BitShares Core & Network Performance

Past, Present and (possible) Future

I. The Past

- „100.000 transactions per second“ (2015)
- Real-life stress test (2017)

100k TX/s

- Claimed in a blog post in June 2015 https://bitshares.org/blog/2015/06/08/measuring-performance/
- Applies to **internal database operations** only
- **No networking** involved
- **No crypto operations** involved
- „Simple“ operations, i. e. no market order matching etc.
- Insert demo

3.3k TX/s

- Real-life stress test on 2017-03-15 15:00 UTC https://bitsharestalk.org/index.php?topic=23829.0
- Global-scale distributed test network with 15 witness nodes
- 90 minute test with 3-second blocks
- Max 60k ops / 10k tx per block -> 20k ops/s, 3.3k tx/s
- Insert demo

II. The Present

- 6.1M ops/day
- Chain state after ~35 months
- Replay time

6.1M ops/day

- Peak operations per day on BitShares mainnet
- 70 ops/s sustained – 10 times peak perfomance of BTC
- Mostly market operations due to bot activity
- Source: http://blocktivity.info

Chain state after ~35 months

- BitShares2 1st block at 2015-10-13 14:12 UTC
- 30 million blocks at 2018-08-28 06:07 UTC
- ~1 million registered accounts
- ~13.5 million transactions
- ~450 million operations

Replay time

- „Replay“ means re-apply all transactions in blockchain on top of genesis state
- Sometimes required after software upgrade
- Time with latest consensus-upgrade release (2.0.180612): 3.5h
- Various code optimizations (thanks @abitmore!)
- Time with latest release (2.0.180823): 1.25h

III. The Future

- Problem: Replay time
- Problem: Database
- Idea: Parallel crypto
- Idea: Fees
- Idea: Reorganization
- Idea: Separation

Problem: Replay time

- 1M ops/day -> +10s/day replay time **at best**
- New chain logic -> prolonged replay time
- More complicated market logic -> prolonged replay time
- Replay becomes impossible when chain activity maxes out

Problem: Database

- Database volume of full API node
  - Offloading history into ES helps
  - Can offload only static data, not accounts, orderbooks etc.
- Single-threaded execution model
- Blockchain logic requires sequential application of operations

Idea: Parallel Crypto

- Current bottleneck in live stress test is crypto
- Crypto operations can be parallelized!
- Requires some restructuring between P2P, API and DB
- Ongoing work, e. g. https://github.com/bitshares/bitshares-core/pull/1251
- **Doesn't** help for replay

Idea: Fees

- Handling fees consumes DB „bandwidth“
- Fees are usually in BTS
  - BTS balance objects must be changed often
  - Hinders parallelization („lock contention“)
- Possible solution: zero-cost rate-limited transactions, as in STEEM

Idea: Reorganize transaction processing

- Re-define execution order of operations within a block
- Separate interdependent operations
- Carry out independent operations in parallel, in map-reduce fashion
- Helps for live network and replay (if it works)
- **Speculative!**

Idea: Separate Processes, pt 1

- witness_node combines several functions:
  - Apply incoming blocks
  - Apply incoming transactions for validation
  - Generate blocks (witness only)
  - Respond to API requests
- All of these interfere with each other!

Idea: Separate Processes, pt 2

- Use linux CoW memory pages
- Main process applies incoming blocks, then forks:
  - 1 process for handling API requests:
    read only, massively parallel, guaranteed read-consistency
  - 1 process for validating incoming tx
  - 1 process for generating blocks (witness only)
- **Speculative!!**