Scalability
pgday.it 2015

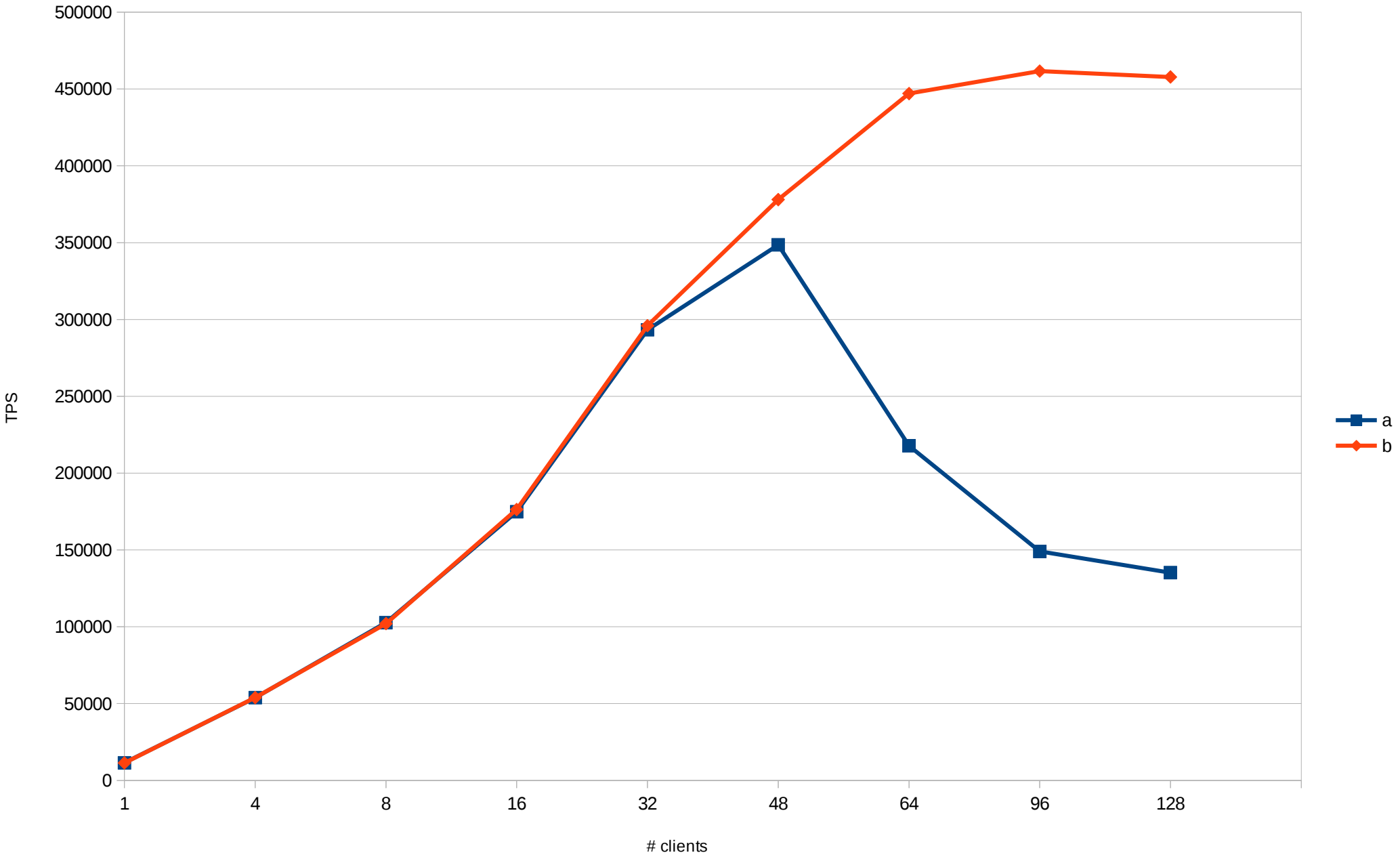
Andres Freund
PostgreSQL Developer & Committer
Citus Data – citusdata.com - @citusdata

citusdata

Scalability

“Scalability is the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged in order to accommodate that growth”

Wikipedia


500000
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
TPS
1
4
8
16
32
48
64
96
128
clients
a
b

citusdata

Vertical

Lenovo x3950-x6

citusdata

Horizontal

HP DL60 * 5

citusdata

Vertically Scalable Systems

- Easier to use
- Easier to maintain
- Stronger Consistency
  - Nearly all horizontally scalable systems are in some form *eventually consistent*
  - some problems are very hard to solve with lower consistency models
- Often faster than horizontally scalable system
- But there's definitely an upper limit

When not to scale vertically

- cost(horizontal) < cost(vertical)
- bigger hardware, even bigger hardware cost
- latency across the world is a critical issue
- current/expected scale bigger than vertically achievable

When to scale horizontally

- Little/No shared state
  - webservers
  - cache servers
  - computations
- Shared state changes infrequently
- Consistency is not paramount
- Global latency is an issue

