PERCONA

Performance Consulting Experts

InnoDB Scalability Limits

Peter Zaitsev, Vadim Tkachenko

Percona Inc

MySQL Users Conference 2008

April 14-17, 2008

Who are the Speakers ?

- Founders of Percona Inc
  - MySQL Performance and Scaling consulting company
- Active MySQL Community Members
- Writers http://www.mysqlperformanceblog.com
- Co-Authors of High Performance MySQL Second Edition

What we'll talk about

- MySQL's main transactional storage engine InnoDB
- Looking into InnoDB Scalability Limits
  - And how to solve them (or work around them)
- Check into other aspects affecting InnoDB Performance
- Focus on existing codebase in 5.0 and 5.1
- Mark Callaghan may already have fixed some of the issues
- Provide benchmark results to support most of our claims.

Kinds of Scalability

- Scalability means different things to different people
- Upwards and Downwards in terms of hardware
  - We're not really interested in the downwards part
- We look at Scaling application first
  - And what InnoDB needs to have in order to support it
- This comes down to a number of InnoDB scaling primitives.

Scalability for your Boss

- For Product Manager scalability means Application can Perform as it Grows.
  - What does **Perform** mean ?
  - What does **Grow** mean ?
- And typically these demands come with budget constraints

What does Performs mean

- Response time is “Decent”
  - No pages take 30 seconds to load
  - Easy to measure in production
- There is enough system capacity
  - System can take the load without degrading in response time and failing
  - Often estimated (benchmarked)
- Operations can manage the system
  - You can perform backups, table maintenance etc

Application Growth Parameters

- Load (think spikes)
  - Growing from 100 q/sec to 1000
- Database Size
  - Going from 10GB to 100GB data size
- Data Distribution
  - Going from 10 friends in average to 250
- Launching new features or changes to the old one
  - Full text search feature can be heavy add on

Application Scaling Challenge

- As application grows all parameters tend to grow at once
  - And you have to deal with larger number of more complex queries on large database size.
- To maintain performance in these conditions you
  - Optimize “application”
    - Schema, Queries, Caching, Sharding, Replication
  - Get more hardware
- We focus on Scaling by Platform Upgrades in this presentation

What do we mean by Platform

- Hardware
- Operating System
- MySQL Server

Testing by Micro Benchmarks

- Micro Benchmarks correspond to some simple operations.
- Tend to stress some particular aspect of behavior
- If problem is seen in micro benchmarks it is seen in some applications
  - Though applications tend to expose wider range of scaling problems
- Micro benchmarks may take things to extreme
  - Showing issue at larger scale than application would have

Operational Issues

- Often forgotten about
- Descriptive nature and does not need to be benchmarked

Large Tables

- Large Tables (and instances are hard to deal)
- Backup – physical backup is must.
- Can't move separate tables between servers physical way
- No **REPAIR** functionality
  - Corruption often means dump and restore
  - Restore from physical backup is often faster
- **ALTER TABLE** is very slow
  - Master-Master replication can help
- Table maintenance **OPTIMIZE TABLE**

Online Index creation in InnoDB

- New plugin just announced today by Ken and Heikki
  - We had early access to the code
- InnoDB can now build indexes without rebuilding whole table
  - Index build done by sort which is much faster
- 10 times faster load and better index sizes
  - But only if you load data and add indexes separately

Load Method	Load Time	Data Size	Index Size
SQL Dump	88m	1333788672	1867513856
LOAD INFILE	90m	1333788672	1867513856
ALTER from MYISAM	90m	1333788672	1867513856
LOAD + ADD INDEX	3m+5m	1333788672	1124073472
SQL Dump MyISAM	3m	1050000000	312579072

How good are Sorted Indexes

- Indexes built by sort can be better physically sorted and have better fill factor.
- Full Index Scan speed (cold)
  - **31 sec** standard vs **22 sec** built by sorting
    - Becomes CPU bound at this stage, could be even better
- Update:
  - **update sample set c=md5(i) where i%1000=1;**
    - **3 min 20 sec** standard vs **8 min 16 sec** sorted
  - Index size growth:
    - **0%** (standard) vs **30%** (sorted)

Many tables

- You may be escaping from large tables by creating many small tables instead
  - InnoDB keeps all table it accessed open
    - Can consume a lot of memory (reduced in 5.1)
    - Lesser issue with modern 64bit platforms
  - **innodb_file_per_table**=1
    - Can get small tables to use more space
    - Crash recovery is a problem with many tables.
  - "Warmup" is a problem
    - Only one table can be opened at a time (5.0)
    - Stats update on open makes it quite slow

