

MySQL/InnoDB performance OS and Hardware



Peter Zaitsev,
MySQL AB

MySQL Users Conference 2004
Orlando, FL April 14-16

Let me Introduce Myself

- Peter Zaitsev, MySQL Inc
 - Senior Support Engineer
 - Benchmark Project leader
 - Does on site and remote consulting
 - Participates in Server Development
- Before joining MySQL
 - CTO of www.mytrix.com - Web statistics projects
 - One of the largest MySQL/InnoDB installations in Russia at the time
- Based in Seattle, WA

About Presentation

- Optimizing MySQL/InnoDB Performance
 - Two presentations in series
- This part is about OS (Kernel) and Hardware configuration
- Using DBT2 by OSDL, TPC-C like benchmark
- Practical approach – how to locate and eliminate next bottleneck
- Real performance figures
 - Numerical value of each performance improvement
- Why InnoDB ?
 - MyISAM can't run transactional benchmark
 - There seems to be lack of InnoDB optimization info

Question Policy

- Interrupt me if something is unclear
- Keep long generic questions to the end
- Approach me during the conference
- Write me to peter@mysql.com

DBT2 Benchmark Info

- Developed by OSDL (Mark Wong and Co)
 - MySQL port done by Alexey Stroganov (not in the mainline yet)
- More information: <http://sourceforge.net/projects/osdl/dbt>
- Quite close TPC-C implementation, but allows wider parameter range
- Results are incompatible with TPC-C and can't be compared to them
- TPC-C Benchmark Description
<http://www.tpc.org/tpcc/detail.asp>

Benchmark Configuration

- Two workloads “large” and “small”
- 200 Warehouse database – about 30Gb real size on the disk
- “small” workload touches 10 of them, being CPU bound
- Using 200 “terminals” and 20 connections in all cases
- Zero think delay to fully load database
- Hardware
 - 4*Xeon 2.0 Ghz MP (with HT), 512K cache, 4G of memory
 - 8SATA 7200 drives in RAID10, 1024K chunk on 3WARE8500-8
 - System on Separate set of SCSI drives
- Software: RH AS 3.0, MySQL 4.1.1-alpha

Running Benchmark

- Default Kernel: 2.4.21-9.ELhugemem
- Default Filesystem: EXT3
- Swap partition disabled
- Run series
 - 4 runs, 2 “large” load , followed by 2 “small” load
 - Sleeps between loads to allow database to settle down
 - 15min + 5 min warmup for each of loads
- Best out of 2 results is taken in most cases
- Additional experiments performed to measure accuracy of approach
- Results in TPM (transactions per minute), More is better

Performance of Different MySQL Binaries

- Max version is faster in this test
 - Probably due to new (2.3) GLIBC being used
- Recompiled by GCC 3.2 binary is faster
- NPTL is close LinuxThreads

	LARGE	Ratio	SMALL	Ratio
Static (default)	1250		4640	
Dynamic (max)	1279	1.02	4695	1.01
Compiled	1297	1.04	4996	1.08
Compiled,NPT	1293	1.03	5050	1.09

EXT3 filesystem mount options

- There is limited improvement from `data=writeback,noatime`
- Note this mode can result in garbage in the end of files – be careful!

	LARGE	Ratio	SMALL	Ratio
<code>ext3,default</code>	1250		4640	
<code>ext3,data=writeback,noati</code>	1299	1.04	4914	1.06

Kernel compile time settings

- These are two binary kernel versions shipped with RH AS 3.0
- The main difference is 3:1 vs 4:4 memory split
- The overhead of 4:4 memory split is huge
 - Our initial kernel choice was not optimal

	LARGE	Ratio	SMALL	Ratio
2.4.21-9.ELhugemem	1250		4640	
2.4.21-9.ELsmp	1485	1.19	8132	1.75

Kernel vendors and Versions

- No clear Winner
- RedHat kernel is best for “small” load
- Vanilla 2.4 kernel is best for Disk Bound load
- 2.6.3 kernel is still to catch up with 2.4

	LARGE	Ratio	SMALL	Ratio
2.4.21-9.ELsmp (RH AS)	1485		8132	
2.4.21-196-smp (SLES 8)	2005	1.35	7252	0.89
2.4.25 (vanilla)	2209	1.49	-	
2.6.3 (deadline scheduler)	1872	1.26	7630	0.94

