



OracleWorld PARIS 2003

Session 40208

**Optimising Oracle® Database and RAC
Performance on UNIX® Servers**

Phil Harman

**Performance & Availability
Engineering**

Sun Microsystems™

**October 22,
2003**



Credits

OracleWorld SAN FRANCISCO 2003

Session 36698

Optimizing Oracle Database and RAC
Performance on UNIX Servers

glenn.colaco@sun.com

biswadeep.nag@sun.com

Background

- Performance & Availability Engineering
 - from single CPUs, to clusters of MP systems
 - from device drivers, to web browsers
 - and everything between!
- PAE is working with ...
 - Solaris™, Java™ & compiler engineering
 - CPU, systems, storage & network engineering
 - Sun field sales and support organisations
 - ISVs, IHVs, partners and end-users
- Years of close co-operation with Oracle

Introduction

- Many flavours of UNIX, most with unique performance features relating to Oracle
- How to make the best use of the available tools to observe and project performance?
- How to take corrective action once major performance issues are identified?
- How to extrapolate from past and present system performance when planning for future application demands?

*"Estimating which configuration will satisfy a particular requirement requires gazing into a **crystal ball** and accurately gauging the future behavior of, and interaction between, user populations, application software, system software, networks, and hardware platforms"*

– Brian L. Wong
*Configuration and Capacity
Planning for Solaris Servers*
ISBN 0-13-349952-9

Agenda

- Capacity Planning Concepts
- Handling Performance Data
- RAC Design and Performance
- Questions?

What is Capacity?

- The measure of a system's ability to process events over time
- e.g. customer service call centre
 - online 18 hours/day, 365 days/year
 - 10,000 inbound calls/day, including a midday peak of 40 calls/minute lasting two hours
 - 1,000 outbound calls/day
 - average call duration is 3 minutes

So what is Capacity Planning?

- Estimating the resources (e.g. people or systems) required to meet future needs
- A multidimensional problem:
 - space, time and cost constraints
 - utilisation and efficiency concerns
 - throughput vs latency trade-offs
 - balance of science and guesswork
- Especially important with grid and distributed systems

Capacity vs Performance

- **Capacity** is not the inverse of **Performance**
- **Performance** should be managed *but Capacity* should be planned
- **Performance** must be measured *before Capacity* can be planned
- The goal of **Capacity** Planning is to deliver an acceptable level of **Performance** in the future

Using the oracle's crystal ball

- Measure performance over time
 - Transactions
 - Processors
 - Memory
 - Storage
 - Network
- Trend historical data to project future system capability requirements
- But beware besetting bottlenecks!

Agenda

- Capacity Planning Concepts
- **Handling Performance Data**
- RAC Design and Performance
- Questions?

Find and Face the Facts

- Transactions
- Processors
- Memory
- Storage
- Network

Apples + Oranges per Second

- Real world transactions
 - always use your head!
 - isolate and characterise business transactions
 - determine workload mixes and data volumes
- Database transactions
 - e.g. use Oracle's Statspack
 - measure commit rates for various workloads
 - monitor for impending potential bottlenecks (e.g. latch contention)

Find and Face the Facts

- Transactions
- **Processors**
- Memory
- Storage
- Network

Do I have enough CPUs?

- How can I find out?
 - mpstat
- How are my CPUs performing?
 - cpustat
- How can I process more efficiently?
 - improving processor affinity
 - scheduler class and priority

What are those CPUs doing?

```
mpstat 5
```

CPU	minf	mjf	xcal	intr	ithr	csw	icsw	migr	smtx	srw	syscl	usr	sys	wt	idl
0	9	0	106	5795	5250	997	66	16	175	0	3283	54	45	0	0
1	9	0	2	2588	2355	1399	31	34	104	0	4583	70	28	2	0
4	21	0	1	17	0	1424	17	28	46	0	5931	82	18	0	0
5	18	0	2	15	0	1393	18	25	45	0	6015	87	13	0	0

How are those CPUs cooking?

cpustat

Cpu Cycles

	CPI	D Stall	I\$ Miss	Br Miss	IU Use	St Buf	RAW	MIPS	MInst/Tx
Total	4.52	2.31	0.96	0.12	0.15	0.37	0.07	3968	0.95
User	3.63	1.76	0.95	0.1	0.15	0.26	0.07	3125	0.74
Kernel	7.77	4.35	0.95	0.19	0.16	0.78	0.04	841	0.2