Large Buffer Pool Size

- In general the more memory you can get for buffer pool the better it is.
- Watch out for **Warmup**
  - Larger buffer pool means longer warmup time
  - 32GB Buffer pool will take an hour to fill
    - reading 600 pages/sec
- SHOW STATUS and SHOW INNODB STATUS gets expensive
  - It takes a global lock to count dirty pages

Large Buffer Pool

- Clean shutdown time may be long
  - Buffer Pool needs to be flushed
  - Set **innodb_max_dirty_pages_pct=0** in advance
- Checkpointing activity may cause uneven performance
  - The “dip” is often longer and deeper with large buffer pool

Replication Speed

- Replication can get behind much earlier when Master or Slave are saturated
  - Becoming more problem with Multi-Core CPUs
  - Or if you do not have RAID with BBU on the slave
- Limited by Transaction Commit speed
  - Set **innodb_flush_log_at_trx_commit=2**
    - Replication is Asynchronous anyway
- Limited by Update execution Speed
  - Disk – Prefetching may help
  - CPU – Check row level replication in 5.1+

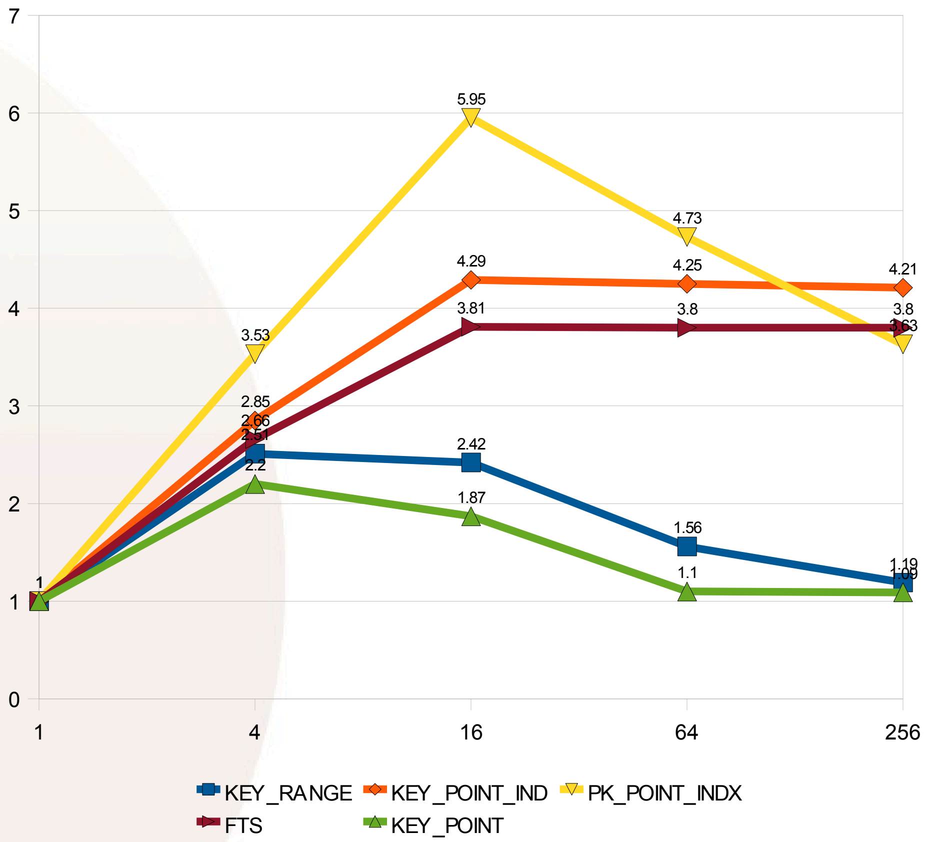
Lets do some Benchmarks

Take Results with grain of salt

Workload Scalability 5.0

- MySQL 5.0.51a
- Dell PE 2950
- 2* Quad Core CPUs
  - Intel Xeon L5335
- CPU Bound
- Scaling depends on workload a lot

