

ORACLE®




ORACLE®

It's 10 Years After Y2K But We Still Party Like It's 1999

Kevin Closson

Performance Architect, Exadata Development Team

Database Server Technologies



The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.



Agenda

- We still think a CPU is a CPU.
- We still think memory is memory.
- We still think bulk data loading is a disk-intensive operation.
- We still think direct-attached storage is for "small" systems.
- We still think "commodity server" means small.
- We still think database==structured and file system==unstructured.
- We still think NUMA is niche technology.
- We still think NFS is a file serving protocol.



Welcome to 2010
Let's Party Like It's 1999

We Still Think a CPU is a CPU



We Still Think a CPU is a CPU

- CPU, Core, Thread, Integer Pipeline, what the heck is my process executing on?
- Where we've been...
- We've seen the coming *and going* of HyperThreading (HT)
 - Netburst Architecture. Northwood/Foster MP Xeons – 2002
 - A.k.a Symmetric Multithreading (SMT)



We Still Think a CPU is a CPU

- We've seen the coming *and staying* of multi-core CPUs
 - Dual-Core
 - Dual-core AMD Opterons – May 2005
 - Paxville DP Xeon – Oct 2005 (Netburst)
 - Woodcrest Xeon 5100 - 2006 (Core Architecture, shared L2) – No HT
 - Quad-Core
 - Clovertown Xeon 5300 – 2006. (“Glued Woodcrest”, 2x4MB L2,etc)
 - AMD “Barcelona” Opteron – 9/2007 (65 nm aimed at Clovertown)
 - Harpertown Xeon 5400 – 11/2007 (2x6MB L2, enter Seaburg chipset huge snoop filter, 128GB memory)
 - Hex-Core and more
 - Dunnington Xeon 7400 – 9/2008 (16MB L3)
 - AMD “Istanbul” Opteron – 6/2009, Magny-Cours 12 Core...

...that leads us to



We Still Think a CPU is a CPU

- The re-emergence of Intel HyperThreading (HT)
 - Aka **Simultaneous** Multithreading (SMT)
- Enter :
 - Xeon 5500 EP “Nehalem”
 - When BIOS enabled SMT, certain CPU components are partitioned (e.g., reorder, load/store buffers, etc)
 - Operating System sees 16 “CPUs”
 - 2s8c16t
 - API available to distinguish between thread and a core
- “Simultaneous?” Really?



We Still Think a CPU is a CPU

- *Simultaneous* Multithreading?
 - A processor core can only “do” one thing at a time
 - A processor core can be “doing” a lot of things at one time
 - Cores have multiple execution units that can do memory/calculation operations
 - A non-threaded core “stalls” while loading/storing memory
 - A threaded core can switch to another thread when one thread stalls
- Code that doesn’t leave cache periodically essentially “owns” the core



We Still Think a CPU is a CPU

- What Causes Thread Switching?
 - Processor stalls
 - OS interruption
 - Two threads can “think” they are running for an entire scheduling time slice (e.g., 10ms) unless otherwise interrupted
 - It’s quite possible, common and sometimes expected that one thread can usurp the whole core for a whole time slice

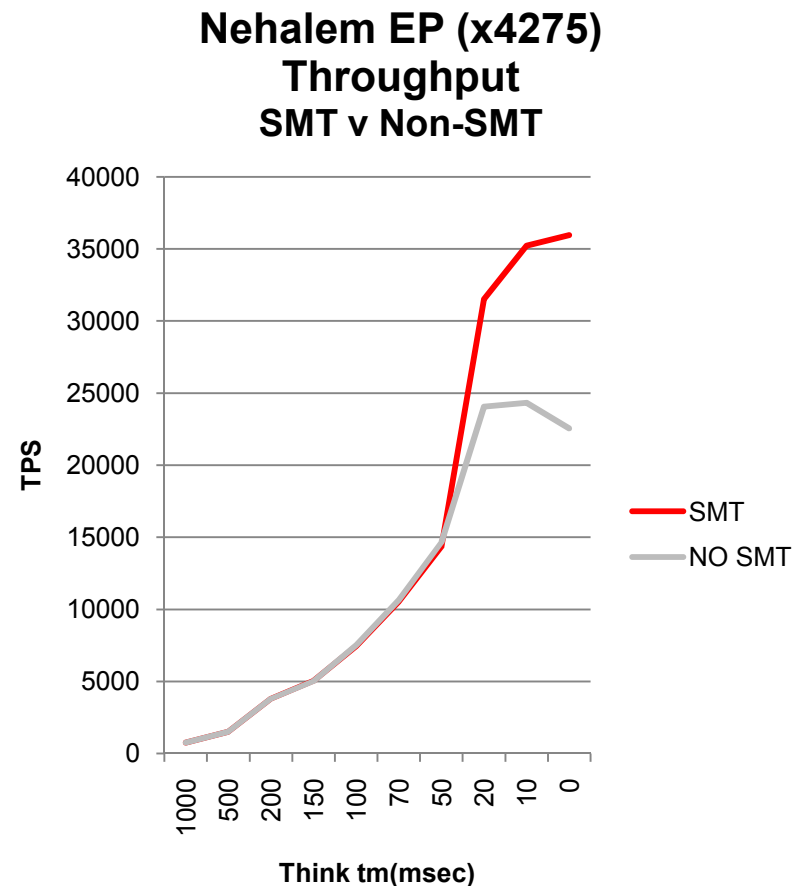


We Still Think a CPU is a CPU

- The OS code can be as simple or as sophisticated as it wants when scheduling on threads.
 - Such “sophistication” requires the OS to become quite intimate with the application
 - Consider: DBWR (PROD), LGWR (DEV), 2 shadow processes (DEV) and a tar process are all runnable, cache warm, same mode (K/U) and same priority.
 - Your Task: Determine what to schedule on the two idle threads of the same core.
 - What, you haven’t solved that problem yet? It’s not a simple topic.
- The holy grail would be an OS that knows how to group complimentary processes into thread-wise classes
 - Don’t expect this