Cache Misses

	I\$ Miss	D\$ Miss	E\$ Miss	E\$ WB
Total	4.78	7.89	1.17	0.36
User	5.11	6.6	0.96	0.28
Kernel	3.79	12.63	1.98	0.64

Improving CPU efficiency

- Binding threads to processors
 - eliminates thread migration
 - doesn't exclude unbound threads
 - helps keep CPU caches warm
 - needs care or system can become unbalanced
- e.g. bind dedicated server 3608 to CPU 4
`$ pbind -b 4 3608`

Improving CPU efficiency

- Binding threads to processor sets
 - limits thread migration
 - excludes unbound threads
 - helps prevent CPU caches from getting cold
 - needs care or system can become unbalanced
- e.g. bind present and future dedicated servers to processor set 2

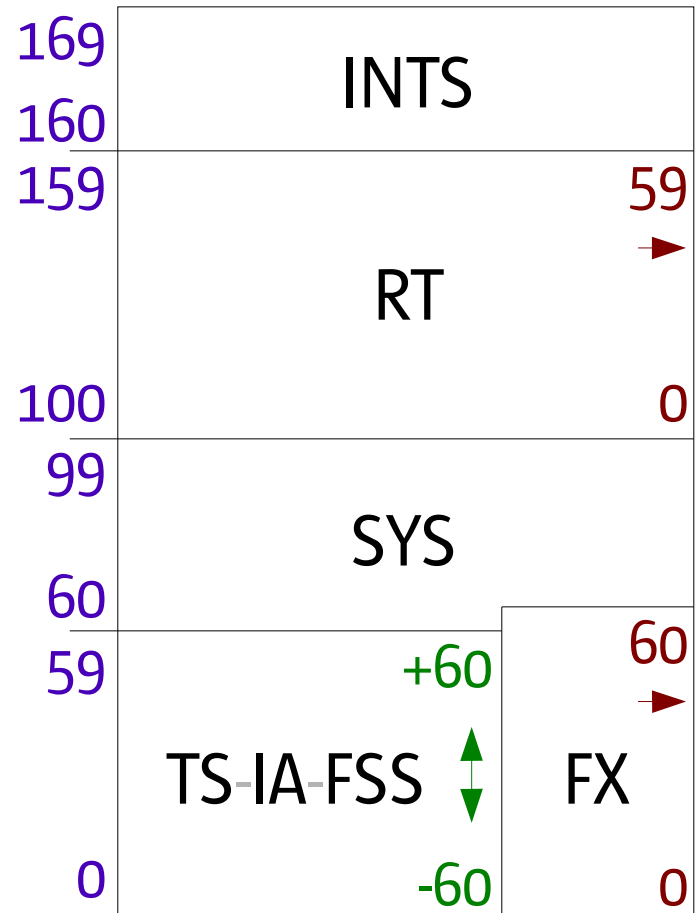
```
# psrset -b 2 \  
`pgrep -f 'tnslsnr|oracleTEST'`
```

Improving CPU efficiency

- Reduce process migrations
 - e.g. set `rechoose_interval=300` in `/etc/system`
- Improve locality for I/O interrupts
 - disable interrupts by processor or processor set
 - e.g. `psradm -i 10` or `psrset -f 2`
 - it is possible to contrive for the entire life-cycle of an I/O to be constrained to a single CPU
 - *Coming soon...* `intradm`

Solaris scheduler classes

- For user threads
 - timeshare (TS)
 - realtime (RT)
 - interactive (IA)
 - fixed (FX)
 - fareshare (FSS)
- For kernel threads
 - system (SYS)
 - interrupts (INTS)



Who gets priority?

- e.g. log writer in RT class (top)

```
# priocntl -s -c RT -p 59 \  
  `pgrep -f ora_lgwr`
```

- e.g. dedicated servers in FX class (high)

```
# priocntl -s -c FX -m 55 -p 55 \  
  `pgrep -f 'tnslsnr|oracleTEST'`
```

- e.g. RAC LMS processes in FX class (top)

```
# priocntl -s -c FX -m 60 -p 60 \  
  `pgrep -f ora_lms`
```

Find and Face the Facts

- Transactions
- Processors
- **Memory**
- Storage
- Network

How's your memory?

