Number of shards ≲ (IB rate) * (shards per group) * (groups)
- 327,680 B/IB maximum
- **No unsharded transactions**

Status of simulators

	Rust	**Haskell**
IBs only	● Panics, #397	● Assertion fails, #393
TXs	● 300 TPS at 200 seconds ○ 60 GB memory ○ 145 hours of CPU ○ 2.5B events ○ 405 GB log file	● Not supported

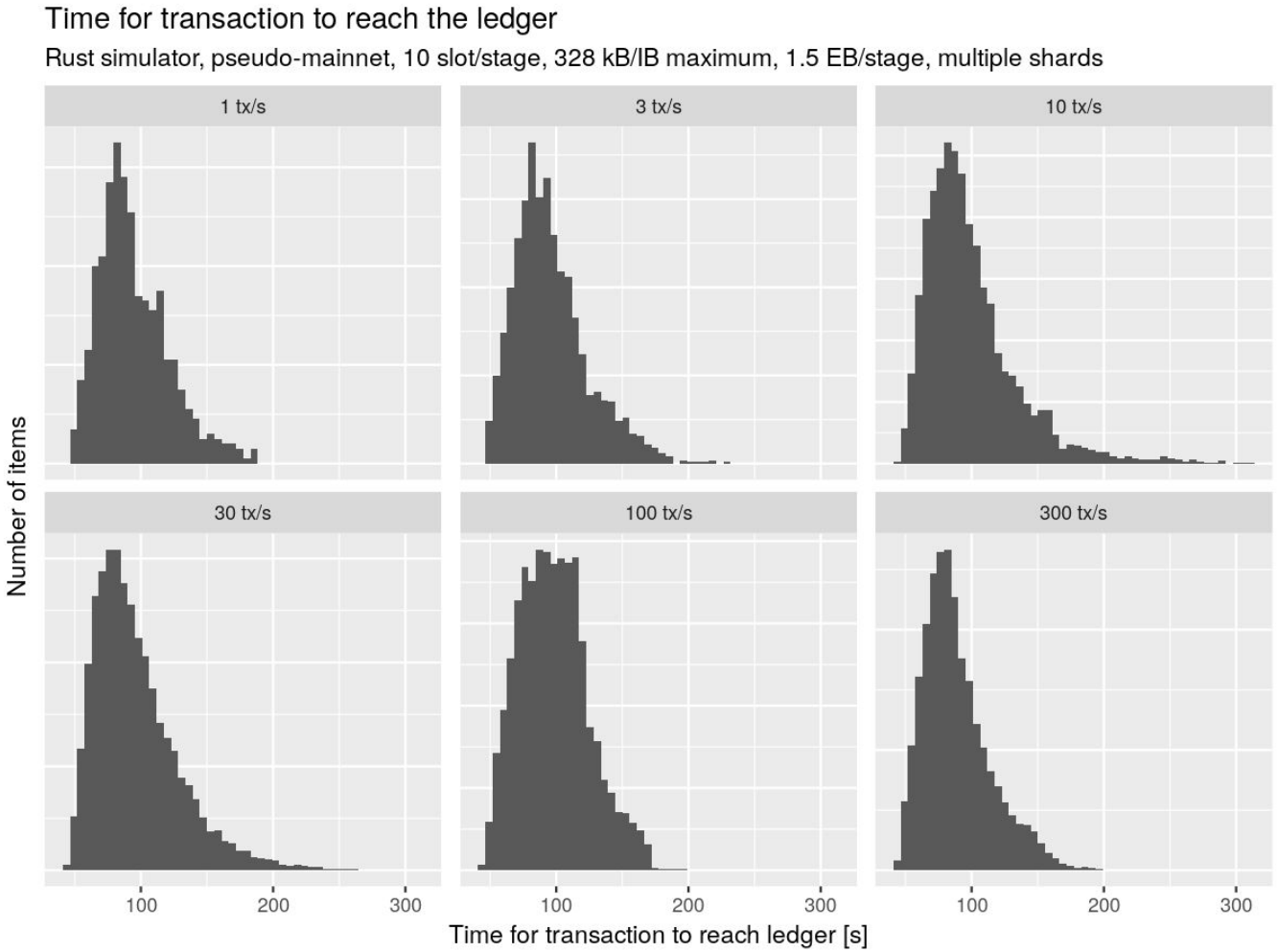
- The simulators are too slow and too resource intensive to run many pseudo-mainnet simulations.
- Strategy
  - Run **one** full set of pseudo-mainnet experiments using the Rust simulator.
  - Create a much smaller topology with similar diameter etc. to the pseudo-mainnet.
  - Verify that simulations on the mini-mainnet have similar characteristics to the pseudo-mainet.
  - Execute future simulations on the mini-mainnet.