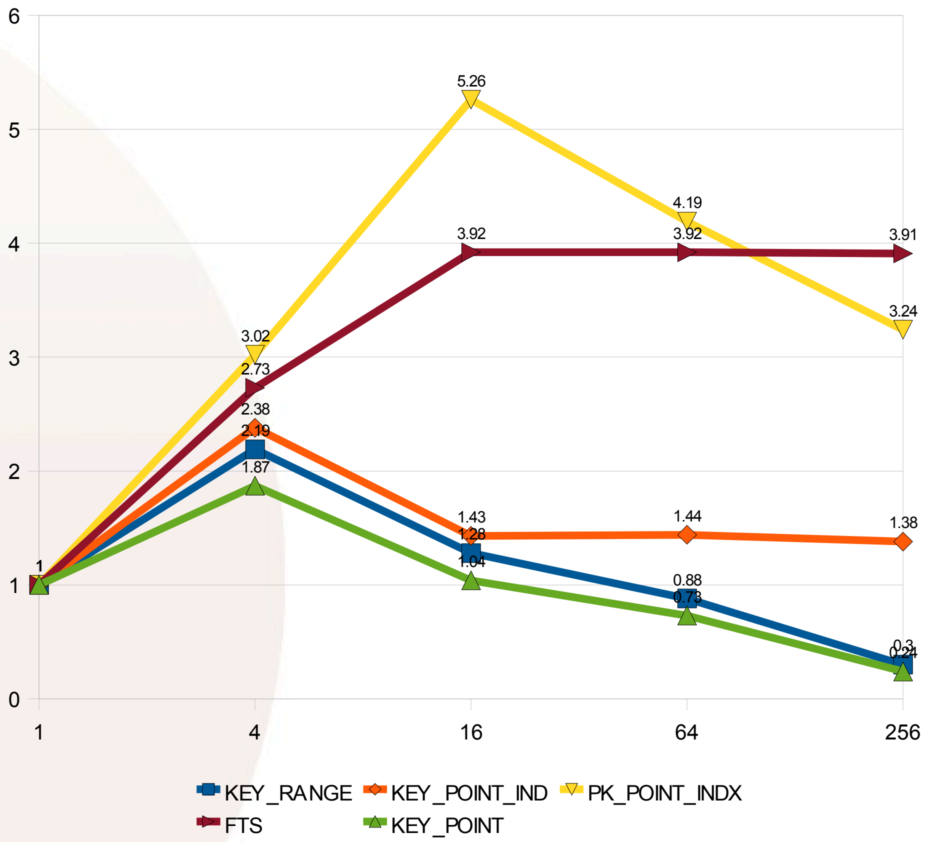
- Scaling factor for different number of threads



Workload Scalability 5.1

- MySQL 5.1.23-rc
- Everything same but MySQL Version
- We can see serious regressions for some workloads

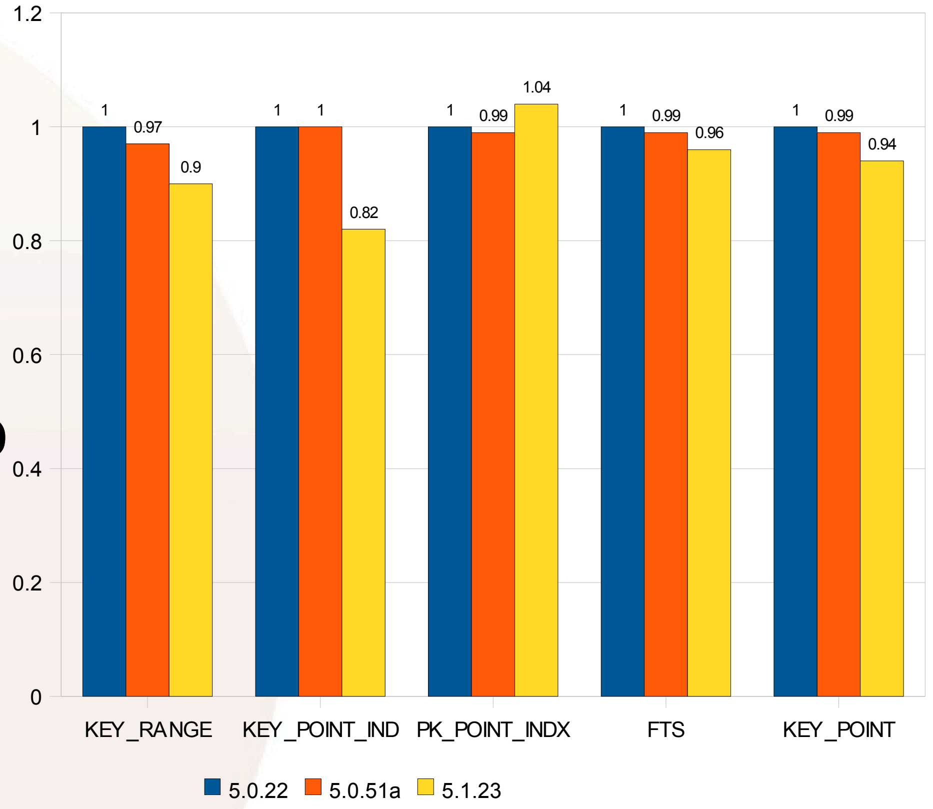
- Scaling factor for different number of threads



