Creating a Benchmark Infrastructure ***That Just Works***

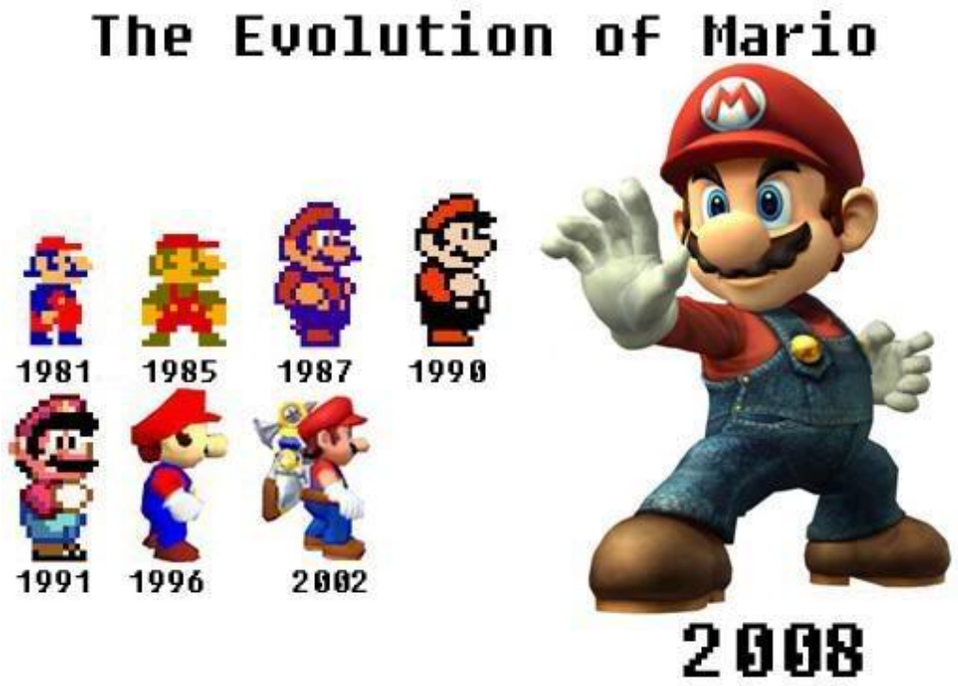
Tim Callaghan
VP/Engineering, Tokutek
email: tim@tokutek.com
twitter: @tmcallaghan

April 24, 2013

Tokutek®

What is this all about?

- Benchmark “infrastructure” is never done.
- It is a constantly evolving journey.
  - 20+ years for me!
- Hopefully you’ll learn a few new things today.
- Raise your hand.
- Share what you learn with others.



Tokutek®

Who am I?

- I’m not Mark.
- He’s that guy at Facebook.
- Sorry if you expected Mark.

Mark

Me

Tokutek

Who am I?

"Mark Callaghan's lesser-known but nonetheless smart brother."

[C. Monash, May 2010]

www.dbms2.com/2010/05/25/voltdb-finally-launches

Tokutek®

Who am I? Seriously.

My Background

- Internal development (Oracle), 92-99
- SaaS development (Oracle), 99-09
- Field engineering (VoltDB), 09-11
- VP Engineering (Tokutek), 11-present

Meaning

- development, administration, management, infrastructure, testing, product management, support

Tokutek®

What is benchmarking?

What is benchmarking?

Wikipedia

"Benchmarking is used to measure performance using a specific indicator ([transactions per second, transactional latency, etc]) resulting in a metric of performance that is then compared to others."

My Interpretation

"Run something while taking a lot of measurements, make changes, rerun and compare."

Tokutek®

Generic Benchmarking Process

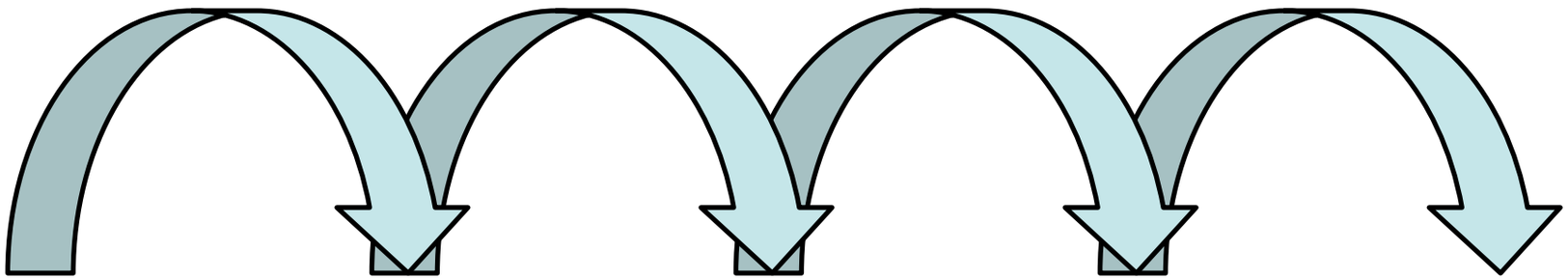
- Prepare for benchmark
  - Hardware, operating system, software
- Run Benchmark
  - Record data as you go
- Save Results
  - Flat files, database tables
- Cleanup
  - Leave things as you found them
- Analyze results
  - Look for positive and negative differences
- Change something
  - 1 thing at a time if possible
- Repeat
  - A lot

Tokutek®

My first benchmark

Commodore-64 (1985)

- Programming graphics demos
- Needed to make sprites “bounce”



- Timed round-trips using my stopwatch
- Using COS() worked, but was slow
- Found that a lookup table was much faster
  - COS(0)=1, COS(1)=.9998, …
- A benchmarker was born!