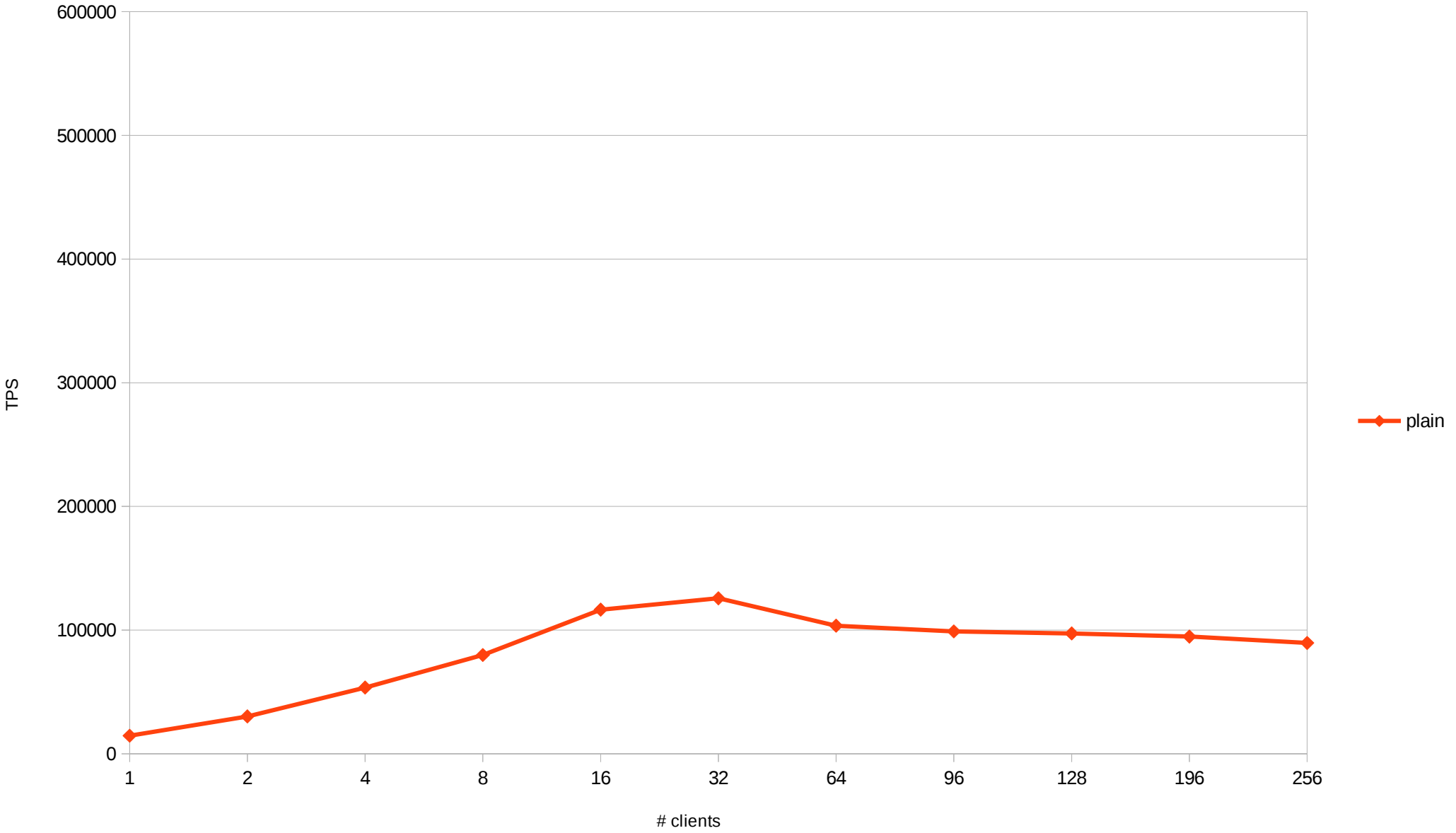
Mix & Mash

- Web-Servers: Horizontally
- Caching Infrastructure: Horizontally
- Critical Data: Vertical
- Bulk Data: Vertical if possible, horizontal otherwise

WHY ARE YOU TELLING ME THIS?!