Single thread performance

- Performance for one thread
- Scaling factor alone is not conclusive
- Actual results too different – use ratios to 5.0.22 instead
- 5.0.22 and 5.0.51 are very close
- 5.1 shows some regressions

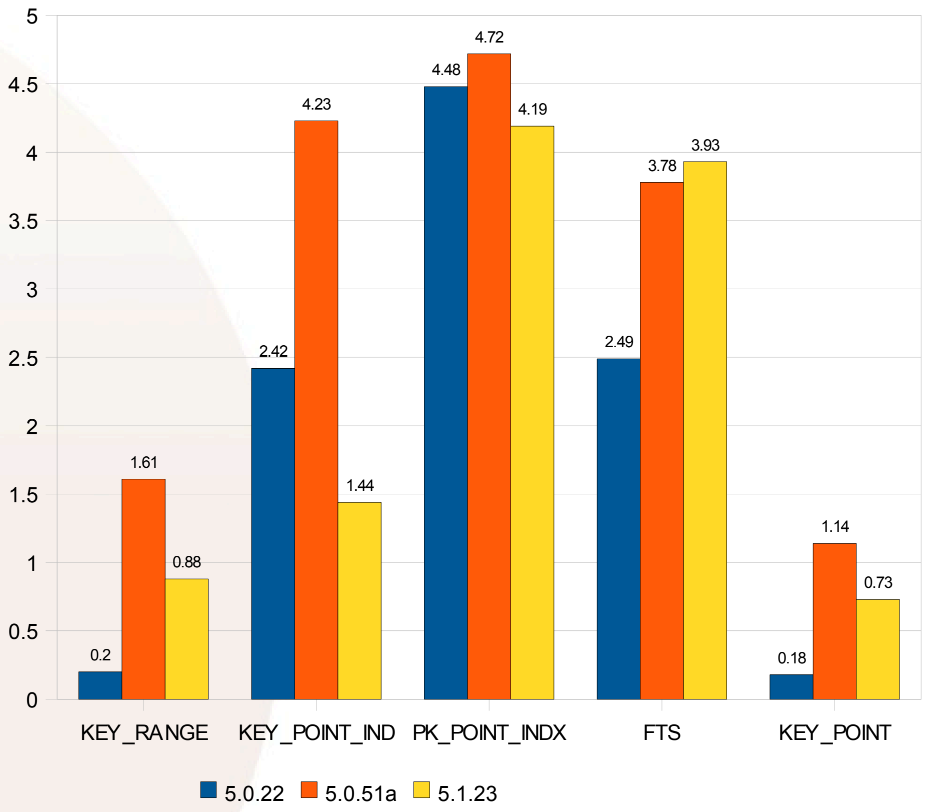
- Relative Performance of MySQL versions



MySQL Versions Scalability

- Scaling factor for 64 threads
- MySQL 5.0.51 overall best results
- Good improvements from 5.0.22
- 5.1 shows serious regressions