Performance metrics

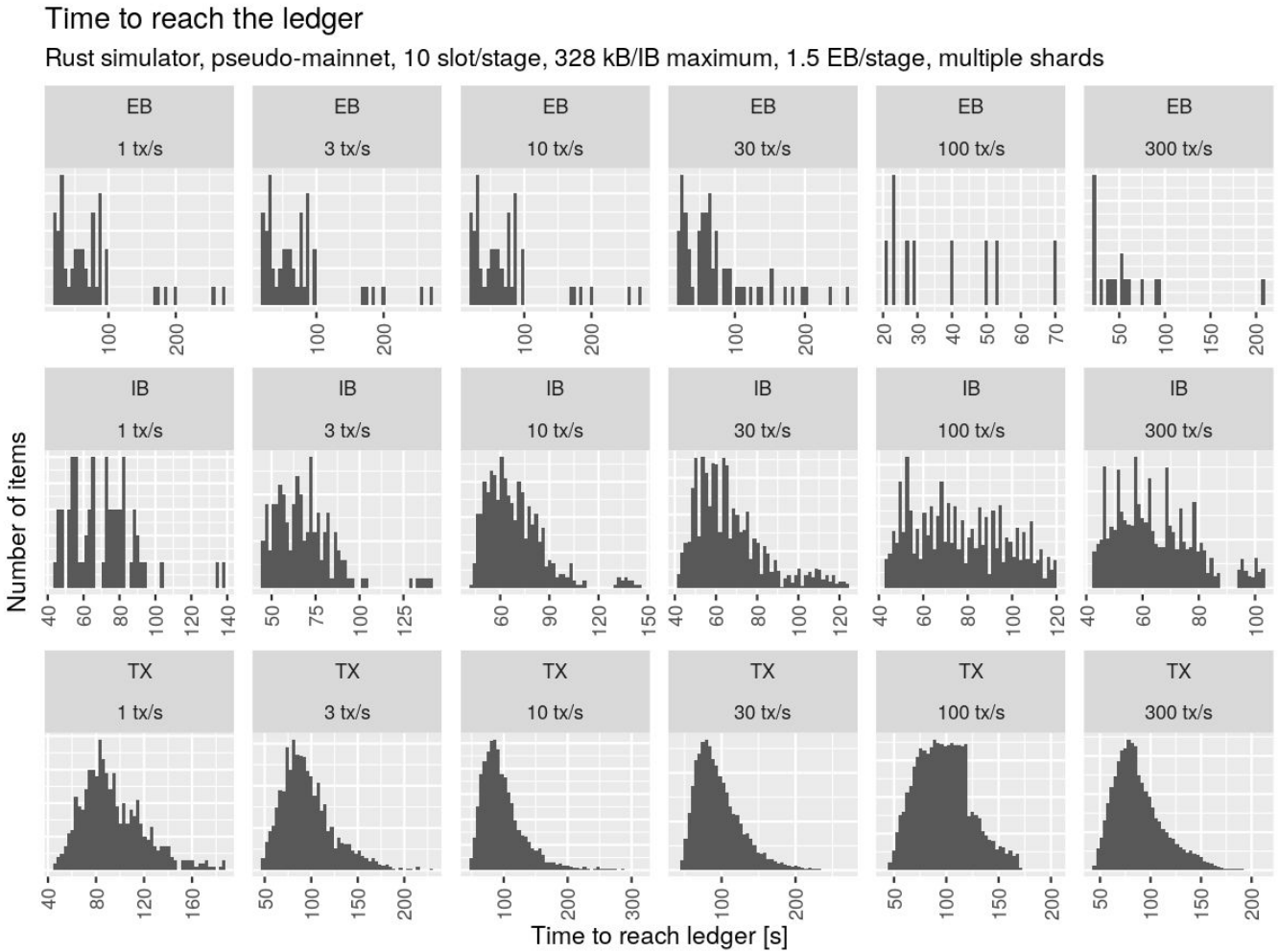
Performance is stable to 300+ TPS.