Tokutek®

Now I’m an Addict (28 years later)

- TokuDB v5.2 – 01/2012
  - End of release (code-freeze)
  - 3 benchmarks
  - 1 platform (Centos5), Disk only
  - By hand: start mysql, run benchmark, graph results
- TokuDB v6.6 – 12/2012
  - Each major development effort
  - 9 benchmarks
  - 2 platforms (Centos5 and Centos6), Disk and SSD
  - Automated: single script produces full comparison
- Soon
  - Nightly
  - 9+ benchmarks
    - Linkbench and many other new ideas

Tokutek®

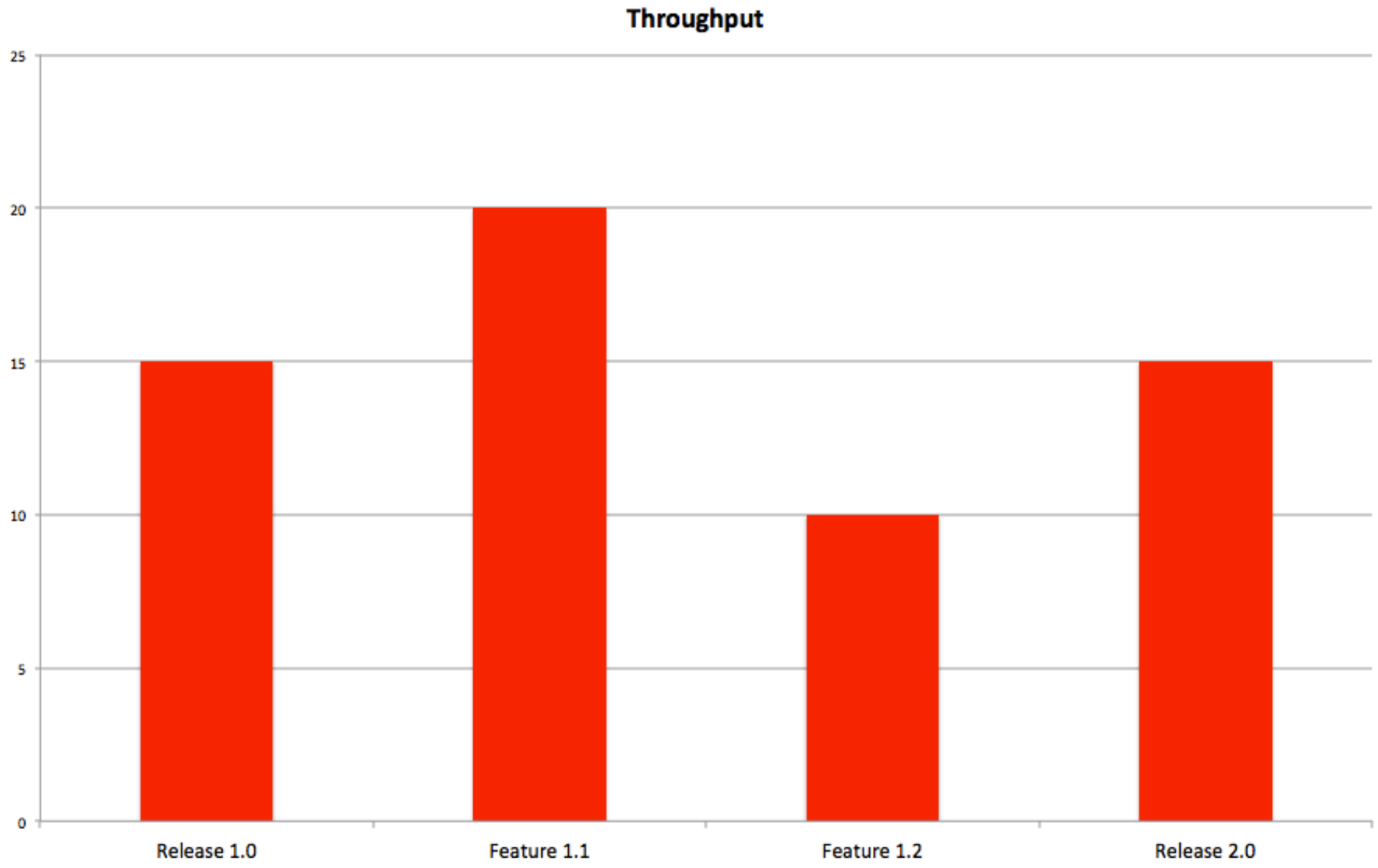
Important benchmarking skills

- Infrastructure
  - Hardware
  - Operating Systems
- Software Development
  - Benchmark applications are applications
  - Bash scripting
- Communications
  - Displaying results, finding trends
  - Blogging
- You are only as good as your worst skill
  - They all matter

Tokutek®

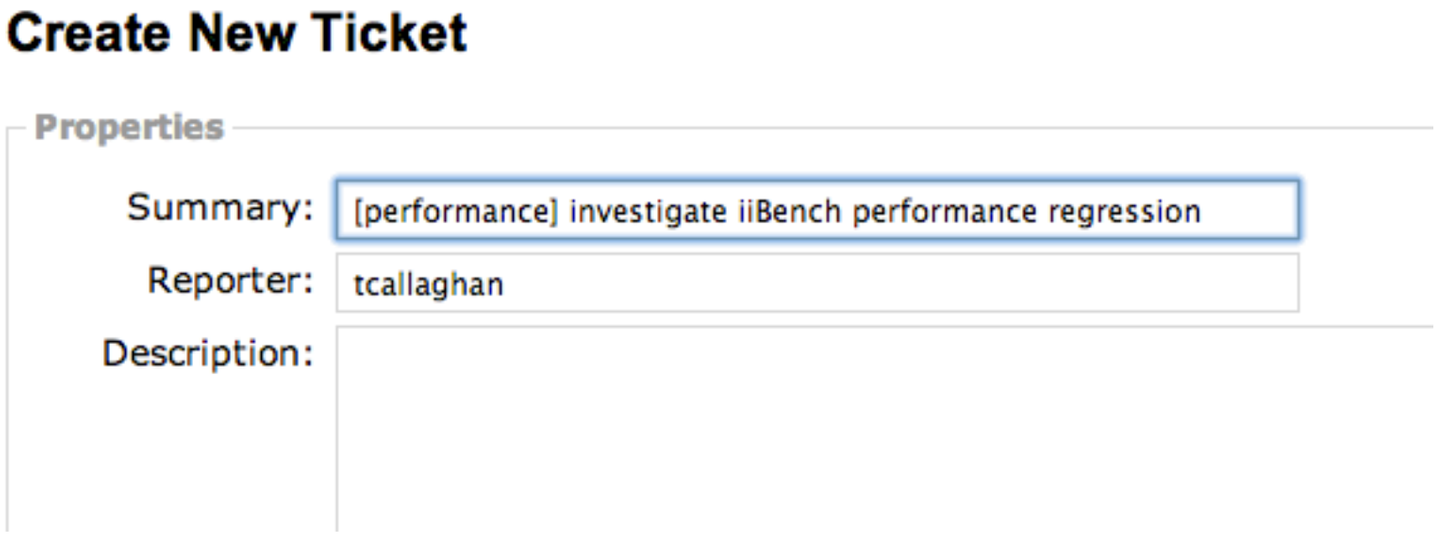
Why is frequent benchmarking important?