- Scaling factor for different MySQL versions

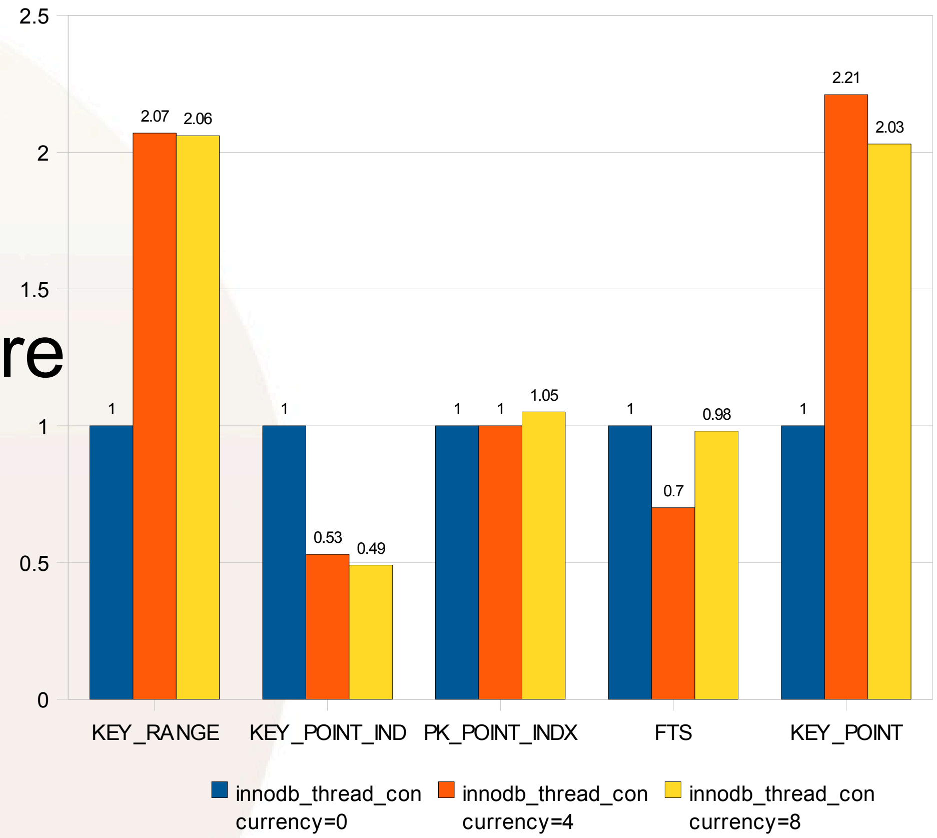


PERCONA
Performance Consulting Experts

innodb_thread_concurrency

- Performance for 64 threads
- MySQL 5.1.23
- Using innodb_thread_concurrency=0 as baseline
- No perfect value – different workloads behave differently

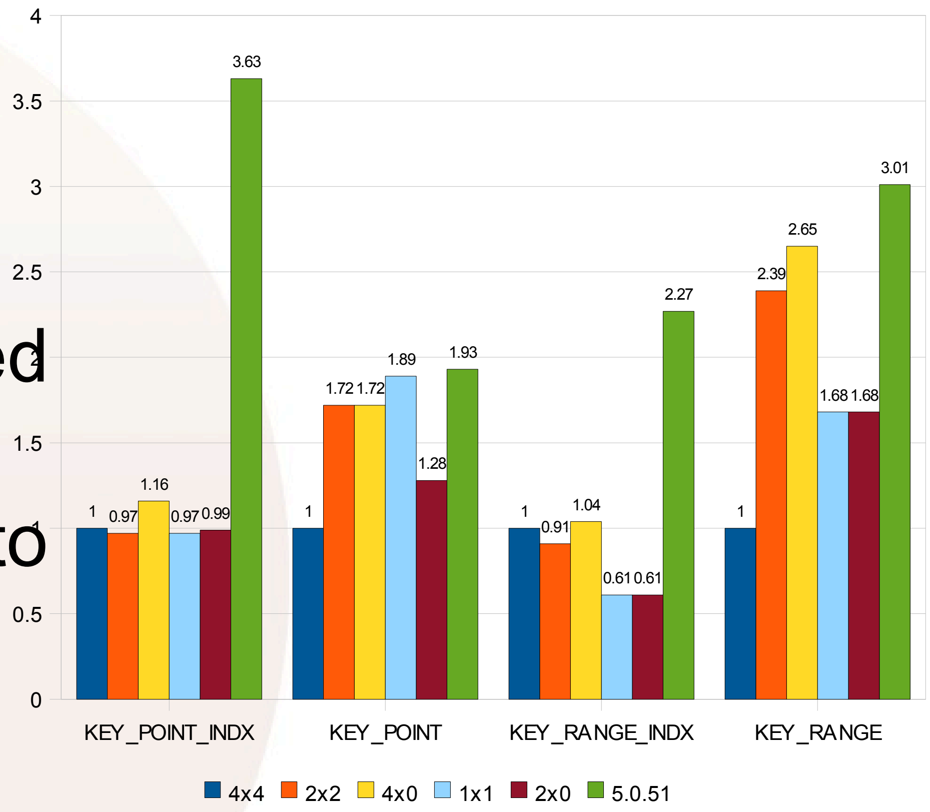
- innodb_thread_concurrency affects performance



Fixing scaling by CPU Affinity

- Performance for 16 threads
- MySQL 5.1.23
  - 5.0.51a for comparison
- Workloads which scaled worse on 5.1.23
- Trying to bind MySQL to specific CPU Cores
- Restricting can help scaling