We Still Think a CPU is a CPU

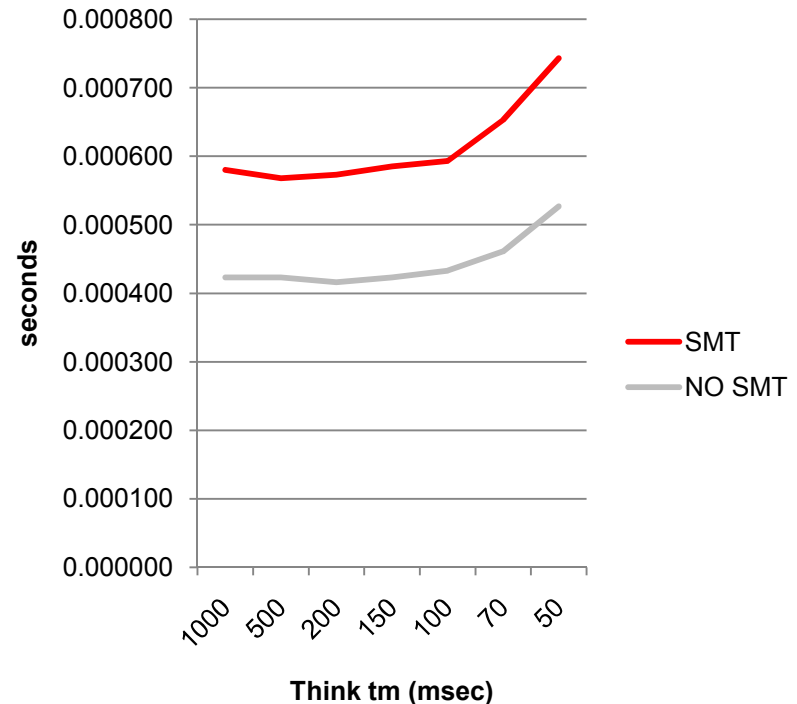
- OK, the graph looks good. That's a good short presentation!
- Yes, SMT takes saturated systems to a higher level of throughput as load increases



We Still Think a CPU is a CPU

- At sub-saturated levels, SMT can **add** to the response times for transactions.
- Having a higher throughput ceiling comes at a slight cost
 - This is a 100% cached workload
 - We are focusing on cpu<->mem characteristics.
 - In this test the low-utilization “tax” is about 40% for having SMT
 - I’m not saying an I/O intensive workload will suffer 40%

**Nehalem EP (x4275) OLTP
Response Times SMT vs
Non-SMT**

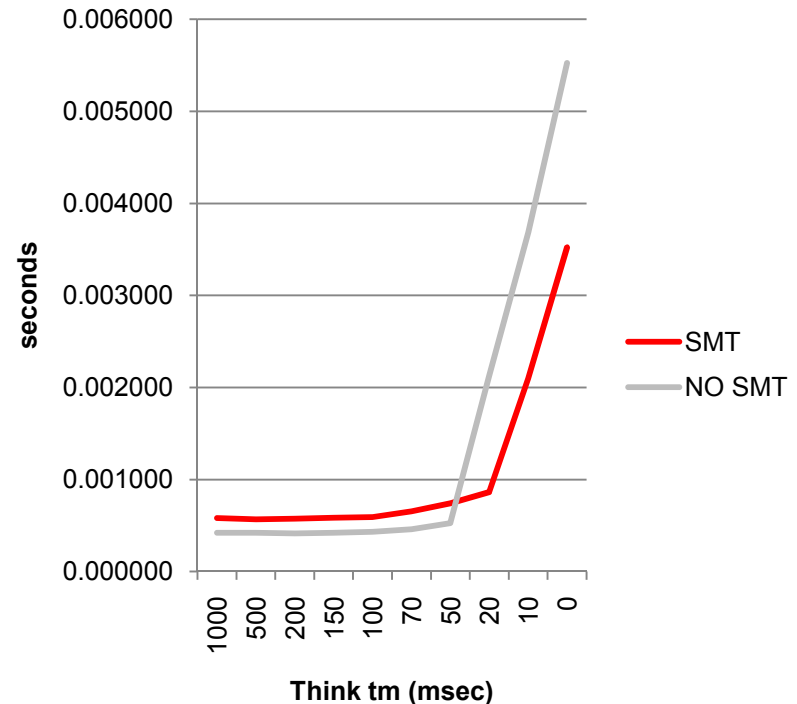


We Still Think a CPU is a CPU

- SMT essentially postpones hitting the **response time** “brick wall”
- SMT “softens the landing”
- Response times under **saturated** conditions :
 - Dr Jeckyl becomes Mr. Hide