- One improvement can mask one drop



Why is frequent benchmarking important?

- Find issues while there is still time to deal with them
  - Avoid creating a “fix last release’s performance regression” ticket



Tokutek®

Why is frequent benchmarking important?

- Developers <3 data
  - Satisfaction of immediate feedback for efforts
  - (or) find out things are worse

```
pileup : main.54235.quicklz.5mm
POINT.PRIMARY
threads/avg/exit : 1 / 5897.0 / 5897.0   (was 6681.9, -11.75%)  -----------
threads/avg/exit : 2 / 15422.3 / 15422.3   (was 14502.3, 6.34%)  ++++++
threads/avg/exit : 4 / 27714.7 / 27714.7   (was 27177.6, 1.98%)  +
threads/avg/exit : 8 / 50294.2 / 50294.2   (was 47400.8, 6.10%)  ++++++
threads/avg/exit : 16 / 96995.3 / 96995.3   (was 90559.1, 7.11%)  +++++++
threads/avg/exit : 32 / 97953.1 / 97953.1   (was 86235.9, 13.59%)  ++++++++++++
threads/avg/exit : 64 / 97397.7 / 97397.7   (was 68373.9, 42.45%)  ++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 128 / 95417.1 / 95417.1   (was 48768.9, 95.65%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 256 / 92101.4 / 92101.4   (was 41325.2, 122.87%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 512 / 88538.7 / 88538.7   (was 34471.0, 156.85%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 1024 / 86304.6 / 86304.6   (was 28875.2, 198.89%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
POINT.SECONDARY
threads/avg/exit : 1 / 8113.5 / 8113.5   (was 8044.3, 0.86%)
threads/avg/exit : 2 / 13442.6 / 13442.6   (was 13648.0, -1.50%)  -
threads/avg/exit : 4 / 24612.7 / 24612.7   (was 24247.2, 1.51%)  +
threads/avg/exit : 8 / 41736.5 / 41736.5   (was 41074.3, 1.61%)  +
threads/avg/exit : 16 / 79966.5 / 79966.5   (was 73071.6, 9.44%)  +++++++++
threads/avg/exit : 32 / 80184.7 / 80184.7   (was 72836.8, 10.09%)  ++++++++++
threads/avg/exit : 64 / 79523.2 / 79523.2   (was 68818.2, 15.56%)  ++++++++++++++++
threads/avg/exit : 128 / 78401.3 / 78401.3   (was 52484.3, 49.38%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 256 / 76585.8 / 76585.8   (was 39196.3, 95.39%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 512 / 74582.3 / 74582.3   (was 33275.8, 124.13%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
threads/avg/exit : 1024 / 73170.2 / 73170.2   (was 28070.2, 160.67%)  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
RANGE.PRIMARY
threads/avg/exit : 1 / 1076.4 / 1076.4   (was 1012.0, 6.36%)  ++++++
threads/avg/exit : 2 / 1733.3 / 1733.3   (was 2134.5, -18.80%)  ------------------
threads/avg/exit : 4 / 3498.7 / 3498.7   (was 3548.6, -1.41%)  -
threads/avg/exit : 8 / 6941.2 / 6941.2   (was 6999.7, -0.84%)
threads/avg/exit : 16 / 8233.4 / 8233.4   (was 7761.5, 6.08%)  ++++++
threads/avg/exit : 32 / 8037.7 / 8037.7   (was 7587.8, 5.93%)  +++++
threads/avg/exit : 64 / 8149.6 / 8149.6   (was 7639.2, 6.68%)  ++++++
threads/avg/exit : 128 / 8136.7 / 8136.7   (was 7624.7, 6.72%)  ++++++
threads/avg/exit : 256 / 8248.2 / 8248.2   (was 7703.0, 7.08%)  +++++++
threads/avg/exit : 512 / 8270.4 / 8270.4   (was 7721.4, 7.11%)  +++++++
threads/avg/exit : 1024 / 8269.7 / 8269.7   (was 7676.9, 7.72%)  +++++++
```

Tokutek®

Level : Beginner

Tokutek®

Basic configuration

- Start with a “known good” environment
  - Learn from what others have shared about server and MySQL configuration, many online resources exist
  - Vadim Tkachenko @ Percona, http://www.mysqlperformanceblog.com/author/vadim/
  - Dimitri Kravtchuk @ Oracle, http://dimitrik.free.fr/blog/
  - Too many to list them all.
  - I always create an extra file system where benchmarking occurs
    - Treat it as if it will not be there tomorrow
    - Enables rebuilding/changing file systems without losing important work

Benchmarking your equipment

- Quick tests to ensure configuration is correct
- If possible, compare to existing equipment
- Sysbench CPU
  - sysbench --test=cpu --cpu-max-prime=20000 run
- Sysbench memory
  - sysbench --test=memory --memory-block-size=1K --memory-scope=global --memory-total-size=100G --memory-oper=read run
  - sysbench --test=memory --memory-block-size=1K --memory-scope=global --memory-total-size=100G --memory-oper=write run
- Sysbench IO
  - sysbench --test=fileio --file-total-size=10G --file-test-mode=rndrw --file-num=1 prepare
  - sysbench --test=fileio --file-total-size=10G --file-test-mode=rndrw --file-extra-flags=direct --file-num=1 --max-requests=50000 run
  - sysbench --test=fileio --file-total-size=10G --file-test-mode=rndrw --file-num=1 cleanup
- Sysbench OLTP
  - sysbench –test=oltp -oltp_tables_count=16 --oltp-table-size=500000 prepare
  - sysbench –test=oltp -oltp_tables_count=16 --oltp-table-size=500000 prepare

Tokutek®

What should I measure?