2.6.3 Kernel IO Schedulers

- Kernel 2.6 allows to use different IO Schedulers
- Default is “anticipatory” designed for desktop and file servers
- Databases normally do better with “deadline”
 - A lot better in our case
- Still it is not as good as you can get with 2.4 kernel

	LARGE	Ratio	SMALL	Ratio
Anticipatory	515		7587	
Deadline	1872	3.63	7630	1.01

File systems, Kernel 2.4.21-9.ELhugemem

- Avoiding transactional overhead of EXT3 improves performance
- ReiserFS is even better choice
- JFS for some reason is even faster than RAW
 - Can be related to 4:4 split
- Comparison to RAW is not exactly fair

	LARGE	Ratio	SMALL	Ratio
EXT3	1250		4640	
EXT2	1305	1.04	4986	1.07
ReiserFS	1314	1.05	4956	1.07
JFS	1463	1.17	4712	1.02
Raw, Logs on separate di	1432	1.15	4651	1

File Systems (Kernel 2.6.3, deadline)

- ReiserFS is a bit better once again
- ReiserFS 4 is outstanding in “small” load but slowest in “large” load
 - Hans Reiser reports fsync() is not optimized in this version yet
- XFS is worse than EXT3 for both loads but not significantly

	LARGE	Ratio	SMALL	Ratio
EXT3	1872		7630	
ReiserFS 3	1904	1.02	7656	1
ReiserFS 4 (alpha)	1644	0.88	10509	1.38
XFS	1780	0.95	7543	0.99

Performance of Direct IO

- Direct IO is enabled by `innodb_flush_method=O_DIRECT`
 - Not all filesystems and kernels support it yet
- Direct IO seems to reduce 4:4 memory split penalty

Kernel 2.4.21-9.ELhugem	LARGE	Ratio	SMALL	Ratio
Buffered IO	1250		4640	
Direct IO	1931	1.54	5543	1.19

Kernel 2.6.3 (deadline)	LARGE	Ratio	SMALL	Ratio
Buffered IO	1830		7630	
Direct IO	2024	1.11	8702	1.14

Linux native Asynchronous IO (2.6.3)

- Using patch by Christoffer Hall-Frederiksen
 - Not yet in the mainline
 - Performance was not tuned yet
- May work on vendors 2.4.x kernels as well
- Extra context switches in Async IO seems to be the **problem**

Buffered IO	LARGE	Ratio	SMALL	Ratio
Standard IO	1872		7630	
Asynchronous IO	1893	1.01	7534	0.99

Direct IO	LARGE	Ratio	SMALL	Ratio
Standard IO	2024		8702	
Asynchronous IO	2186	1.08	8656	0.99

RAID10 Raid Chunk Size

- RAID10 3WARE8500-8, 8SATA 7200RPM Drives,
- Kernel 2.4.21-9.ELhugemem, EXT3
- No clear winner
- Large chunk is best for “large” workload
- 256K seems to be optimal for set of the two
- Chunk size has huge impact on performance

	LARGE	Ratio	SMALL	Ratio
16K	761	0.61	4719	1.02
64K	1059	0.85	4869	1.05
256K	1237	0.99	4924	1.06
1024K	1250		4640	

RAID Levels

- Using 64K chunk
 - It is only supported by this hardware for RAID5
- RAID0 best results but, not secure
- RAID5 is very slow, especially for “LARGE” workload
- RAID10 is normally the best choice

	LARGE	Ratio	SMALL	Ratio
RAID10	1059		4869	
RAID0 (insecure)	1206	1.14	4925	1.01
RAID5	264	0.25	3422	0.7

RAID5 During maintainance

- It is very important to consider the case in “worse case scenario”
- Serve performance hit in degraded mode
- Rebuilding makes things even worse
- Over 10 times difference between RAID10 and rebuilding RAID5

	LARGE	Ratio	SMALL	Ratio
RAID5 Normal	264		3432	
RAID5 Degraded	165	0.63	2220	0.65
RAID5 Rebuild	73	0.28	1910	0.56

RAID10 Software vs RAID10 Hardware

- 1024K chunk, Kernel 2.4.21-9.ELhugemem, EXT3
- Software RAID can be close or faster than Hardware one
- Sequence of mirroring and stripping matters a lot
- With battery backed up cache situation can change a lot
- No clear winner for both workloads