Think (ms)	SMT	Non SMT
20	0.9	2.1
10	2.1	3.7
0	3.5	5.5

**Nehalem EP (x4275) OLTP
Response Times SMT vs
Non-SMT**

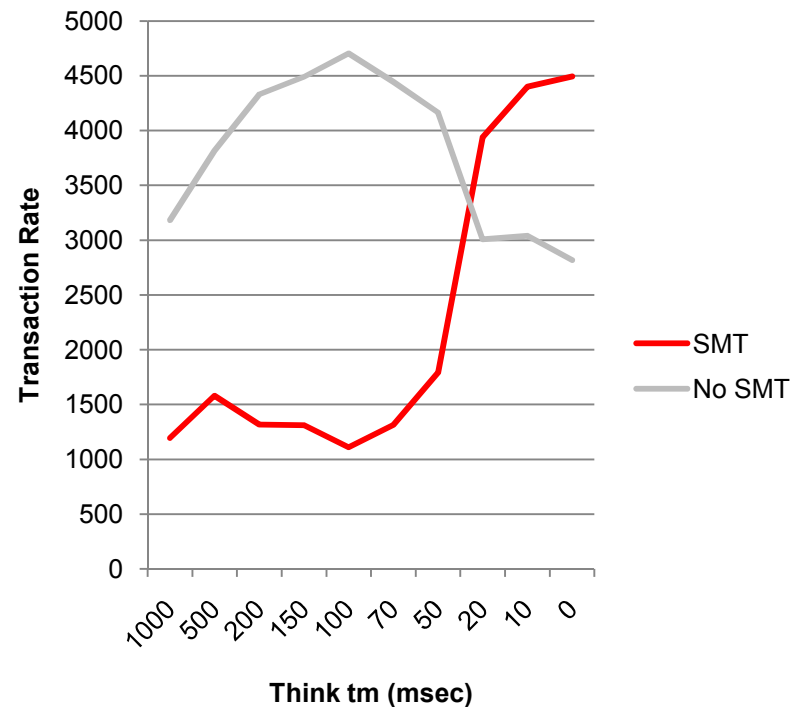


We Still Think a CPU is a CPU

- An SMT thread is treated as a CPU by the Operating System
- Workloads that perform well with 8 OS CPUs (SMT off) will require more OS CPU cycles when SMT is on

Think Tm (ms)	SMT Actual Cores	Non-SMT Actual Cores
1000	0.64	0.24
500	0.96	0.40
200	2.88	0.88
150	3.84	1.12
100	6.72	1.60
70	8.00	2.40
50	8.00	3.52
20	8.00	8.00
10	8.00	8.00
0	8.00	8.00

**Nehalem EP Transaction
Rate per
Core Cycles Used**





Welcome to 2010
Let's Party Like It's 1999

We Still Think Memory is Memory



We Still Think Memory is Memory

- Well, it's not. Now we have FLASH as well
- With the advent of Oracle Database 11g Release 2, there are two tiers of caching
 - The virtual address space buffer pool (db_cache_size).
 - Also known as L1 SGA
 - Database Flash Cache (db_flash_cache_size)
 - Also known as the L2 SGA
- Extremely large caches are feasible
 - For instance, 4 PCI Gen2 slots each with 396GB Sun FlashFire cards



We Still Think Memory is Memory

- Database Flash Cache Facts
 - Not directly mapped into the SGA
 - Buffers have to be read in from the flash cache file to be manipulated by the instance
 - Flash cache is populated by DBWR
 - Based on demand/capacity and aging
 - L1 SGA requires 200 byte header per L2-cached block
 - 25 MB per GB when using 8K blocks
 - 1TB Database Flash Cache needs 25 GB for headers

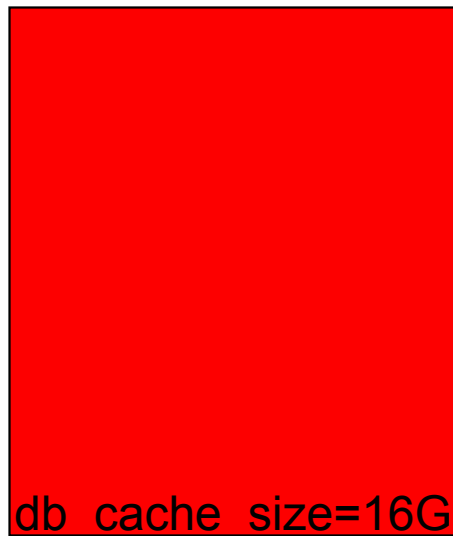


We Still Think Memory is Memory

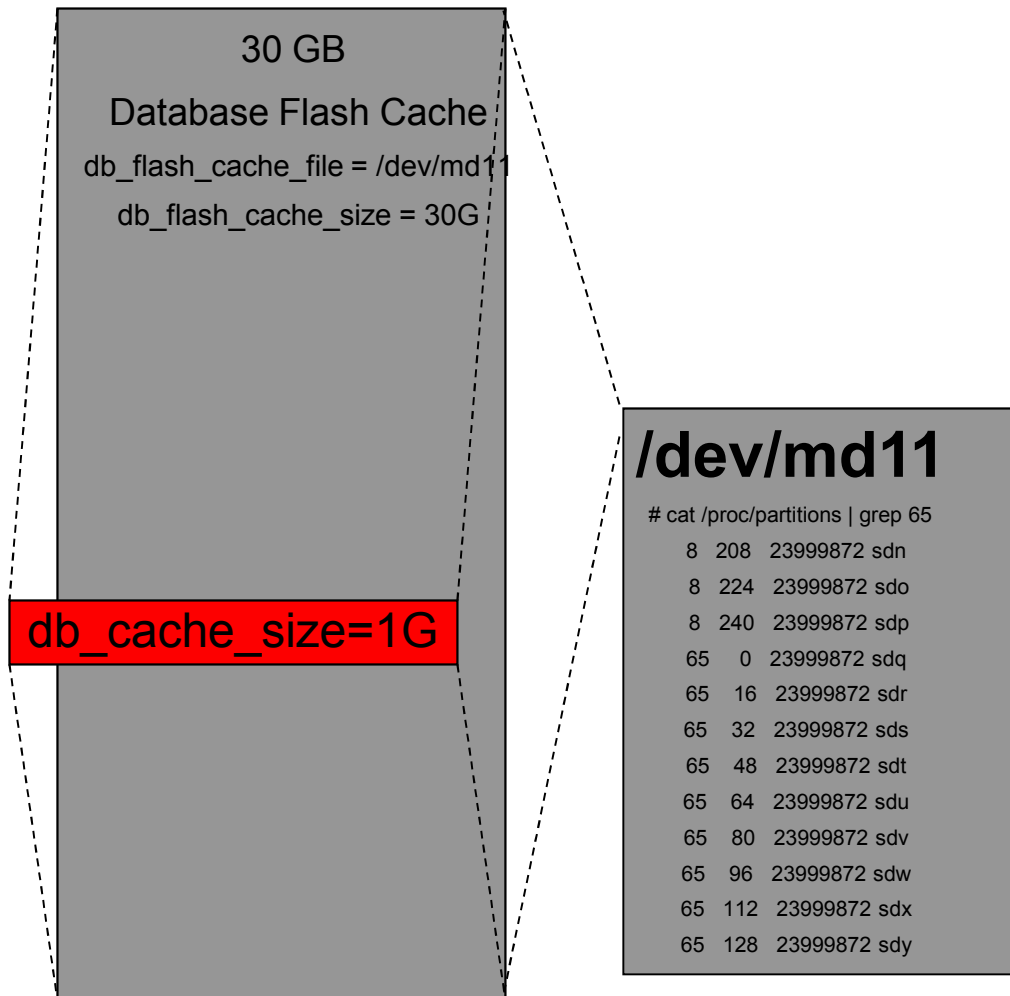
- Database Flash Cache Facts (cont)
 - The db_flash_cache_file assignment is a **single file**
 - Can be any file (file system or raw device)
 - Can be downwind of any protocol. It's just a file.
 - FS-> SRP (IB) > any FLASH storage
 - RAW->iSCSI->SSD
 - RAW->md(4)->PCIe SCSI "disks" (e.g., Sun FlashFire, FusionIO, TMS)
 - Can be HW or SW RAID
 - Standard datafile I/O is used for reading/writing Flash Cache
 - Direct I/O and async I/O if filesystemio_options=setall
 - Database Flash Cache is not mandatory for the Instance
 - If anything "goes wrong" with the flash cache the instance just invalidates it and stops using it. No interruption.