- Start with throughput and latency
  - Throughput = the number of operations per specific time interval, often referred to as transactions per second (TPS)
  - Latency = the amount of time required to satisfy a single transaction
- Use Intervals
  - Benchmarks generally run for several minutes or hours, throughput and latency should be captured in smaller windows (minutes or seconds).
  - Interval data provides the information needed to compute cumulative averages, min, max…

Where do you go with questions?

- Ask the benchmark owner
  - sysbench = Percona (launchpad)
  - Iibench = Mark Callaghan (launchpad)
  - tpc-c = Percona (launchchpad), *many other implementations exist*
- www.stackoverflow.com / www.serverfault.com

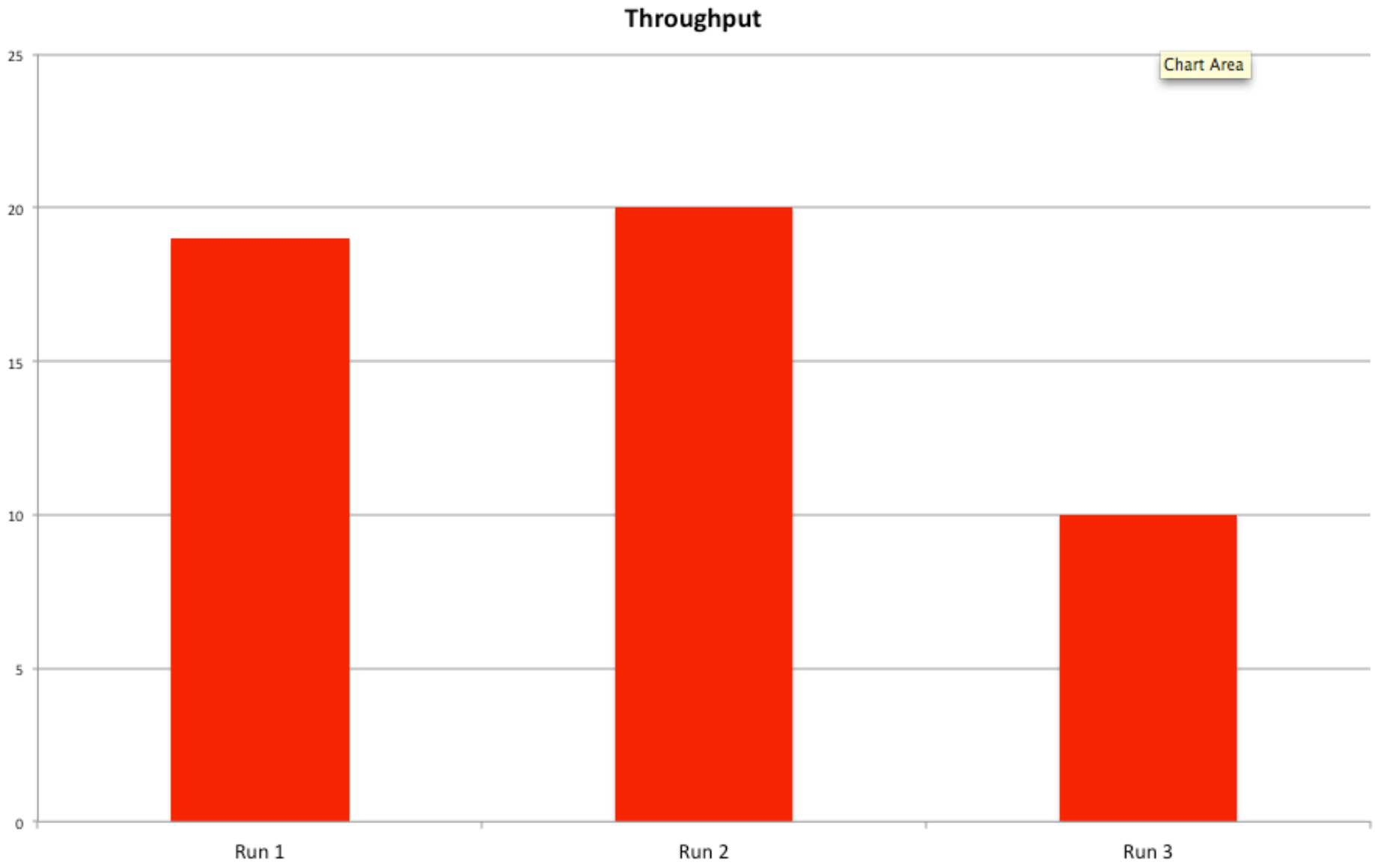
The power of 2 (but buy 3 if you can)

- Historical results are critical
  - *(but sometimes you just need to start over)*
    - *Equipment dies or becomes irrelevant*

Tokutek®

The power of 3 (for a while)