PostgreSQL and Vertical Scalability

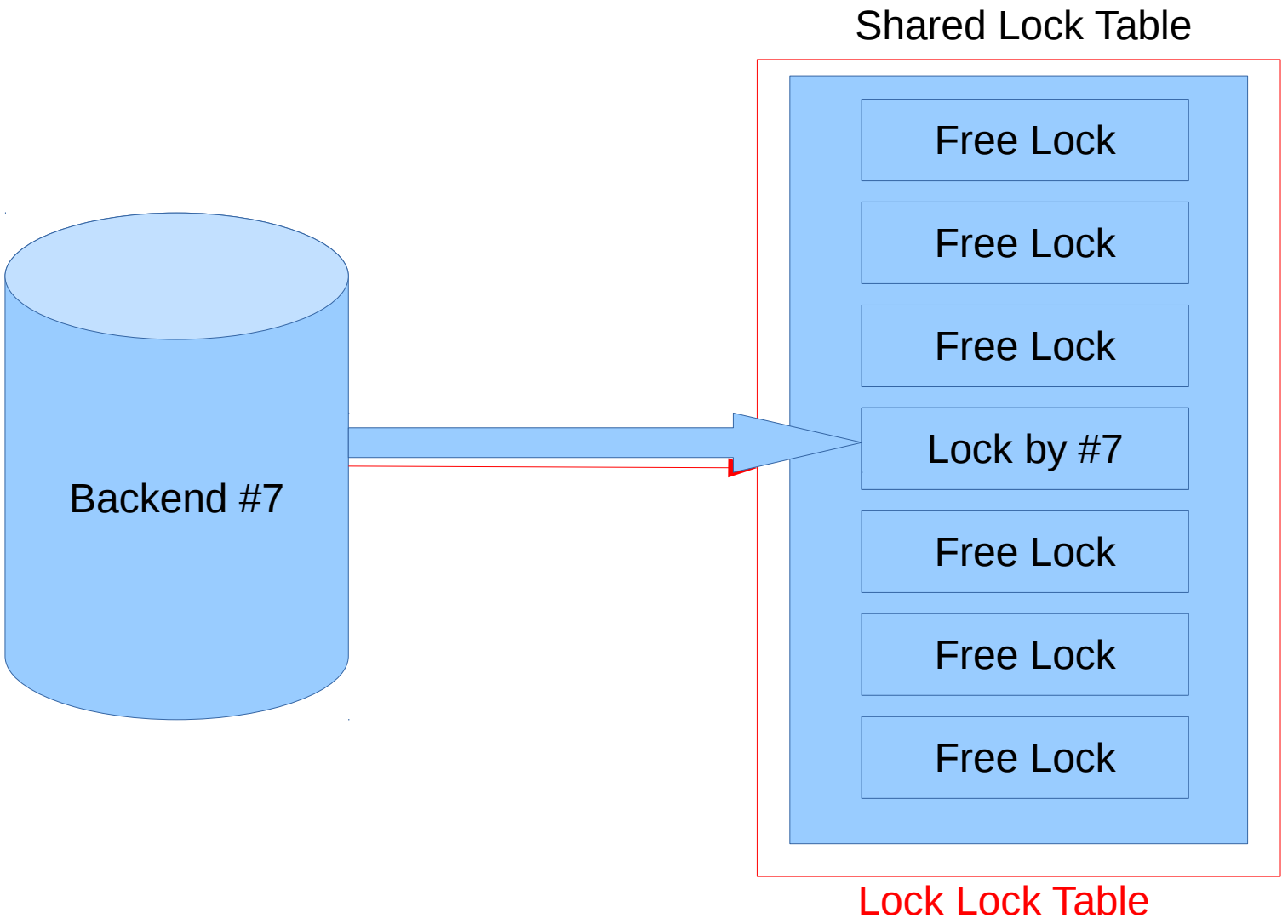
- Used to be very good – ca. 2003
- Important fixes have been made since 2009
  - Locking tables scales very good (9.2)
  - Low Level Locks scale better (9.5)
  - Cache Management scales a bit better (9.5)
  - Parallel Short Read/Write Xacts scale better (9.6)
- Very good for many concurrent workloads
- Several important problems remain



- readonly pgbench scale 300
- EC2 m4.8xlarge - 2 x E5-2676 (2 x 10 cores/20 threads)
- master @ aa6b2e6
- fastpath disabled in code