DRAM Model



Database Flash Cache Model



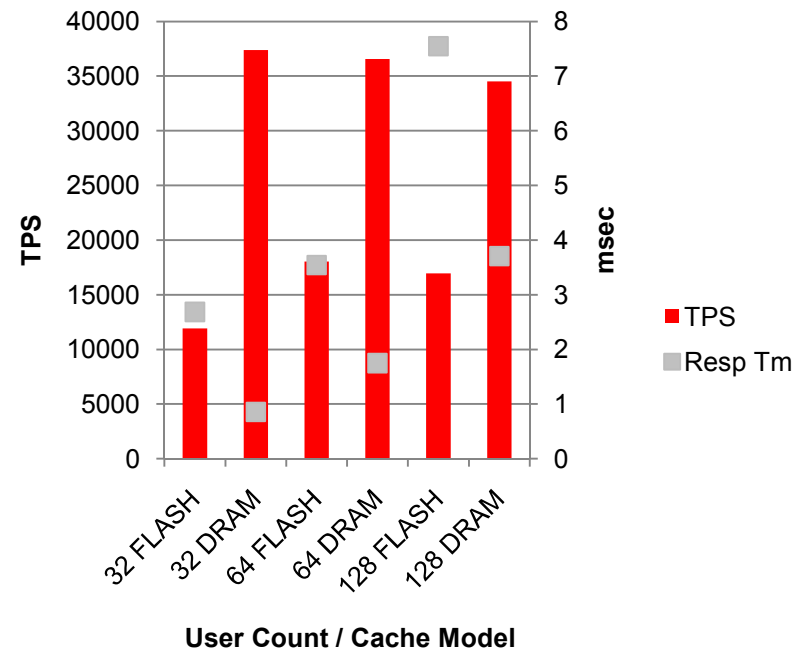
We Still Think Memory is Memory

Test Database: Smaller Than RAM

- The first performance characteristic to assess in any external caching model is what tax is imposed on small dataset workloads?
 - Collect baseline on fully-cached DRAM test
 - Compare to artificially small L1 SGA backed by sufficient L2 SGA
 - An L1 versus L2 comparison
- Compare at different levels of saturation
 - Throughput and response times are important

16G L1 SGA vs 1G L1 + 30G L2 (Flash)

**Read Intensive Workload -
Fully Cached**

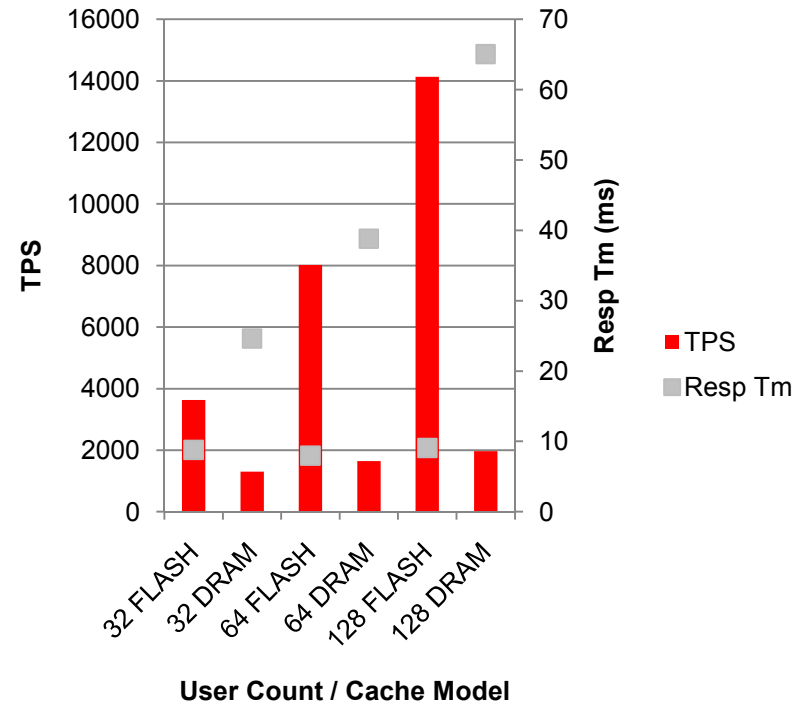


We Still Think Memory is Memory

Large Scale Read-Intensive

- So, as long as physical reads from Database Flash Cache are faster than reads from datafiles, Flash cache is a no-brainer
 - If your storage is, say, Sun 5100 Flash storage or SSD then it is as fast or faster than Database Flash Cache
- Database Flash Cache I/O includes scheduling overhead
 - A 1ms Flash Cache physical I/O may include many ms of scheduling delay
 - Should still be faster than ~6ms HDD + scheduling overhead