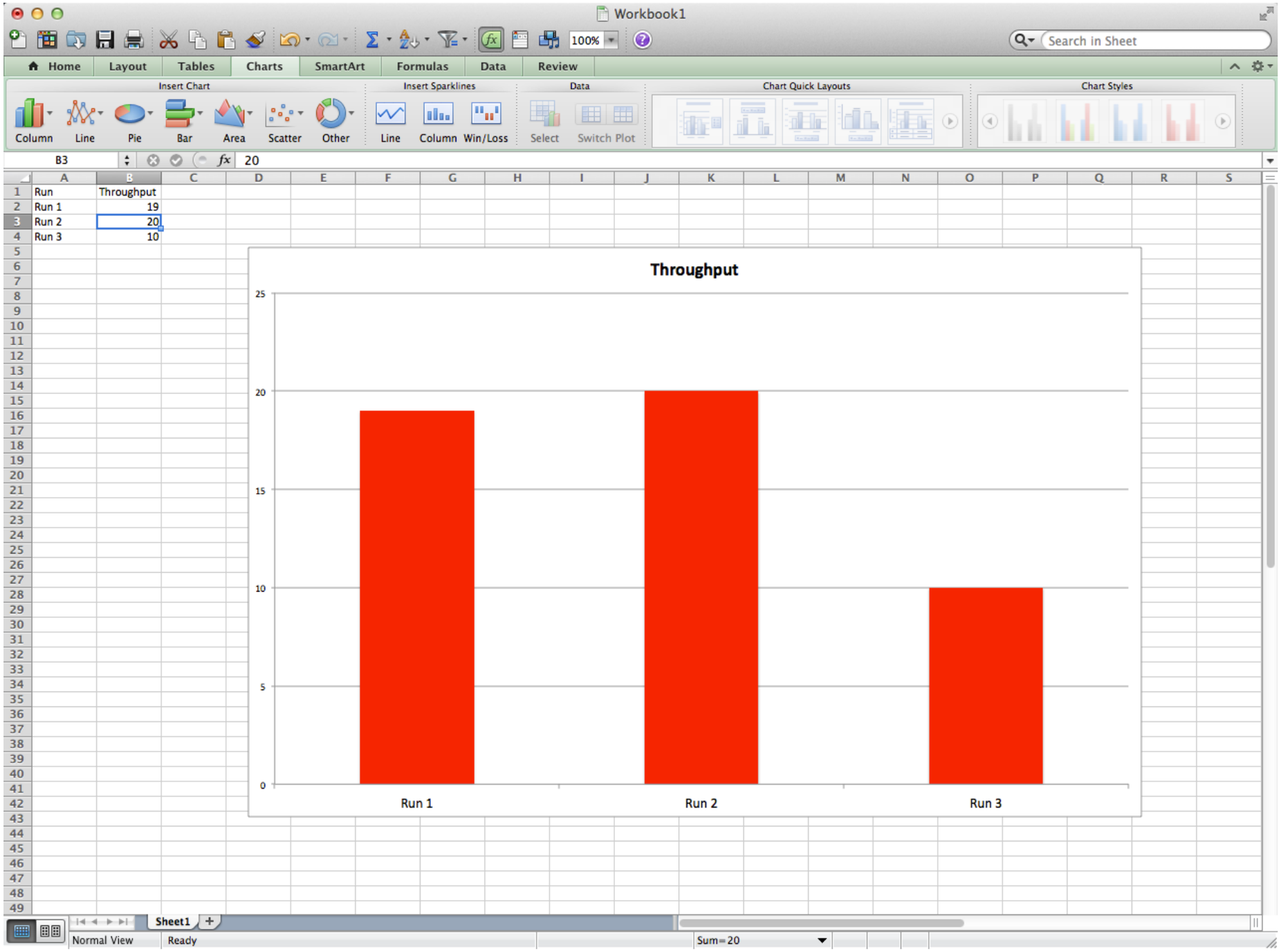
- Run 3 times until you regularly produce consistent results



Tokutek®

Visualization, for now

- Use Excel or LibreOffice



Tokutek®

Hacks

- Low tech solutions are fine
  - Franken-switch – partition detection benchmarking

Vs.

Find hardware
Buy it
Install it
Configure it
Run benchmarks

Tokutek

Ready-to-go MySQL

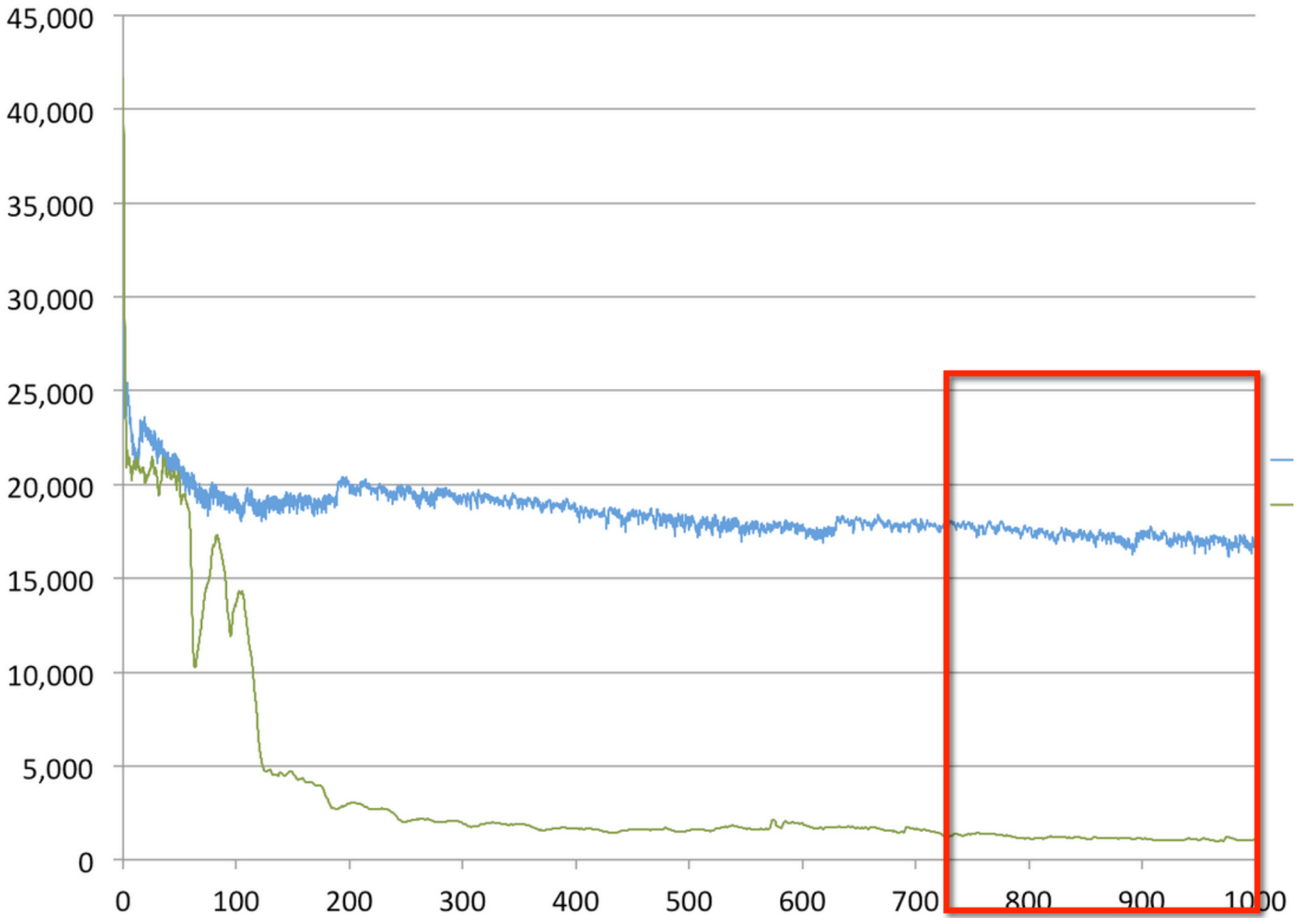
- Evolutionary process – I've probably done each > 1000 times
  - Phase 1 : before each benchmark manually create MySQL and start
    - untar MySQL
    - copy my.cnf from known good directory
    - bin/mysql_install_db
    - bin/mysqld_safe
  - Phase 2 : for each new build, unpack and repack MySQL (ready-to-go)
    - in empty folder : copy my.cnf, fresh-db, and MySQL tar
    - ./fresh-db (untars MySQL, copies my.cnf, runs install, starts/stops mysql)
    - rm –rf mysql-test sql-bench
    - tar xzvf $ARCHIVE_DIR/blank-toku610-mysql-5.5.28.tar.gz .
  - Phase 3 : for each new build, unpack and repack MySQL (ready-to-go)
    - in empty folder : copy MySQL tar
    - gf blank-toku610-mysql-5.5.28.tar.gz (bash script that does everything else)
  - Phase 4 : at some point
    - in empty folder : copy MySQL tar
    - gf2 (bash script that does everything else, including detecting version of TokuDB and MySQL/MariaDB)