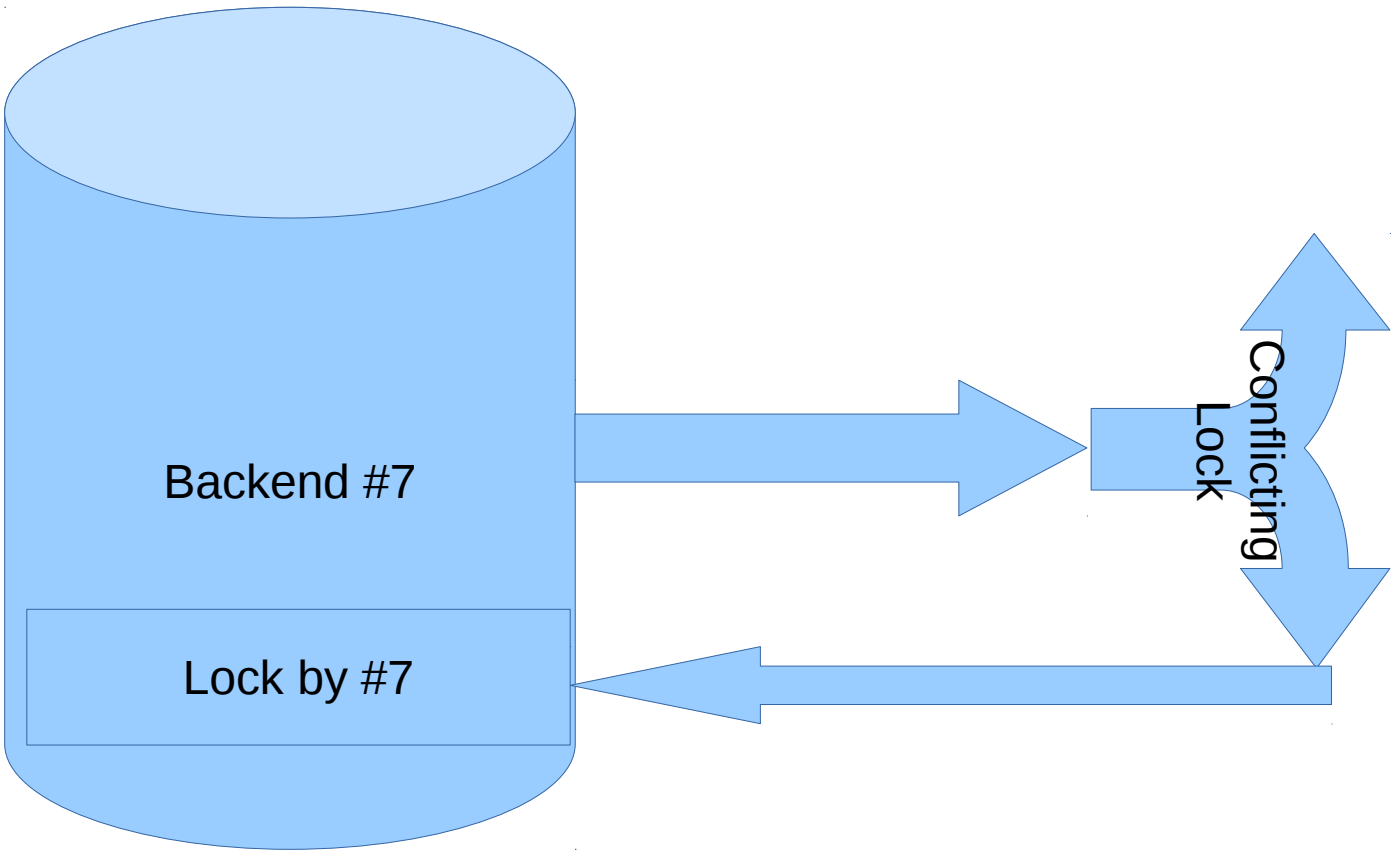
citusdata

Acquiring a Heavyweight Lock

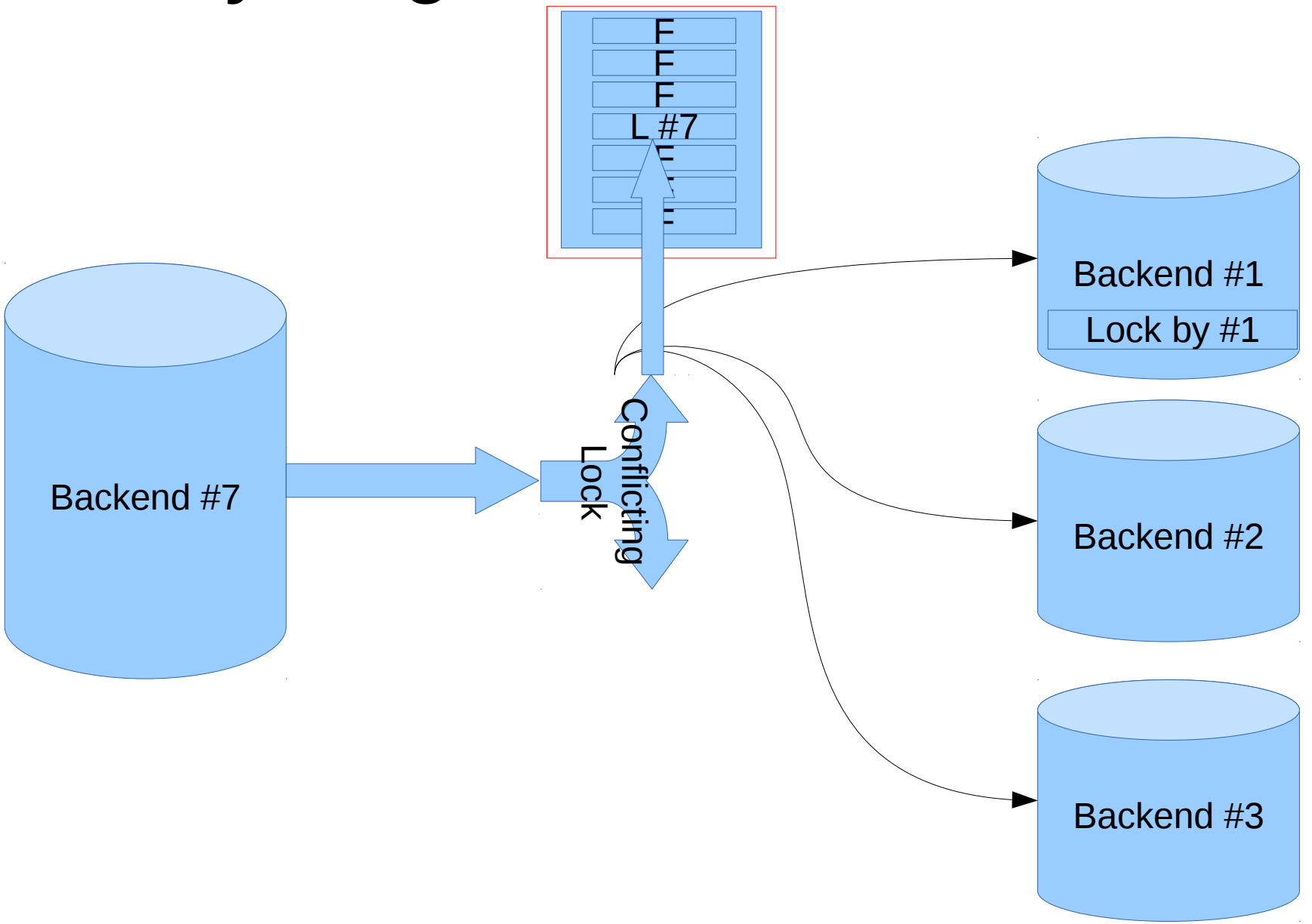


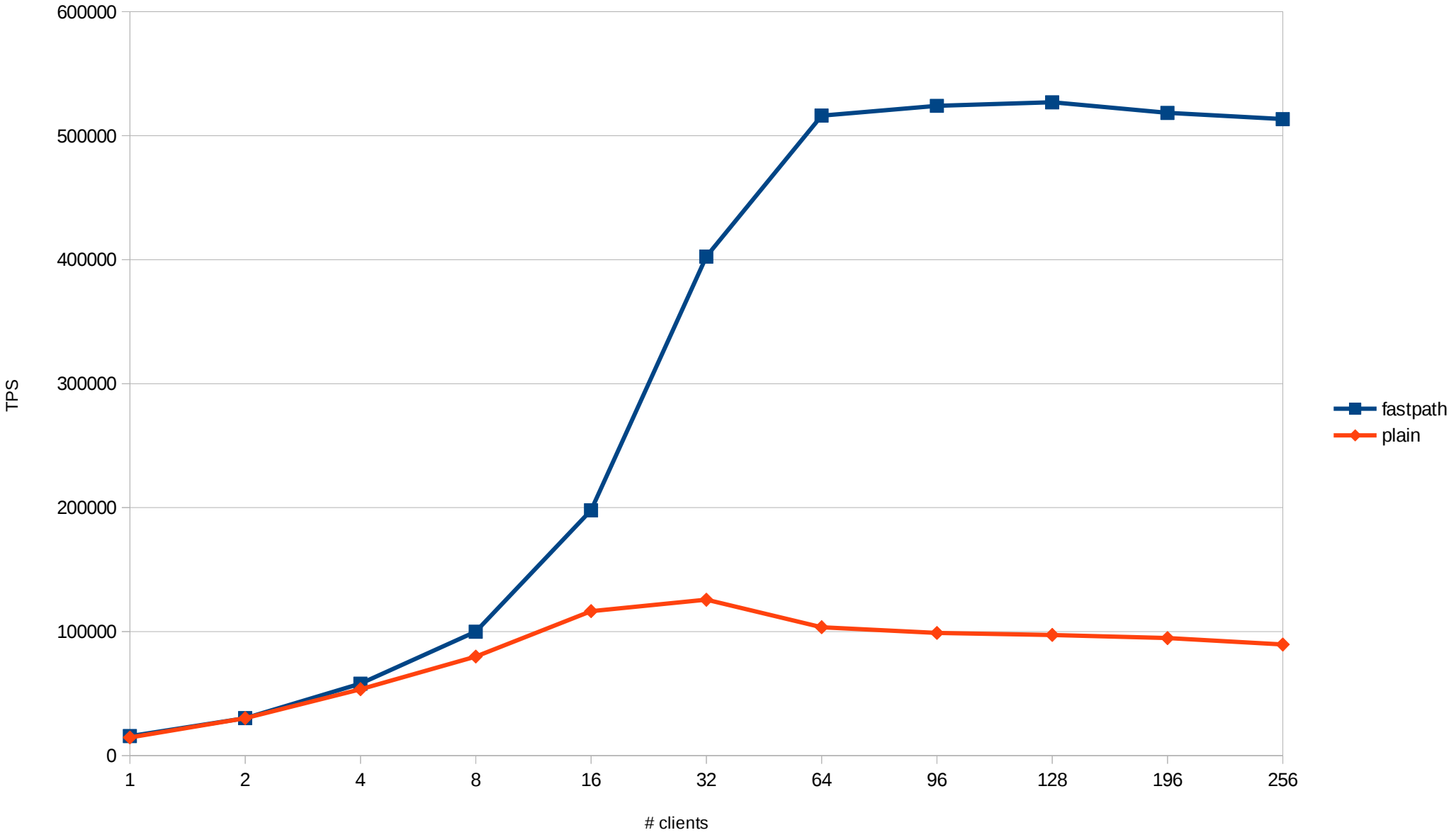
citusdata

Heavyweight Lock - Fastpath



Heavyweight Lock – Slow Path