16G L1 SGA vs 16G L1 + 270G L2 (Flash)
Read Intensive Workload

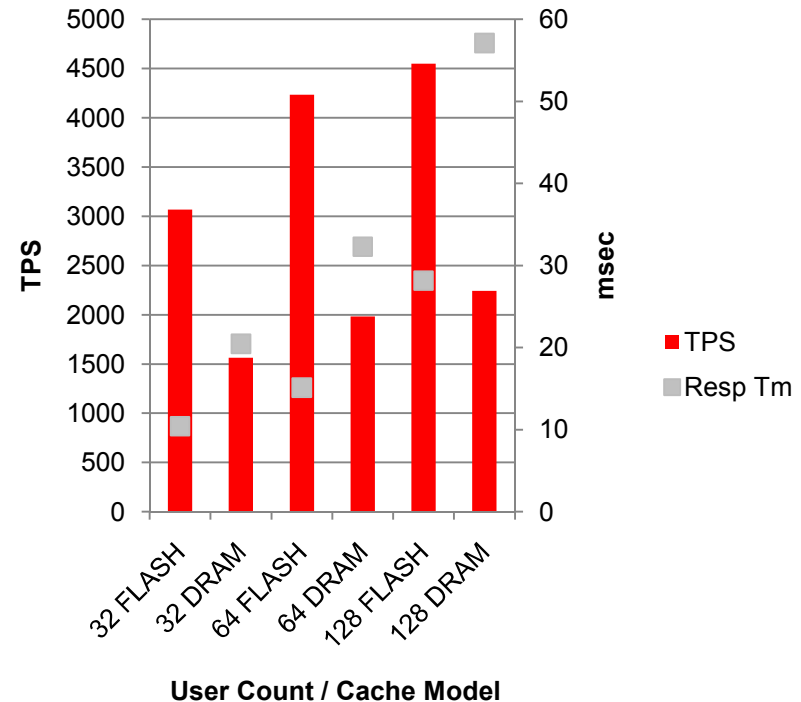


We Still Think Memory is Memory

Large Scale Write-Intensive

- Database Flash Cache do not speed up writes
 - Workload writes come with increased overhead with Database Flash Cache
 - Expect increased transaction response times if HDD side is not commensurate with Flash side (bandwidth)
- Should still be faster than HDD alone as Flash Cache relieves HDD pressure (a cache hit means one less datafile IOP)

**16G L1 SGA vs 16G L1 +
270G L2 (Flash)**
Write Intensive Workload



We Still Think Memory is Memory

- Database Flash Cache changes the physical I/O profile.

- Generally speaking, there will be more writes per read

- L2 SGA “hits” require a physical I/O from Flash
- L1 aged buffers are written to the Flash Cache

- More datafile reads means more Flash writes

	LIO/tran	PW/trans	PR/trans	BCHG/ tran	R:W
Flash	17.6	0.5	2.4	2.4	4.6:1
No Flash	17.2	0.3	2.1	3.0	7.3:1



**Welcome to 2010
Let's Party Like It's 1999**

**We still think bulk data loading is
a disk-intensive operation**



We Still Think ...data loading is a disk-intensive task

- It is if you are still partying like it's 1999
 - The best drives of the era were 18GB 15KRPM Seagate Cheetah which offered 2GFC FC connectivity and an internal track buffer fill/flush rate of 50MB/s
 - We packed these drives into drawers with, perhaps, 2x2GFC loops
 - We packed the drawers behind array heads that sustained, maybe, 600 MB/s.
 - Load that with a RAC cluster and data loading would seem like a disk throughput sensitive workload
- ... and... we loaded with SQL*Loader as opposed to External Tables
- Modern high-bandwidth storage is very easy to find and deploy
 - Exadata
 - General-purpose IB (e.g., SRP)
 - 10 GbE iSCSI and NFS with direct I/O
 - 8 GFC Fibre Channel



We Still Think ...data loading is a disk-intensive task

- So, data loading in the modern world is a host cpu and network intensive operation
 - ...and it doesn't take all that much network to saturate even the very best processors
 - Converting ASCII flat file data to internal data types is very cpu-intensive
 - Instruction intensive
 - Cache intensive
- What to do if a particular loading operation is network bound?
 - More on this in a moment