Nominal throughput	TX in IB	TX per IB	IB per EB	IB in EB	Spatial efficiency	TX redundancy	Time to IB	Time to EB	Time to ledger
1 TPS	1.111	9.150	4.921	4.156	65.781%	74.382%	27.031s	57.315s	94.524s
3 TPS	1.101	9.654	11.911	5.179	77.468%	73.573%	27.419s	57.381s	95.972s
10 TPS	1.087	10.076	37.978	5.250	81.222%	73.479%	30.601s	59.970s	99.317s
30 TPS	1.104	9.860	117.413	5.401	81.099%	81.029%	30.989s	58.858s	96.501s

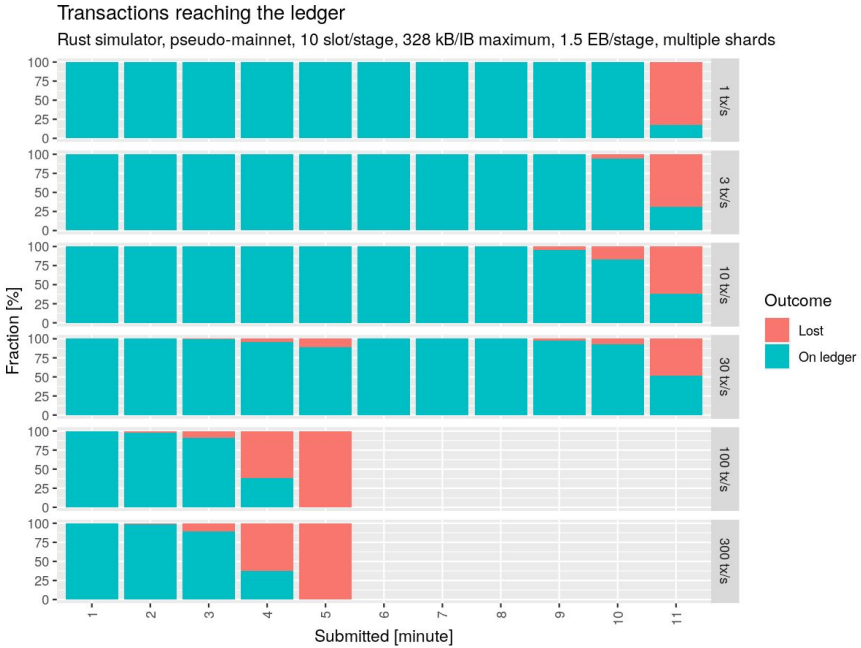
Some transactions take several minutes to first reach the ledger.

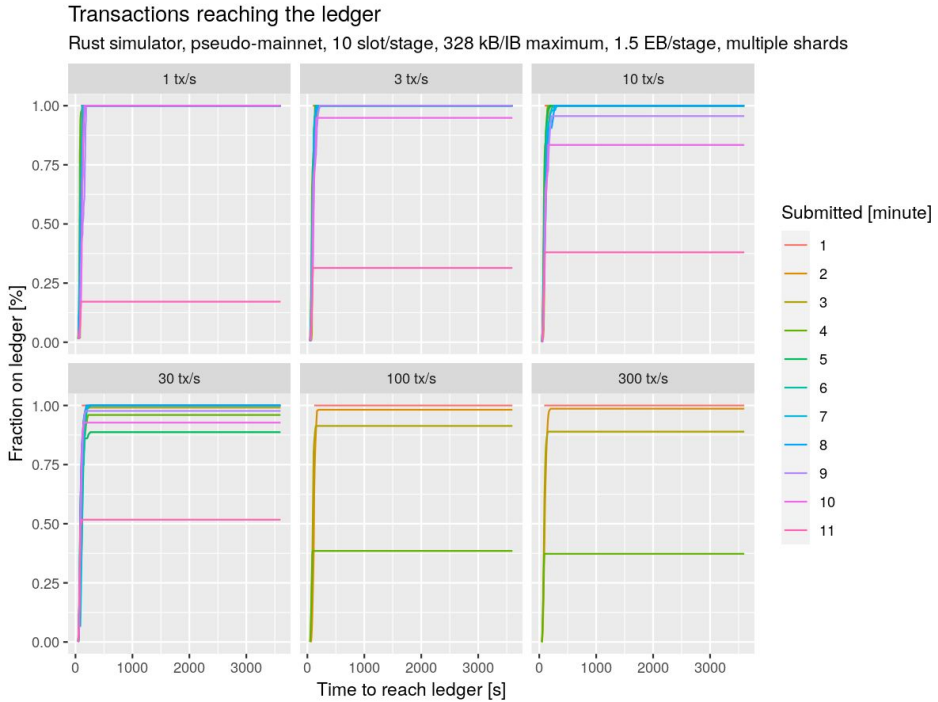


IBs and EBs take a moderate amount of time to reach the ledger.