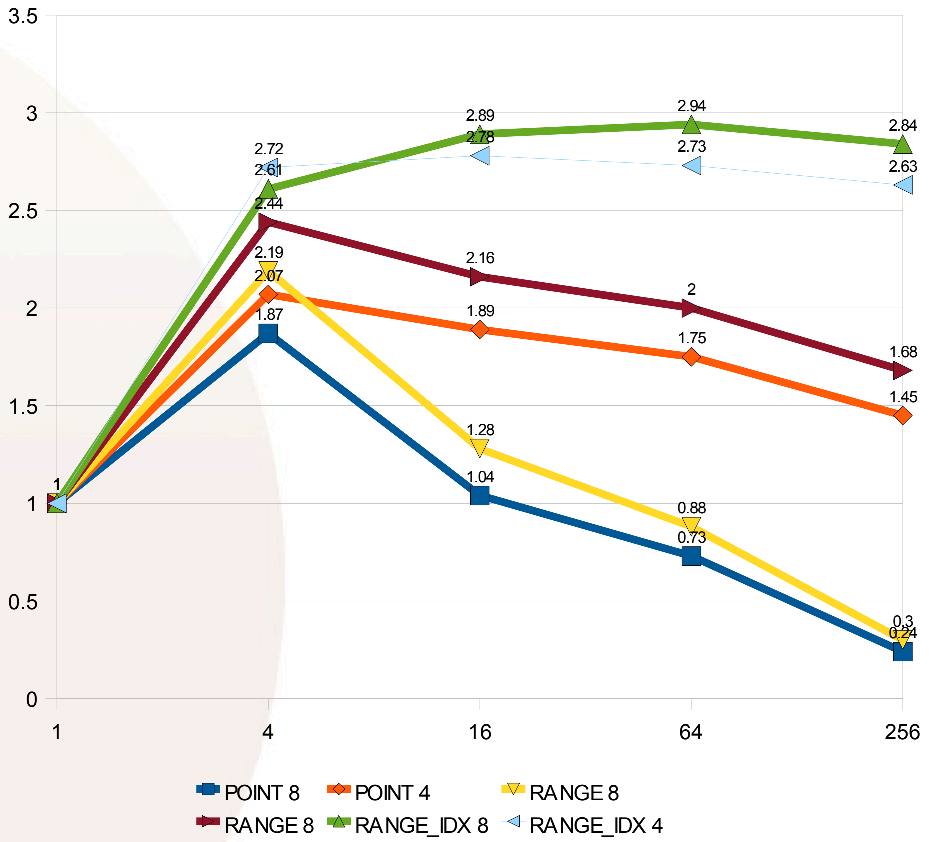
- How binding to CPUs affects performance



Quadcore vs Dual Core

- MySQL 5.1.23rc
- Worst scaling patterns
- For 2 out of 3 query patterns Dual core system scales and performs much better

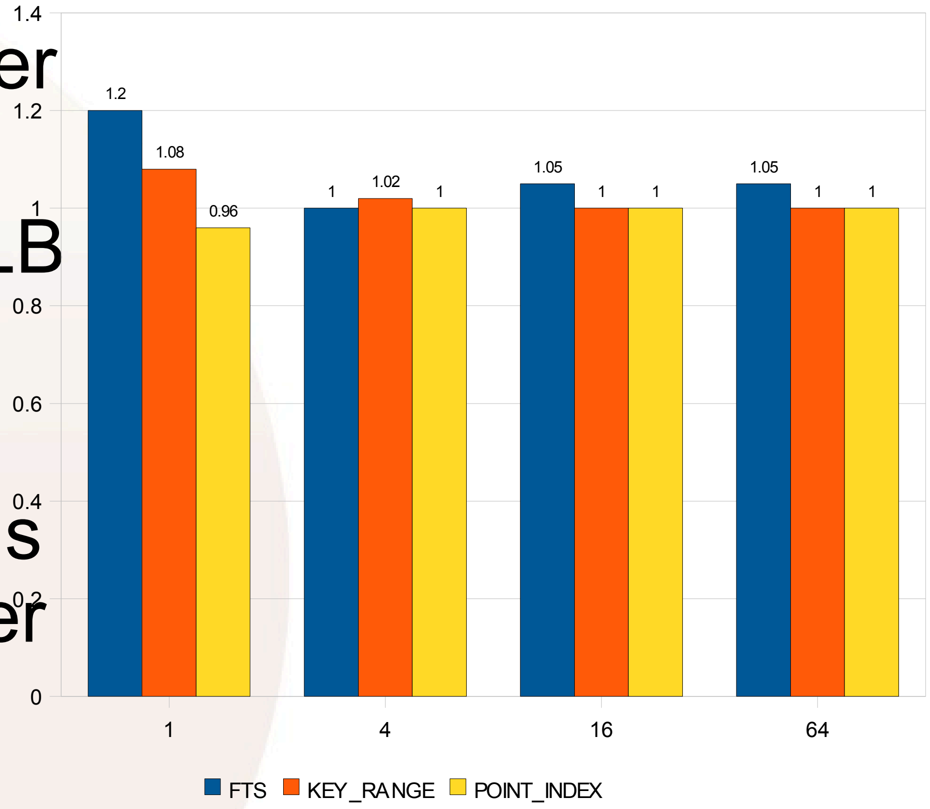
- Scaling factor for different number of threads