	LARGE	Ratio	SMALL	Ratio
Hardware (Stripped RAID)	1250		4640	
Software (Mirrored RAID)	1230	0.98	4763	1.03
Software (Stripped RAID)	584	0.47	4957	1.07

Number of disks affecting performance

- “Large” workload scales pretty well
- “Small” workload almost does not scale at all
 - Not every workload will benefit from increasing amount of disks
- We do not expect 100% scalability due to logging overhead

	LARGE	Ratio	SMALL	Ratio
RAID10, 8 Drives	1250		4640	
RAID10, 4 Drives	778	0.62	4797	1.03
Single hard drive	227	0.18	4358	0.94

Put logs on separate disk

- RAID10, 1024K, 8 Drives, EXT3
- This is very common advice for all DBMS !
- Slower for “small” load as SCSI does real fsyncs
- If fsync() would be real, improvement would be larger.

	LARGE	Ratio	SMALL	Ratio
Shared drive for data and	1250		4640	
Logs on dedicated SCSI driv	1355	1.08	4560	0.98

Improvement from using HT

- HT (HyperThreading) – technology to get 2 logical CPUs in one physical
 - Available in Intel Xeons, P4
- It is not fully separate CPUs as a lot of resources are shared
- Testing kernel 2.6.3 as it should have improved HT handling
- “large” load is even a bit faster without “HT”
- “small” load shows some improvement

	LARGE	Ratio	SMALL	Ratio
HT ON	1872		7630	
HT OFF (acpi=off)	1893	1.01	6988	0.92

Should I keep my Swapping enabled ?

- Kernel 2.4.21-9.ELhugemem
- Swap partition on separate SCSI disk
- VM is highly different between kernel versions and vendors
 - Results may vary a lot
- Active swapping with 1800M of buffer pool and 4G of physical memory
- Keep swap off or at least lock MySQL in memory
- Direct IO also helps

	LARGE	Ratio	SMALL	Ratio
Swap partition disabled	1872		7630	
Swap partition enabled	384	0.21	3386	0.44

File system Read-Ahead

- Kernel 2.4.21-9.ELhugemem
- InnoDB implements its own read-ahead so it might not need OS help
- Lets disable it
 - `echo 0 > /proc/sys/vm/min-readahead`
 - `echo 0 > /proc/sys/vm/max-readahead`
- Disabled Read-ahead can improve performance for some workloads
- Direct IO is normally better alternative

	LARGE	Ratio	SMALL	Ratio
Read-Ahead Enabled	1240		4640	
Read-Ahead Disabled	1433	1.16	4569	0.98

Flushing data to the disk background

- `fsync()`, `O_SYNC`, `O_DIRECT` in Linux do not guaranty data is on the disk
 - Drive or RAID may cache the data in their writeback cache.
 - Sometimes cache is battery backed up - safe, sometimes lost on power down
 - It is drive cache not OS cache, so you're OK with kernel crashes
 - Linux 2.4 does not flushes drive cache on `fsync` and synchronous writes
 - There is work on the way in 2.6 to fix it
 - IDE drives have write back cache enabled in most cases
 - Some IDE drives can't disable writeback cache at all.
- Test your `fsync()` speed to be sure
 - Single drive can't do more than 200-300 real `fsyncs/sec`
 - Use “SysBench” to measure

How much would I pay for true durability ?

- Kernel 2.4.21-9.ELhugemem, EXT3
- Results apply to this kernel and hardware only
- Work to improve performance with cache off is done in 2.6
- Battery backed up RAID cache allows to get durability for “free”

	LARGE	Ratio	SMALL	Ratio
Write Cache ON	1240		4640	
Write Cache OFF	187	0.15	2270	0.49

Conclusions

- Hardware selection and optimization is important for performance
- Even the same hardware can be configured to give a lot different performance
- Both kernel version and settings make a difference
- Even minor items may matter for performance (single kernel option, RAID chunk etc)
- Performance difference from Worst case investigated to Best case is over 10 times.

Resources

- <http://www.mysql.com/doc> - MySQL online Manual
- <http://lists.mysql.com> - MySQL mailing list
 - Especially Main mailing lists, Benchmarks list
- Get help and advice from MySQL employees
 - <http://www.mysql.com/support> - Support
 - <http://www.mysql.com/consulting> - Consulting
- Write me to peter@mysql.com if you have questions
- Ask me during the the conference
- Come to the next MySQL User Conference