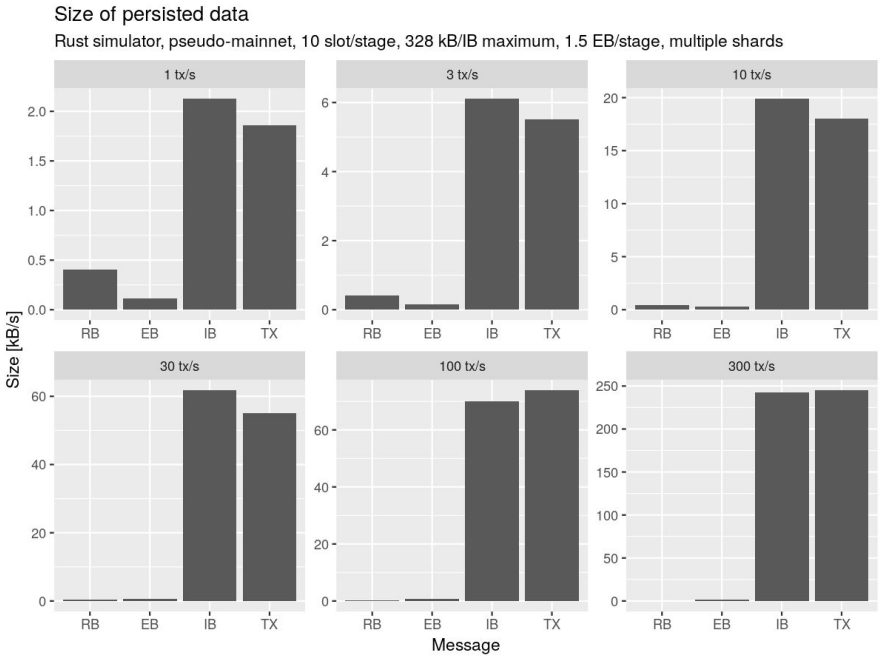


A few transactions never reach the ledger.

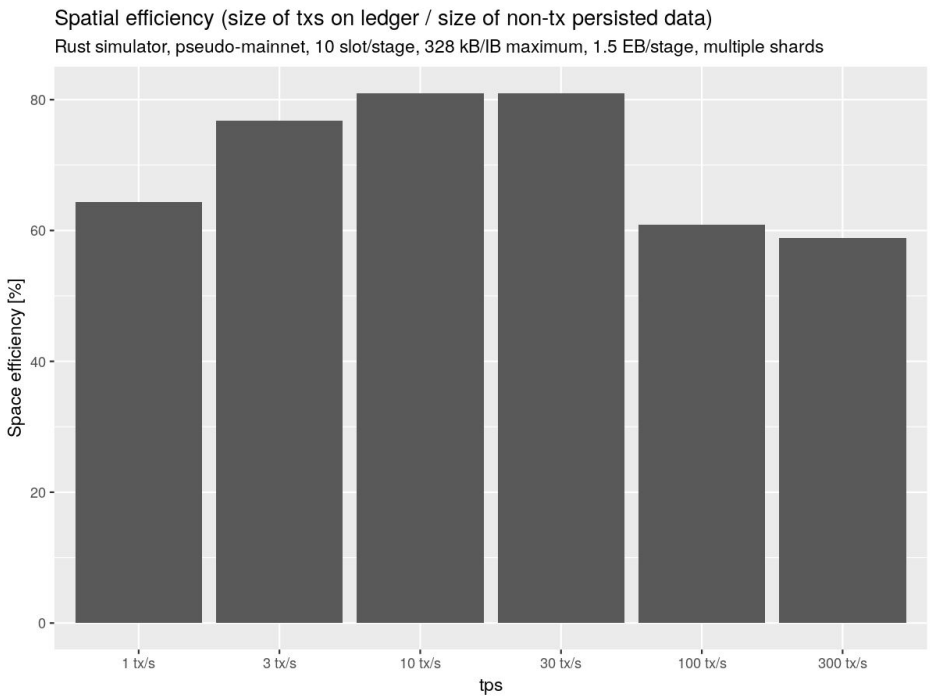




The total size of IBs is on the order of the size of the transactions.