Large Pages

- MySQL can use Large Pages for InnoDB Buffer Pool
- Huge Pages reduce TLB cache misses
- Non Swappable
- Mediocre results for this workload, may be better in case of skewed working set

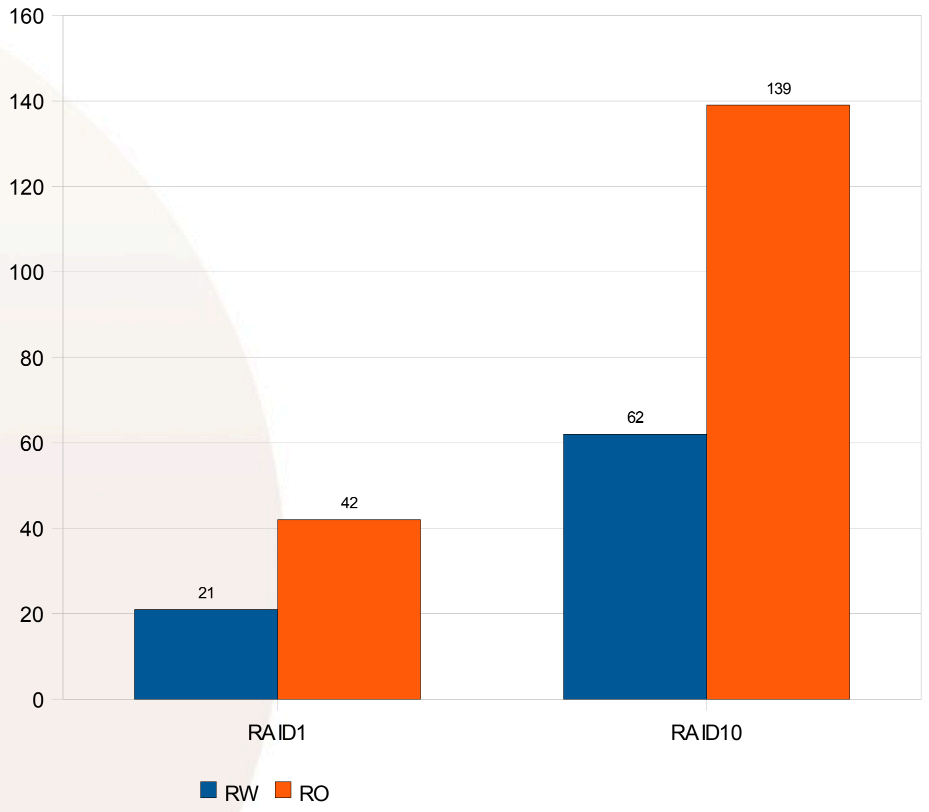
- Performance effect of using Large Pages



InnoDB IO Scalability

- Dell PowerEdge 2950, Perc6, CentOS 5.0
- RAID1 vs RAID10 (6 disk)
- SysBench MySQL Test workload