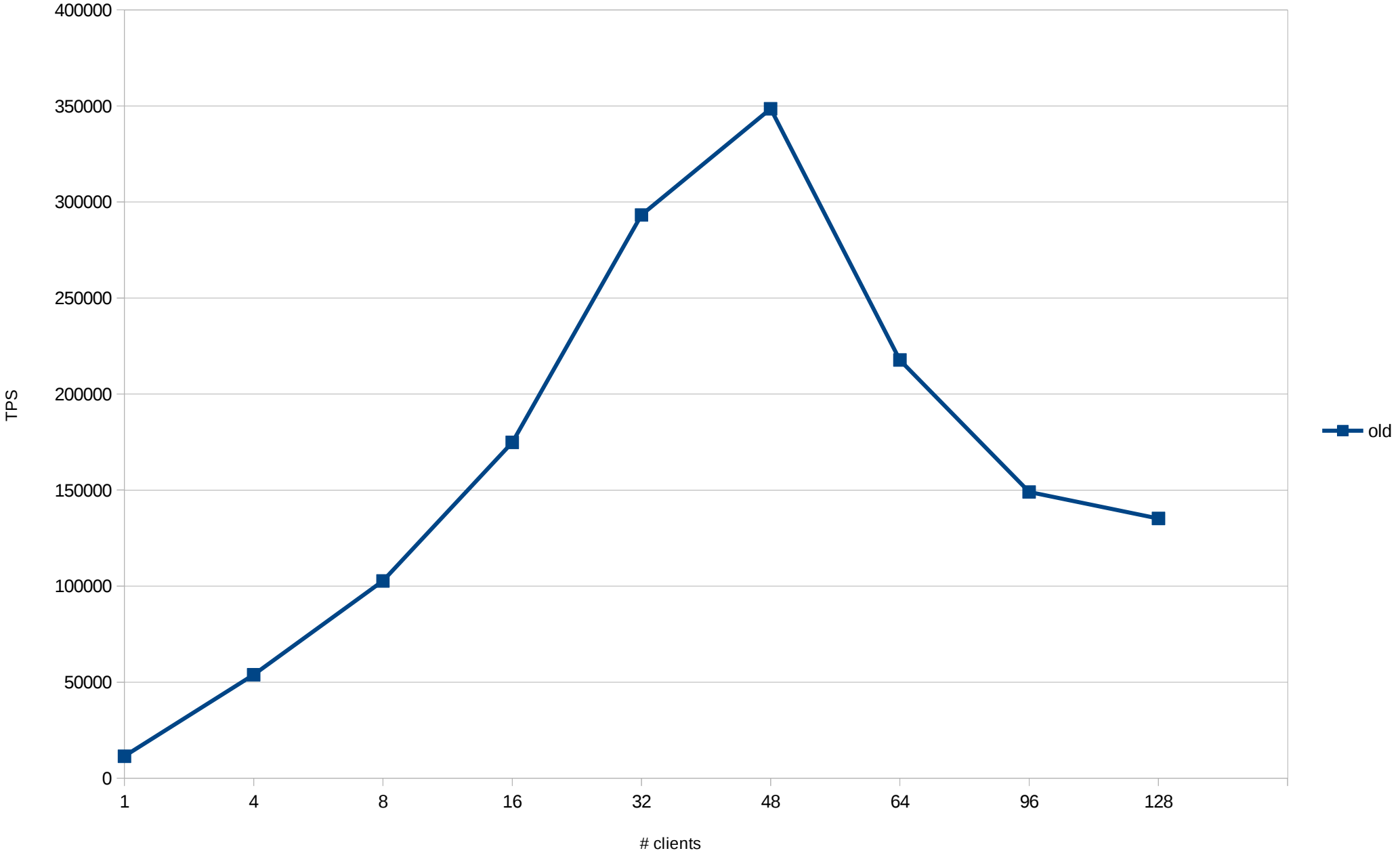


- readonly pgbench scale 300
- EC2 m4.8xlarge - 2 x E5-2676 (2 x 10 cores/20 threads)
- master @ aa6b2e6
- fastpath disabled in code

citusdata

LWLock scalability

```
# perf top -az
     89.53%  postgres  postgres  [.] s_lock
      2.53%  postgres  postgres  [.] LWLockAcquire
      1.79%  postgres  postgres  [.] LWLockRelease
      0.63%  postgres  postgres  [.] hash_search_..._value
```