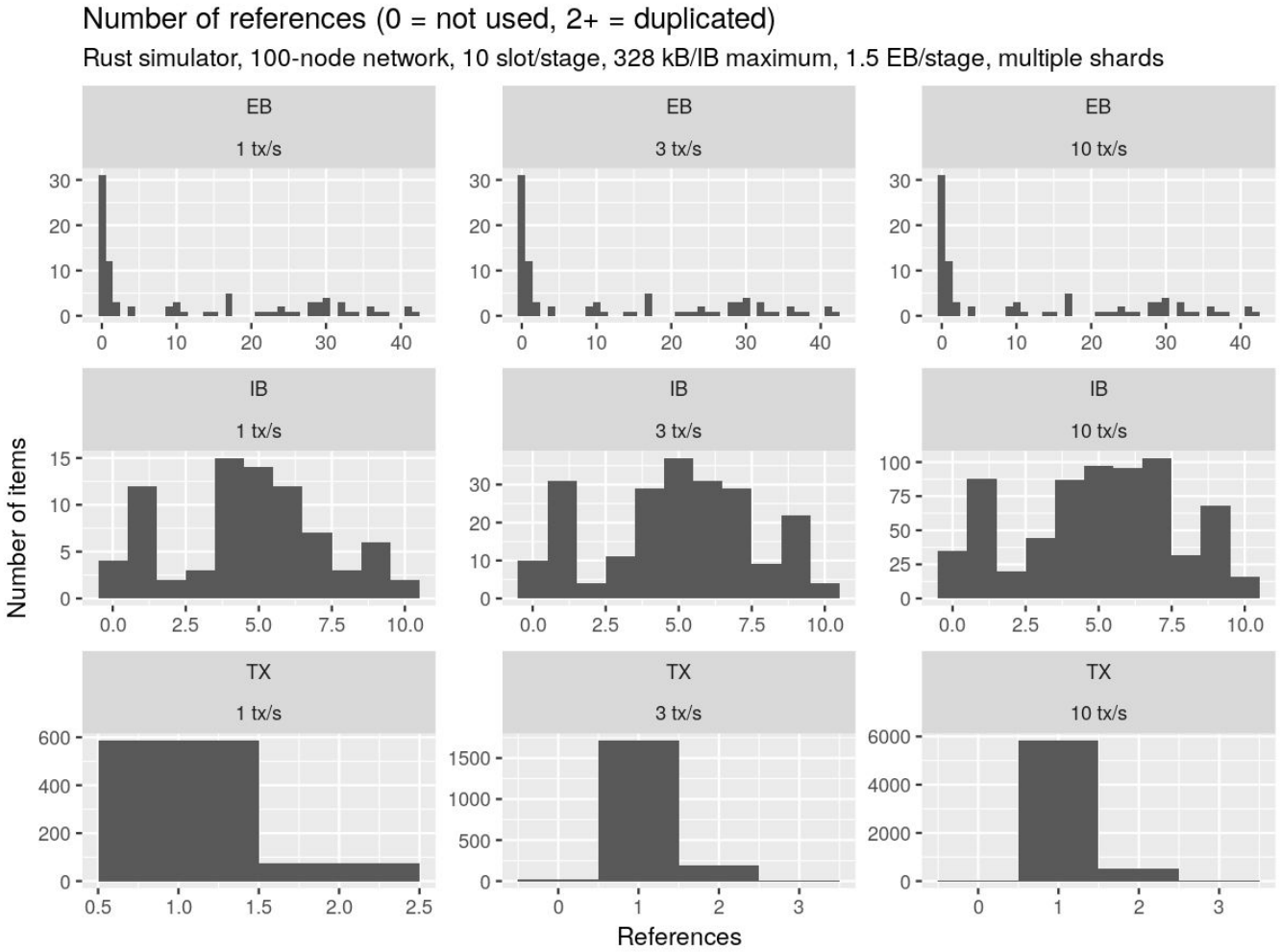
The amount of persistent storage is somewhat larger than the total size of transactions being stored.



EBs are often referenced by other EBs multiple times.

IBs are often referenced by EBs multiple times.

Transactions are occasionally included in multiple IBs.



IB experiment

- Compare Rust vs Haskell results on pseudo-mainet
- Blocked by bugs