We Still Think ...data loading is a disk-intensive task

- Study based on Winter Corporation Exadata Proof of Concept

```
create table all_card_trans
(
  card_no      varchar2(20) NOT NULL,
  card_type    varchar2(20),
  mcc          number(6) NOT NULL,
  purchase_amt number(6) NOT NULL,
  purchase_dt   date NOT NULL,
  merchant_code number(7) NOT NULL,
  merchant_city varchar2(40) NOT NULL,
  merchant_state varchar2(3) NOT NULL,
  merchant_zip number(6) NOT NULL
)
```

```
$ more /nfsdata/mnt1/data/all_card_trans.ul
8068355617582716|VISA|5199|94|08/27/2009|8122615|Colorado Springs|CO|80920
6888978486961227|VISA|7641|38|09/14/2009|6600355|Fort Lauderdale|FL|33323|
5013171935966192|VISA|5812|44|04/18/2009|1554005|Lancaster|CA|93536|
4765526842173533|VISA|4722|58|07/09/2009|8224398|Rosalie|NE|68055|
8715822561929847|VISA|5499|24|10/30/2009|6076244|Carson City|NV|89714|
9560787167589693|VISA|7278|59|04/10/2009|5255392|Dupont|WA|98327|
8190159616334782|VISA|3501|26|06/26/2009|8022942|Turtle Creek|WV|25203|
4256994226963241|VISA|8031|91|05/09/2009|3558670|Walcott|WY|82335|
6284823448273263|VISA|5942|88|12/24/2009|5971082|Oakville|TX|78060|
6342853444192223|VISA|5422|84|11/22/2009|5427853|Jamestown|OH|45335|
8543125648836414|VISA|742|45|07/09/2009|2071552|Red Springs|NC|28377|
9018895545327437|VISA|7991|43|09/18/2009|1755891|Ball Ground|GA|30107|
5682741567383923|VISA|5310|38|03/22/2009|3851393|West Milford|WV|26451|
5394385546988611|VISA|5111|40|10/12/2009|5151576|Farmland|IN|47340|
4835131414913578|VISA|5722|24|10/21/2009|1327176|Millers Creek|NC|28651|
8106313891733686|VISA|5947|68|01/01/2010|3810233|Cissna Park|IL|60924|
4801523988495967|VISA|5300|56|03/25/2009|3097791|New Orleans|LA|70158|
4687666269522768|VISA|5199|86|11/16/2009|5851124|Kendrick|ID|83537|
7154151318666777|VISA|7210|69|06/04/2009|7726808|Bellevue|IA|52031|
6333928915122183|VISA|5977|58|11/02/2009|8706826|Wardtown|VA|23482|
7633122734623646|VISA|6011|95|06/04/2009|1865450|Granite City|IL|62040|
```



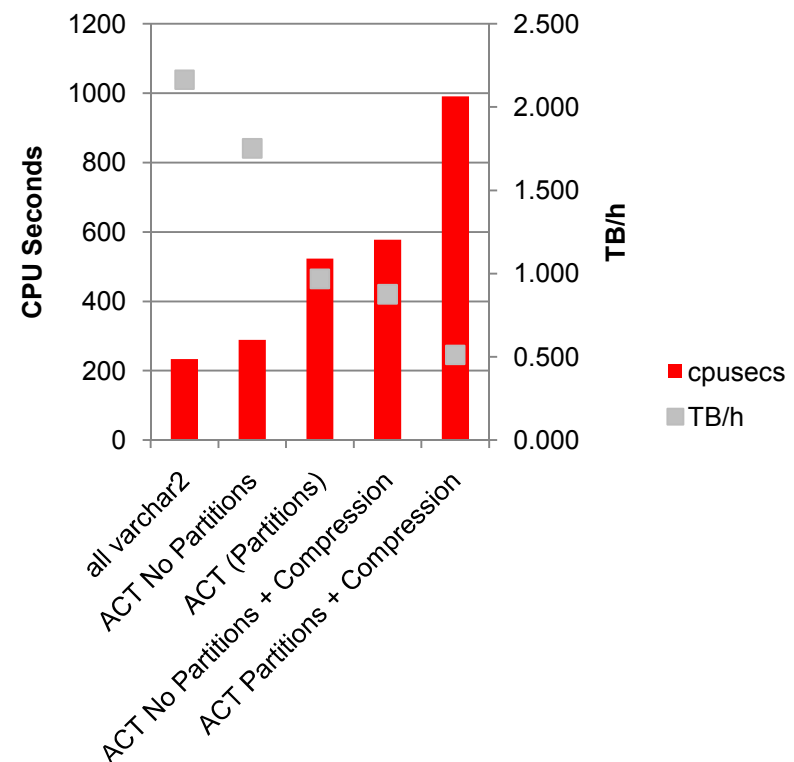
We Still Think ...data loading is a disk-intensive task

- Study 4 Data Loading Models focused on loading into ALL_CARD_TRANS (ACT) table:
 - ACT No Partitions
 - Load into the ALL_CARD_TRANS table non-partitioned
 - ACT (Partitions)
 - Load into ALL_CARD_TRANS table partitioned
 - partition by range (purchase_dt)
 - subpartition by range(card_no)
 - ACT No Partitions + Compression
 - Standard compression, not Hybrid Columnar Compression
 - 2.7:1 compression ratio
- ACT Partitions + Compression

We Still Think ...data loading is a disk-intensive task

- Modeled on 2s8c16t Nehalem EP
 - Use a single Database Host with Parallel Query
 - Target tablespace in Exadata Storage Server
 - Database character set US7ASCII
 - Load 9 GB flat file data
 - From NFS mount
 - External Table with O_DIRECT! (bug fix 9301862)
 - Measure CPU cost
 - Correlate to TB/h load rate

Data Loading Attributes and Processor Utilization





We Still Think ...data loading is a disk-intensive task

- But what if my loading operation is ingest-side network bound?
 - Load compressed data via the Oracle Database 11g PREPROCESSOR feature
 - Compressed ASCII files are fed to PQ slaves with a script
 - Script considerations:
 - Scrape the data with direct I/O! E.g., `dd iflag=direct`
 - Have `dd` (or equivalent) feed `gzip -d -c`
- But don't go overboard
 - Do you really need to compress, say, 9X?
 - Costs a lot at both compression time and decompression time
 - Likely would overdrive the load side anyway
- Consider...



We Still Think ...data loading is a disk-intensive task

- Consider...
 - What is my loading goal?
 - E.g., 1 TB/h?
 - But I “only” have 2xGbE for ingest-side data flow
 - ~230MB/s == 4,600 second (I’m 16 minutes over budget)
 - Do I need to compress this 9x to make my 1TB/h goal?