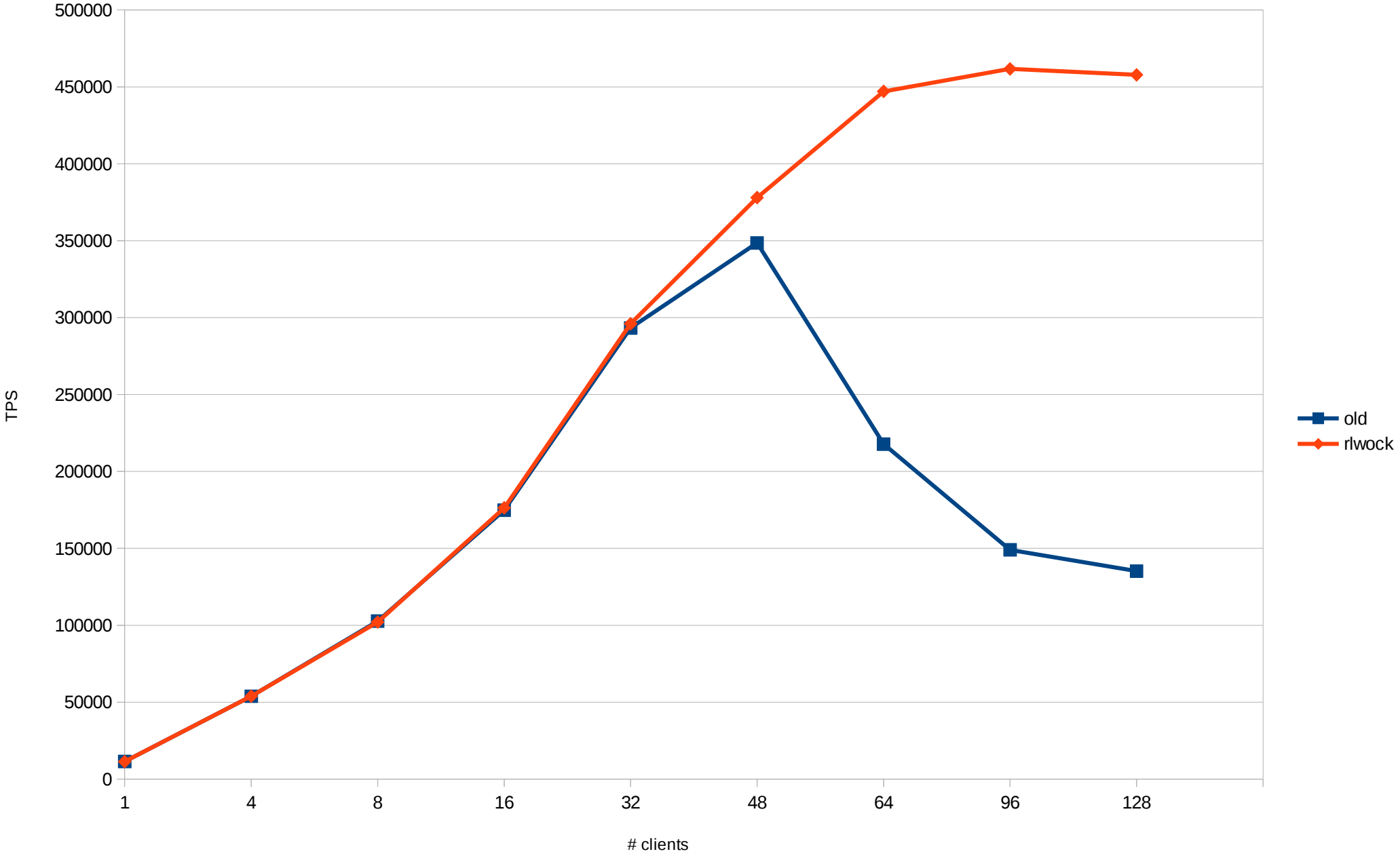


- readonly pgbench
- 4xE5-4620
- scale 100

citusdata

```
LWLockAcquire(LWLock *l, LWLockMode mode)
{
  retry:
    SpinLockAcquire(&lock->mutex);

    if (mode == LW_SHARED)
    {
        if (!lock->exclusive)
        {
            lock->shared++;
        }
        else
        {
            QueueSelf(l);
            SpinLockRelease(&lock->mutex);
            WaitForRelease(l);
            goto retry;
        }
    }
    ...
    SpinLockRelease(&lock->mutex);
}
```