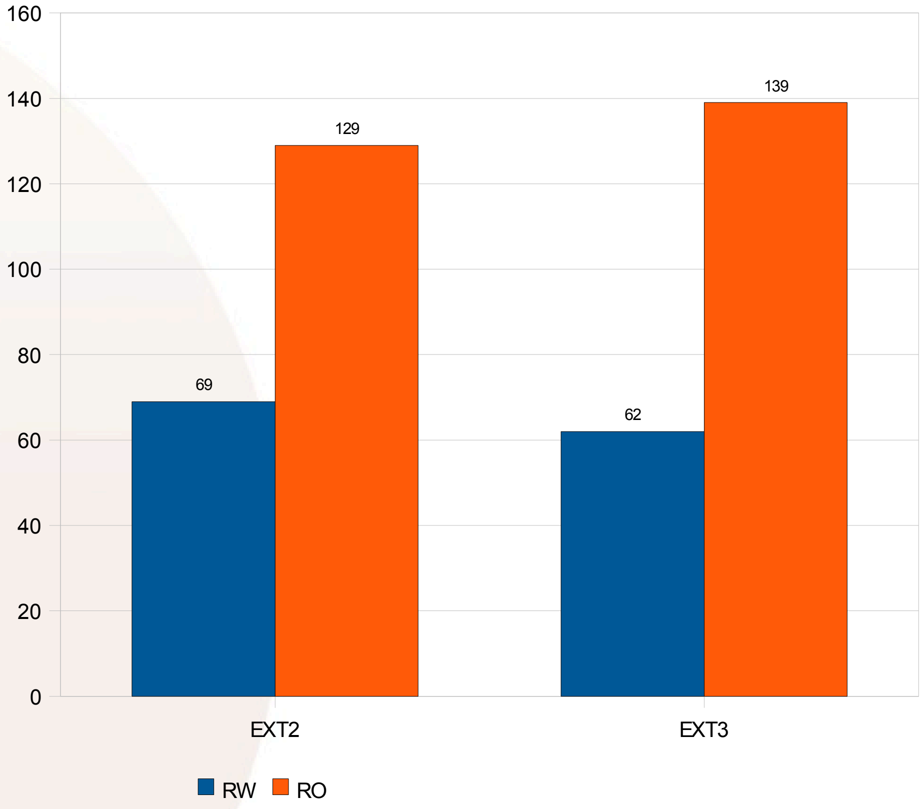
- Performance effect of using Large Pages



EXT3 vs EXT2

- Same Hardware
- RAID10
- Ext3 does worse on writes (journaling overhead) but better with reads

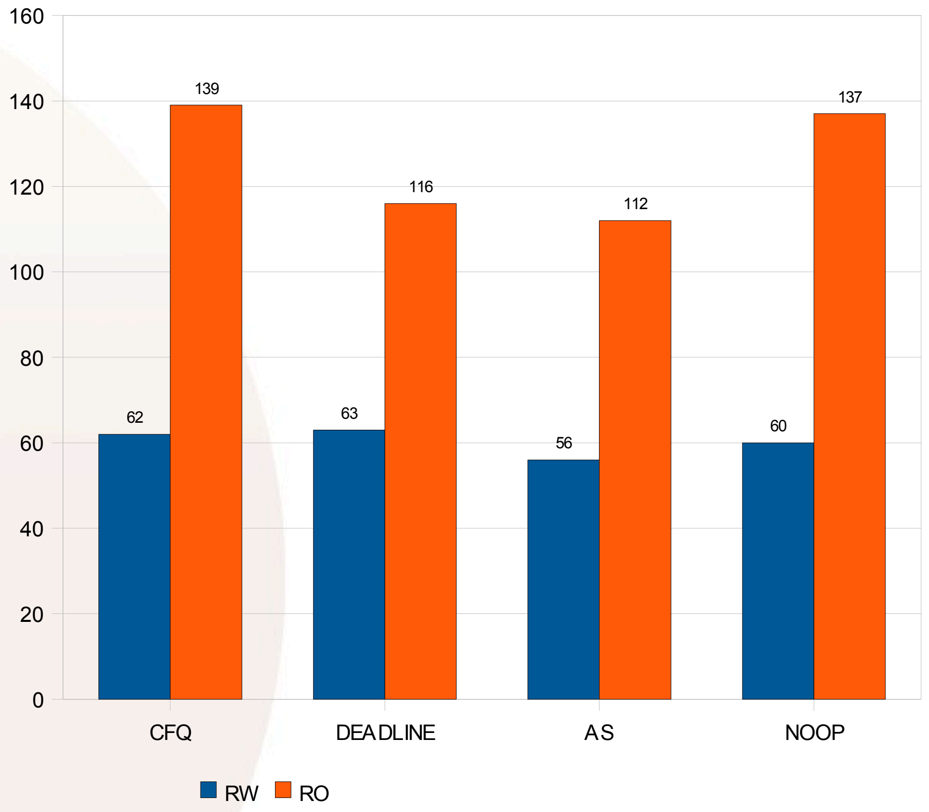
- Performance effect of using Large Pages



Linux IO Schedulers

- Elevators improved
  - Difference is not as huge as years ago
- CFQ (default) is best for this workload on this box

- Performance effect of using Large Pages



Thanks for Coming

- Time for Questions
- Write us
  - pz@percona.com vadim@percona.com
- Come and visit us
  - For Information: http://www.mysqlperformanceblog.com
  - For Business: http://www.percona.com