No



We Still Think ...data loading is a disk-intensive task

- Compress to a lower degree, say, gzip -3 (-6 is default)
 - Example,:
 - Winter PoC ALL_CARD_TRANS data compresses 2.1:1 with gzip -3
 - To produce 291MB/s (satisfies 1 TB/h) I only need 138 MB/s (physical) from the compressed files which inflates to ~291MB/s
 - My in-bound data flow requirement is therefore satisfied
 - At 2.1:1 compression gzip -d (to stdout) can produce 20 MB/s at 25% of 1 Nehalem EP core
 - To load 291 MB/s (1TB/h) I need 15 “units” of 2.1:1 gzip -d data flow or ~4 Nehalem EP cores



We Still Think ...data loading is a disk-intensive task

- There, we've converted a network bottleneck to a CPU bottleneck
- Fact about PREPROCESSOR:
 - DOP is affected by having too few files.
 - Chop the data up before compressing
 - Try to have files of similar size so the load doesn't go lop-sided



**Welcome to 2010
Let's Party Like It's 1999**

**We still think direct-attached storage is
for "small" systems**



We still think direct-attached storage is for "small" systems

- Well, that was then and this is now
- Think SAS
- There are very interesting SAS-attach storage options available
 - E.g., Storagetek 2530 Array
 - 12 drives, expandable to 48
 - Remember, balance!
 - Scan I/O patterns will deliver ~1.7GB/s from only 12 drives
 - 6x3Gb SAS attach ports
 - 1.8GB/s per unit
 - Connect several via x8 PCI and manage with ASM
 - ...that doesn't sound like "small" systems to me!



Welcome to 2010
Let's Party Like It's 1999

We still think commodity server means
“small”



We still think commodity server means “small”

- In 1999 there were no scalable commodity 8-processor x86 servers
 - Yes, I know about Profusion
- Cascades Xeon was 180nm process , 256 KB L2 and 133 MT!
- That was then this is now...
- Think Intel QuickPath (QPI) and AMD Hypertransport (HT)




We still think commodity server means “small”

- Think Nehalem EX or AMD Opterons
 - EX
 - 8 socket volume servers
 - 24 MB L3 with snoop filter
 - Single-hop memory
 - Up to 1TB DRAM
 - Gosh, ‘twas only Harpertown when phys addressable was 38 bit (256GB)
 - 4 memory lanes and 16 DIMM slots per socket
- That doesn’t sound that small to me and there will be >8 socket servers



**Welcome to 2010
Let's Party Like It's 1999**

**We still think database==structured and
file system==unstructured**

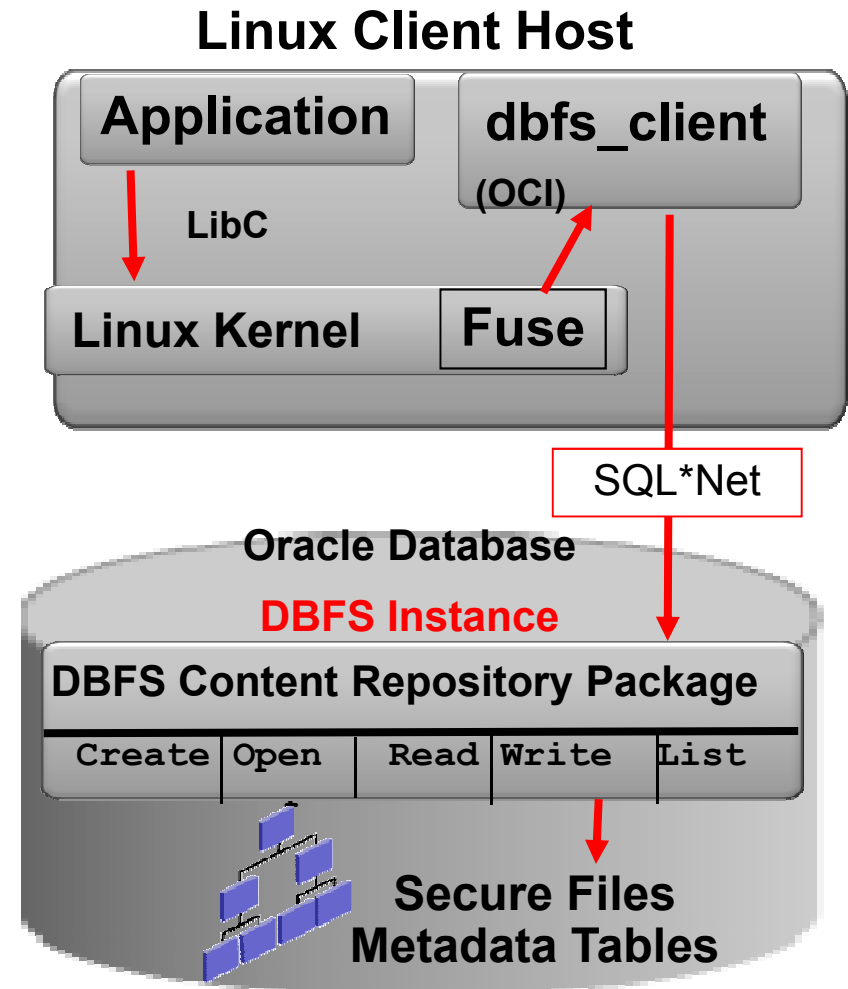



We still think database==structured and file system == unstructured

- Think Oracle Database 11g Database File System (DBFS)
 - A file system inside the database
 - All Oracle Database features apply
 - Data Guard!
 - DR for unstructured data regardless of storage
 - RAC
 - Multi-headed file serving with cache coherency
 - Compression, encryption, deduplication, RMAN, etc,etc,etc
- Access as a mounted file system with FUSE (file system in user space) on Linux

DBFS – Architecture Overview

- FUSE (**F**ile system in **U**ser**S**pace**E**)
 - An API and Linux Kernel module used to implement Linux file systems in user land.
- DBFS is a file system interface for SecureFiles
 - DBFS Content Repository implements a file server
 - PL/SQL package implements file calls
 - File create, open, read, list, etc.
 - Files are stored as Secure File LOBs
 - Directories and file metadata stored in tables/indexes
- Combining DBFS with FUSE offers mountable file systems
 - With RAC, DBFS is a scalable, distributed file system.