Useful technologies

- rsync
  - Synchronize your files with benchmark servers
- nfs
  - (If you can) make all of your files available to your benchmark servers without needing to manually rsync
- dropbox
  - Easy way to synchronize your files with EC2 benchmark servers
  - (thanks for Amrith Kumar @ ParElastic for this tip)

Exit throughput?

- Allow your benchmark to stabilize
- Average final <x> transactions or <y> minutes



Trust no one

- If possible, dedicated benchmarking equipment
- Prior to starting, check for clean environment (users, processes)
  - users
  - ps aux | grep mysqld
  - ls /tmp/*.sock
- Periodically check for the duration of benchmark run
- Leave it as you found it
  - Shutdown MySQL
  - Kill any stray monitoring scripts
  - Delete all MySQL files

Tokutek®

Level : Intermediate

Tokutek®

Environment variables

- I find it convenient for my benchmarking scripts to pass information using environment variables
  - Command line arguments can be used as well
- Create a commonly named set of environment variables for all machines.
- Load it on login (add to .bashrc)
  - source ~/machine.config

```
export machine_name=mork
export tokudb_cache=24G
export innodb_cache=24G
export storage_type=disk
export benchmark_directory=/data/benchmark
```

Scripting 1

- Lets do some bash, simple Sysbench runner

```
#! /bin/bash

num_tables=16

sysbench –test=oltp num_tables=${num_tables} > results.txt

tar ${benchmark_results_dir}/${machine_name}.tar results.txt
```


Tokutek®

Scripting 2

- Make sure an environment variable is defined

```
#! /bin/bash

if [ -z "$machine_name" ]; then
  echo "Need to set machine_name"
  exit 1
fi

num_tables=16

sysbench –test=oltp num_tables=${num_tables} > results.txt
tar ${benchmark_results_dir}/${machine_name}.tar results.txt
```

Scripting 3

- Make sure a directory exists

```
#! /bin/bash

if [ ! -d "$benchmark_results_dir" ]; then
  echo "Need to create directory $benchmark_results_dir"
  exit 1
fi

num_tables=16

sysbench –test=oltp num_tables=${num_tables} > results.txt
tar ${benchmark_results_dir}/${machine_name}.tar results.txt
```


Tokutek

Scripting 4

- Make sure a directory is empty

```
#! /bin/bash

if [ "$(ls -A $benchmark_results_dir)" ]; then
  echo "$benchmark_results_dir has files, cannot run script"
  exit 1
fi

num_tables=16

sysbench –test=oltp num_tables=${num_tables} > results.txt
tar ${benchmark_results_dir}/${machine_name}.tar results.txt
```


Tokutek®

Scripting 5

- Default values

```
#! /bin/bash

if [ -z "$num_tables" ]; then
    export num_tables=16
fi

sysbench –test=oltp num_tables=${num_tables} > results.txt
tar ${benchmark_results_dir}/${machine_name}.tar results.txt
```


Tokutek®

Scripting 6 – scripts calling scripts

controller.bash

```
#! /bin/bash
if [ -z "$mysql" ]; then
  export mysql=blank-toku701-mysql-5.5.30
fi
if [ -z "$num_tables" ]; then
  export num_tables=16
fi
for numThreads in 1 2 4 8 16 32 64; do
  export clientCount=${numThreads}
  ./runner.bash
done
```

runner.bash

```
#! /bin/bash
if [ -z "$mysql" ]; then
  echo "Need to set mysql”; exit 1
fi
if [ -z "$num_tables" ]; then
  export num_tables=16
fi
if [ -z ”$clientCount" ]; then
  export clientCount=64
fi
mkdb ${mysql}    #unpack and start ready-to-go-mysql
sysbench –test=oltp num_tables=${num_tables} –num_threads=${clientCount} > results.txt
tar ${benchmark_results_dir}/${machine_name}-${clientCount}.tar results.txt
mstop            #shutdown mysql and cleanup
```