- Do I have enough memory?
 - vmstat
- Are my page sizes optimal?
 - trapstat
 - ISM / DISM
 - MPSS
 - pmap
- Where is my memory?
 - MPO

Make sure you're not paging!

```
vmstat 5
```

```
 kthr      memory          page          disk          faults        cpu
  r  b  w    swap    free re  mf  pi  po  fr  de  sr  s0  in   sy  cs  us  sy  id
52 40 0 117034 95723 0 25 0 0 0 0 0 0 5685 36466 25524 83 17 0
55 41 0 117032 95721 0 27 0 0 0 0 0 0 5614 36350 25473 83 16 0
34 35 0 117030 95719 0 30 0 0 0 0 0 0 5838 36448 25532 83 16 0
33 33 0 117028 95715 0 30 0 0 0 0 0 0 5824 36156 25593 84 16 0
44 30 0 117025 95712 0 32 0 0 0 0 0 0 5698 36273 25404 83 17 0
```

Does page size matter?

- Modern machines have lots of memory
 - e.g. upto 512 GB
- Oracle can use a large address space
 - i.e. Oracle SGA
- Large number of page table entries
 - # of entries = $512 \text{ GB} / 8\text{K} = 64 \text{ M}$
 - cached in translation lookaside buffer (TLB)
 - TLB miss is expensive, impacts performance

Measuring TLB misses

trapstat 5

```

cpu m| itlb %tim itsb %tim | dtlb %tim dtsb %tim | %tim
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
 0 u| 2047  0.1    0  0.0 |720519 15.9  378  0.1 |16.1
 0 k|   64  0.0    0  0.0 | 5642  0.2    2  0.0 | 0.2
-----+-----+-----+-----+-----+-----+-----+-----+
 1 u| 1654  0.1    0  0.0 |745764 16.6 1368  0.4 |17.1
 1 k|   38  0.0    0  0.0 | 1898  0.1    6  0.0 | 0.1
-----+-----+-----+-----+-----+-----+-----+-----+

```

Intimate Shared Memory

- Shared kernel page table structures
- Locked in physical memory (no paging)
- Solaris API
 - `shmat(shmid, shmaddr, SHM_SHARE_MMU)`
- Attempts to use allocate 4MB pages
 - # of entries = 512 GB / 4 M = 128 K
 - Fewer TLB misses, so better performance

Dynamic ISM

- Able dynamically to grow or shrink SGA
- Solaris API
 - `shmat(shmid, shmaddr, SHM_PAGEABLE)`
- `init.ora`:
 - `sga_max_size`
 - `db_cache_size`
- Oracle 9i / Solaris 9

Multiple Page Size Support

- Can use 8K, 64K, 512K or 4M pages for
 - stack
 - heap
- Can reduce TLB misses
- Examples

```
$ export LD_PRELOAD=mpss.so \  
    MPSSHEAP=4M MPSSTACK=512K
```

```
$ ppgsz -o heap=4M,stack=512K -p 3608
```

What page sizes did you get?

```
pmap -xs <pid>
```

Address	Kbytes	RSS	Locked	Pgsz	Mode	Mapped File
0000000100000000	8	8	-	8K	r-x--	oradism.modified
0000000100162000	16	16	-	8K	rwx--	[heap]
0000000380000000	4096	4096	4096	4M	rwxs-	[dism shmidx=0x64]
FFFFFFFF7F100000	8	8	-	8K	r-x--	libc.so.1
FFFFFFFF7FFFC000	16	16	-	8K	rw---	[stack]

total Kb	4144	4144	4096			

Does memory location matter?