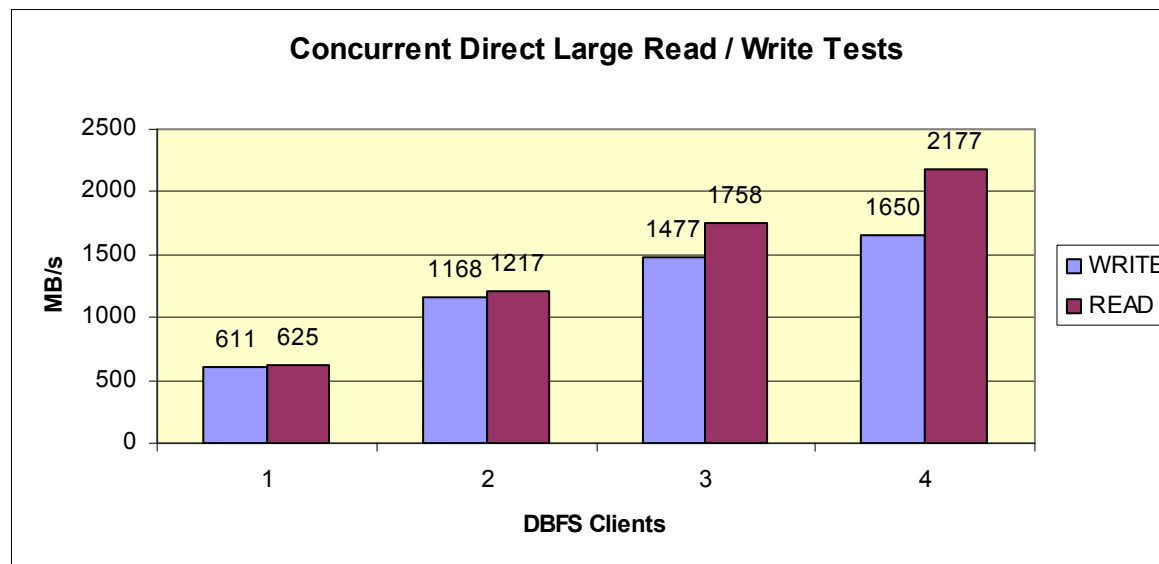


We still think database==structured and file system == unstructured

- That's nice, but does it perform?

DBFS – Performance Baseline Exadata V1 !

- 16 Concurrent dd(1) processes
 - 1MB read/write operations
 - DBFS file system mounted with direct I/O option
 - Each process reads or writes a 10 GB file
 - All files reside in one DBFS mount
- Process schematic: dd->libC->[fuse.ko](#)->dbfs_client [BEQ] ->shadow [iDB] ->Exadata



Exadata V2 ?

4 node:

4 GB/s write

6 GB/s read



Welcome to 2010
Let's Party Like It's 1999

We still think NUMA is niche technology



We Still Think NUMA is Niche Technology

- Well, it ain't...They're all pretty much NUMA system now, but...
- You probably don't have to care all that much
 - Modern Commodity NUMA is not the NUMA of yester-year
 - Local v Remote for Nehalem EP with QPI is ~20% tax!
 - Cache coherency damage is minimal
 - Large super-fast L3 caches
 - Snoop filters!
- NUMA optimizations in Oracle Database are disabled by default
- Oracle Database Machine best practice is to boot database hosts with `numa=off`
- ...and... nobody tests with Oracle Database 11gR2 NUMA optimizations...how do I *know* that?

Web [Images](#) [Videos](#) [Maps](#) [News](#) [Shopping](#) [Gmail](#) [more ▼](#)

[Web History](#) | [Search settings](#) | [Sign in](#)



[Advanced Search](#)

Web [+ Show options...](#)

Your search - **_enable_NUMA_support** - did not match any documents.

Suggestions:

- Make sure all words are spelled correctly.
- Try different keywords.
- Try more general keywords.

[Google Home](#) - [Advertising Programs](#) - [Business Solutions](#) - [Privacy](#) - [About Google](#)

ORACLE®



We Still Think NUMA is Niche Technology

- If nobody is Googling it, nobody is doing it
- But that doesn't mean you shouldn't test it. It's not like we're omniscient
 - NUMA optimization init.ora parameter changed from 11gR1 to 11gR2
 - `_enable_NUMA_optimizations` became `_enable_NUMA_support`
 - With Linux, libnuma is discovered and linked with `dlopen()` ,but..
 - It's looking for the library installed with the devnuma RPM
 - You have to symlink libnuma.so from `/usr/lib64` to `/usr/lib`
 - Look for proof of NUMA-awareness after instance boot
 - The SGA buffer pool is segmented into 1 per "node" (socket) so use `ipcs -m`
 - The alter log has NUMA-related strings showing the segmentation of the instance memory



**Welcome to 2010
Let's Party Like It's 1999**

**We still think NFS is a file serving
protocol**



We Still Think NFS is a “file serving” Protocol

- If it's a file serving protocol:
 - Why can I open with O_DIRECT (or directio(3c) on solaris)?
 - Why can I perform ~12,000 random 8K IOPS with measly Gigabit Ethernet?
 - Why can I scan datafile (PQO) in NAS via 10GbE at 1 GB/s?
 - Why can I scan 1.2 GB/s from NAS via TCPoIB ?
 - But what about CPU?
 - Consider:
 - Scanning NFS files (O_DIRECT) vi TCPoIB at 1060 MB/s costs only 2 Nehalem EP **threads**
 - Scanning 460 MB/s via 4xGbE costs only 3 Nehalem EP threads



Q&A



Welcome to 2010
Let's Party Like It's 1999

We Still Think Memory is Memory