- readonly pgbench
- 4xE5-4620
- scale 100

citusdata

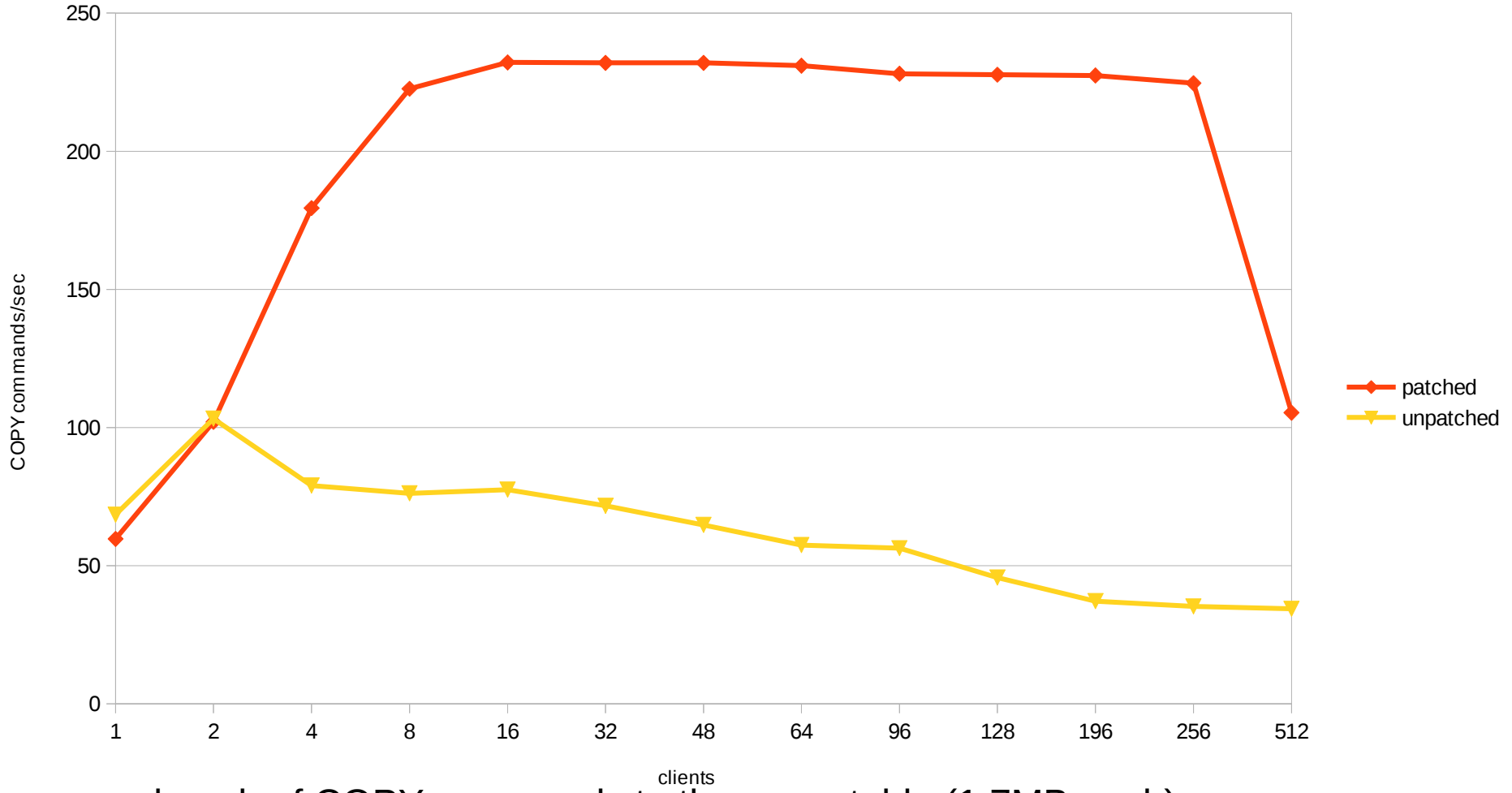
Not Fixed – Query Parallelism

- Each query only use one core
  - fine for transactional workloads
  - horrible for analytics workloads
- Initial parallelism Infrastructure in 9.5 u. 9.6
- First parallel queries hopefully in 9.6
  - will take a while to work for many types of query constructs

Further Scalability Issues

- Expensive Snapshot Computation
  - Problematic: High QPS (combined read & write) workloads, many clients
  - Solution: connection pooler
- Extension Lock
  - Problematic: Parallel bulk write workloads to single table
  - Workaround: Uh.
  - Fix hopefully in 9.6
- Buffer Replacement Complexity & Accuracy
  - Problematic: Larger than memory workloads
  - Solution: Try higher or lower shared_buffers

Extension Lock Scalability



- pgbench of COPY commands to the same table (1.7MB each)
- 4xE5-4620 (32 cores, 64 threads)
- 48 GB shared memory, 256 GB in total

citusdata

Horizontal Scalability & Postgres

- Manually shard
- Slony & Londiste (uh, forever)
- Streaming Replication / Hot Standby (9.0)
  - scale reads
- Logical Decoding (9.4)
  - coordinate systems
  - basis for logical replication solutions
- BDR & UDR (9.4)
- Foreign Data Wrappers (9.1, 9.5)
- postgres-xc / postgres-xl
- pg_shard

Scaling Analytics Workloads

- Commercial Forks of Postgres:
  - Redshift
  - Greenplum
  - CitusDB
  - …

Help!

- Contribute Problems
  - detailed descriptions of things being to slow
  - detailed descriptions of things you'd like to do
- Contribute Solutions
  - fix things that are too slow
- Contribute Contributions
  - help others to contribute