Tokutek®

Scripting 7 – scripts calling scripts calling scripts

multibench.bash

```
#! /bin/bash
for mysqlType in mysql mariadb ; do
  export mysql=blank-toku701-${mysqlType}-5.5.30
  sysbench-directory/controller.bash
  iibench-directory/controller.bash
  for numWarehouses in 100 1000 ; do
    export num_warehouses=${numWarehouses}
    tpcc-directory/controller.bash
  done
done
```

Tokutek

Add more measurements

- IO (iostat)
- CPU/Memory (ps)
- MySQL engine status
  - "show engine <storage-engine-name> status;"
- MySQL global status
  - "select * from information_schema.global_status;"

- You are only measuring too much when it affects the benchmark
- You never know when it will be helpful
- De-dupe
  - Many counters are the same at each interval, save space by only capturing values that changed since the last measurement

Tokutek®

Optimize everything not benchmarked

- Spend as little time as possible on surrounding steps
- Example : load Sysbench data, 16 tables, 50 million rows per table
  - trickle load
    - client application generating insert statements
    - 5+ hours to load
  - bulk load
    - after trickle loading, mysqldump each table to TSV
    - "load data infile …"
    - 70 minutes to load
  - preloaded database
    - save MySQL data folder after bulk loading
    - "cp –r" prior to running benchmark
    - 10 minutes to load

Tokutek®

Anomalies

- Graph results, review graphs regularly
  - Huge dip in the middle of the benchmark
  - Exit throughput was unaffected

Don’t automate too early

- You’ll end up automating the wrong things
- You’ll over-engineer your automation
- Never automate “until it hurts”

Put results into a database

- Create a simple schema for historical results
- benchmark_header
  - bench_id
  - name (sysbench, tpcc, iibench)
  - duration (# seconds | # rows)
  - thruput (use exit throughput)
  - attr1 (# sysbench rows | # tpc-c warehouses | # iibench rows)
  - attr2 (# sysbench tables)
  - clients (concurrency)
  - machine_name (server)
- benchmark_detail
  - bench_id
  - duration (# seconds | # rows)
  - thruput (interval)
  - latency (interval)
  - throughput_avg (cumulative)
- create script to read benchmark logs and load database
- run immediately as benchmark ends

Tokutek®

Pre-compute interesting numbers

- Pre-compute interesting numbers
- SQL to compute running average
  - extremely slow on benchmarks with 10,000+ data points
  - add calculation of running average to script that reads/loads results

```
select dtl1.duration as duration,
       dtl1.thruput as interval_tps,
       avg(dtl2.thruput) as avg_tps
from benchmark_detail dtl1,
     benchmark_detail dtl2
where dtl1.bench_id = 1 and
      dtl2.bench_id = dtl1.bench_id and
      dtl2.duration <= dtl1.duration
group by dtl1.duration, dtl1.thruput;
```


Tokutek®

Visualization - gnuplot

- I’ve been graphing benchmark results in Excel for years
- Last year someone asked me why don’t just learn gnuplot
- OMG!
- If you don't use it, start using it, today.

```
set terminal pngcairo size 800,600
set xlabel "Inserted Rows”
set ylabel "Inserts/Second”
set title "iiBench Performance - Old vs. New"
set output ”benchmark.png”
plot "./old.txt" using 1:6 with lines ti "Old",
"./new.txt" using 1:6 with lines ti "New"
```

Tokutek®

Idle machines could be benchmarking

- Obvious benchmarks
  - increase cache size
  - buffered vs. direct IO
  - RAID levels
  - File systems
  - rotating disk vs. ssd vs. flash
  - InnoDB : search the web for ideas
  - TokuDB : compression type, basement node size
- Not so obvious
  - decrease cache size
  - alternative memory allocators

Tokutek®

Level : Advanced

Tokutek®

Level : Advanced

Everything that follows is
on my wish list...

Tokutek®

Additional data capture

- Find a data capture tool that gathers more server metrics
  - collectd
- Graph and review the data
  - make it actionable
- Another gnuplot moment?

Continuous Integration

- Implement CI server (Jenkins?)
  - Nightly build
  - Distribute and execute benchmarks across group of servers
  - Graph results
  - Compare to prior day
  - Alert if "threshold" change from previous day, or over previous period

Self-service

MySQL Version

TokuDB Version

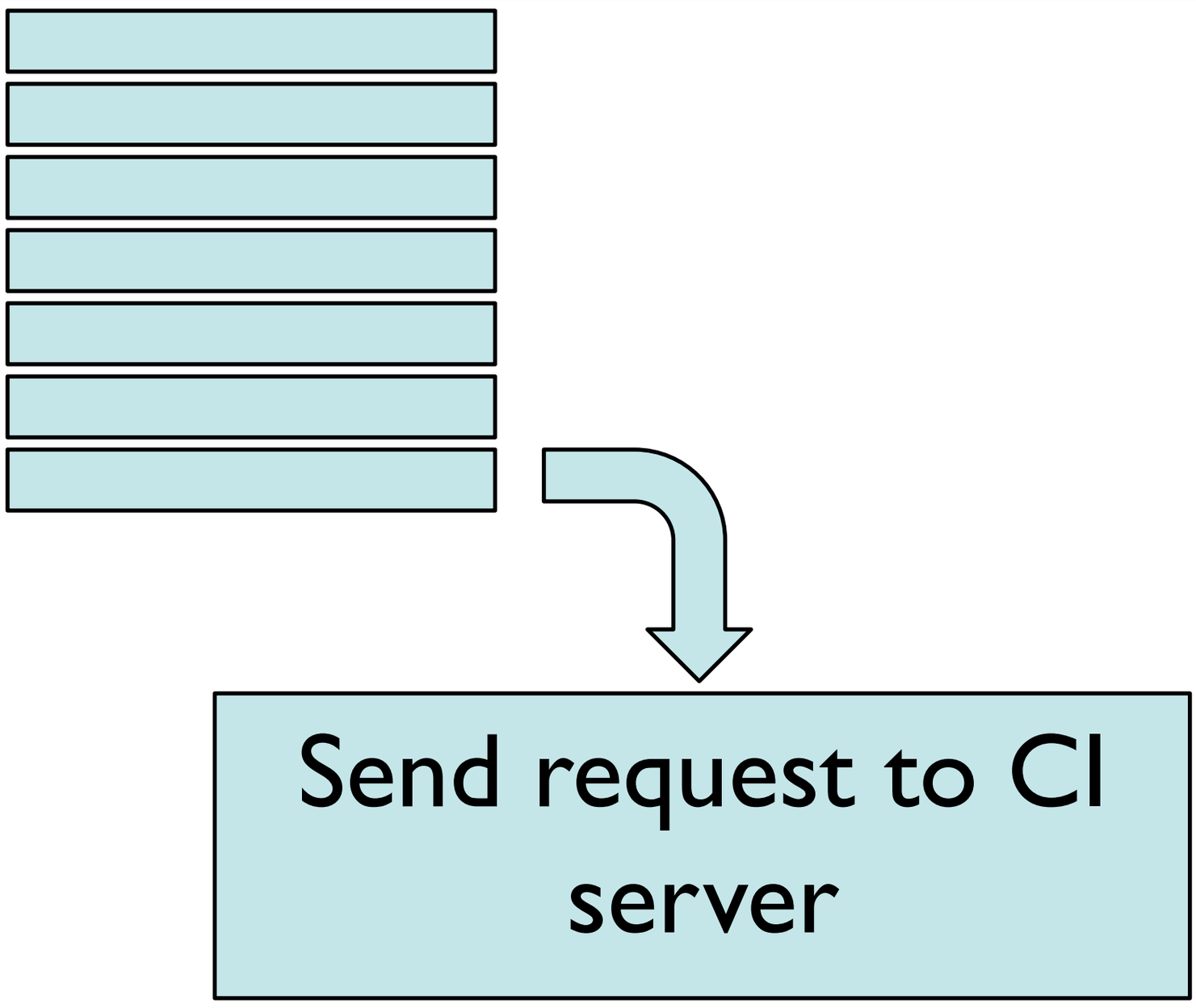
Storage Engine

Benchmark

Num Tables

Num Rows

Compare To



Tokutek

Automated anomaly detection

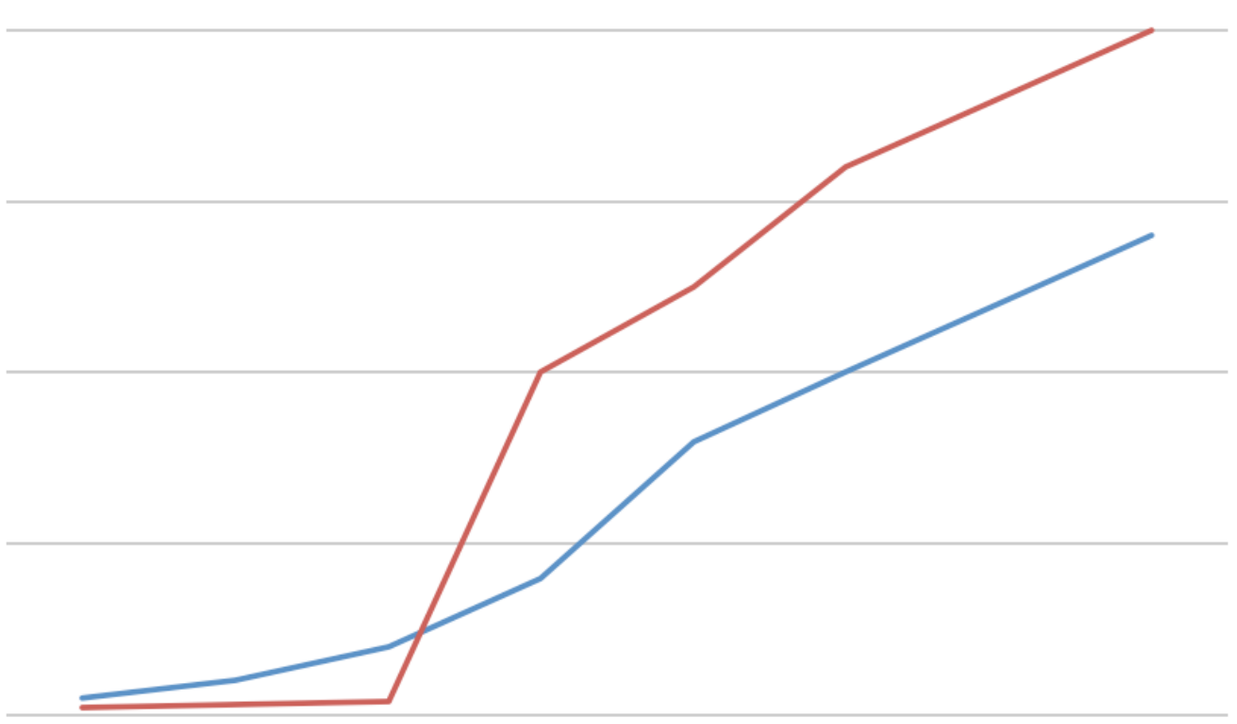
- Problem is easy to see, not always easy to detect

PSA

"Many benchmarks are like magic tricks. When you know how the results were achieved, you are no longer impressed."

twitter.com/GregRahn/status/263771033583104000

Show the whole picture



Tokutek®

Wrapping it up

Dive In.

Benchmark Everything.

Learn and Share.

Tokutek®

Please rate this session.

Feedback is important!

Tokutek®

Tokutek is hiring!

Position: QA++

Testing

Support

Benchmarking

Release Engineering

Tokutek®

Questions?

Tim Callaghan
tim@tokutek.com (email)
@tmcallaghan (twitter)

www.tokutek.com/tokuview (blog)

slides at

http://www.slideshare.net/tmcallaghan/20130424-perconalivebenchmarking

Tokutek®