- Physical memory at various distances from processors
- The further the memory, the longer the processor stalls
- Should avoid memory access hot spots

	CPI	D Stall	I\$ Miss	Br Miss	IU Use	St Buf	RAW	MIPS	MInst/Tx
Total	4.52	2.31	0.96	0.12	0.15	0.37	0.07	3968	0.95
User	3.63	1.76	0.95	0.1	0.15	0.26	0.07	3125	0.74
Kernel	7.77	4.35	0.95	0.19	0.16	0.78	0.04	841	0.2

Memory Placement Optimisation

- Memory/processors divided into **lggroups**
- Process with heap/stack in same lgroup
- Process affinity in terms of lgroups
- Per lgroup dbwriters
- `init.ora: _enable_NUMA_optimisation = True`
- **Solaris 9 / Oracle 10g**

Avoiding Hot Spots

cpustat

	CPU 0				CPU 1			
	B0	B1	B2	B3	B0	B1	B2	B3
Mrd/s	1.2	0.91	0.95	0.91	1.05	0.94	0.93	0.96
Mwr/s	0.31	0.27	0.28	0.3	0.3	0.28	0.25	0.31

- SGA round-robin across lgroups
- `/etc/system: kernel_cage_enable = 0`
 - disabling the kernel cage also disables DR
 - only worth considering on multi-board systems

Find and Face the Facts

- Transactions
- Processors
- Memory
- **Storage**
- Network

How's the I/O doing?

- Do I have enough I/O Capacity?
- Should I use Raw disk or Filesystems?
- 64 vs 32 bits - does it really matter?
- What about asynchronous I/O?

How busy are my disks?

```
iostat -nxz 5
```

```

                extended device statistics
  r/s    w/s    kr/s    kw/s  wait  actv  wsvc_t  asvc_t   %w   %b  device
  0.1    0.0     0.8     0.0   0.0   0.0    0.0    6.3    0    0  c0t4d0
 23.8    3.1   192.8    24.8   0.0   0.1    0.0    4.4    0   11  c8t3d0
 23.2    2.0   196.8    16.0   0.0   0.2    0.0    7.8    0   17  c6t24d0
 25.5    2.5   210.4    20.0   0.0   0.2    0.0    7.4    0   19  c5t7d0
 25.8    1.9   211.2    15.2   0.0   0.2    0.0    7.6    0   19  c1t24d0
 28.5    2.0   237.6    16.0   0.0   0.3    0.0    8.4    0   22  c3t33d0
  0.0   50.9     0.0  1180.8   0.0   0.2    0.0    3.6    0   13  c10t19d0
 32.8    3.1   265.6    24.8   0.0   0.2    0.0    4.7    0   16  c9t16d0
 30.5    1.8   258.4    14.4   0.0   0.3    0.0    8.8    0   23  c13t24d0
 32.4    3.0   270.4    24.0   0.0   0.3    0.0    7.7    0   24  c12t0d0
 25.0    3.2   200.0    25.6   0.0   0.1    0.0    4.7    0   12  c8t8d0

```

Raw disk or Filesystems?

- Performance vs Ease-of-Use
 - Direct I/O : best of both worlds?
- Filesystem single writer lock bottleneck
 - Concurrent Direct I/O
- DB may benefit from Solaris page cache
 - can waste memory (e.g. cached twice)
 - DB should have the best cache policies
- Consider mix and match

64 bits : Twice as good as 32!

- Increasing the database cache size
 - can significantly reduce I/O
 - read/write ratio will shift to writes
 - may improve transaction performance
- 32 bit Oracle cache limited to \ll 4GB
 - use filesystem to leverage the Solaris page cache
 - RAC may provide a solution for some workloads
- 64 bit allows huge database cache
 - all caching under Oracle's control
 - no wasteful secondary caching

Asynchronous I/O possibilities

- Basic async I/O is now on by default
 - aio_read() and aio_write() to submit
 - aio_wait() to reap
 - raw devices leverage Kernel Async I/O (KAIO)
- List I/O now available as an option
 - lio_listio() to submit
 - aio_waitn() to reap (aio_suspend() with Solaris 8)
 - `init.ora: _enable_list_io = TRUE`

Find and Face the Facts

- Transactions
- Processors
- Memory
- Storage
- **Network**

Is the net working?

- Am I killing my network interface?
- Is it about latency or throughput?

Am I killing my interface?

```
netstat -I <interface> 1
```

input		ce0			output			input (Total)		output	
packets	errs	packets	errs	colls	packets	errs	packets	errs	colls		
910751	0	573948	0	0	1862977	5999	8393573	0	0		
57	0	865	0	0	3125	0	4286	0	0		
46	0	1265	0	0	2904	0	4190	0	0		
160	0	872	0	0	2615	0	3115	0	0		

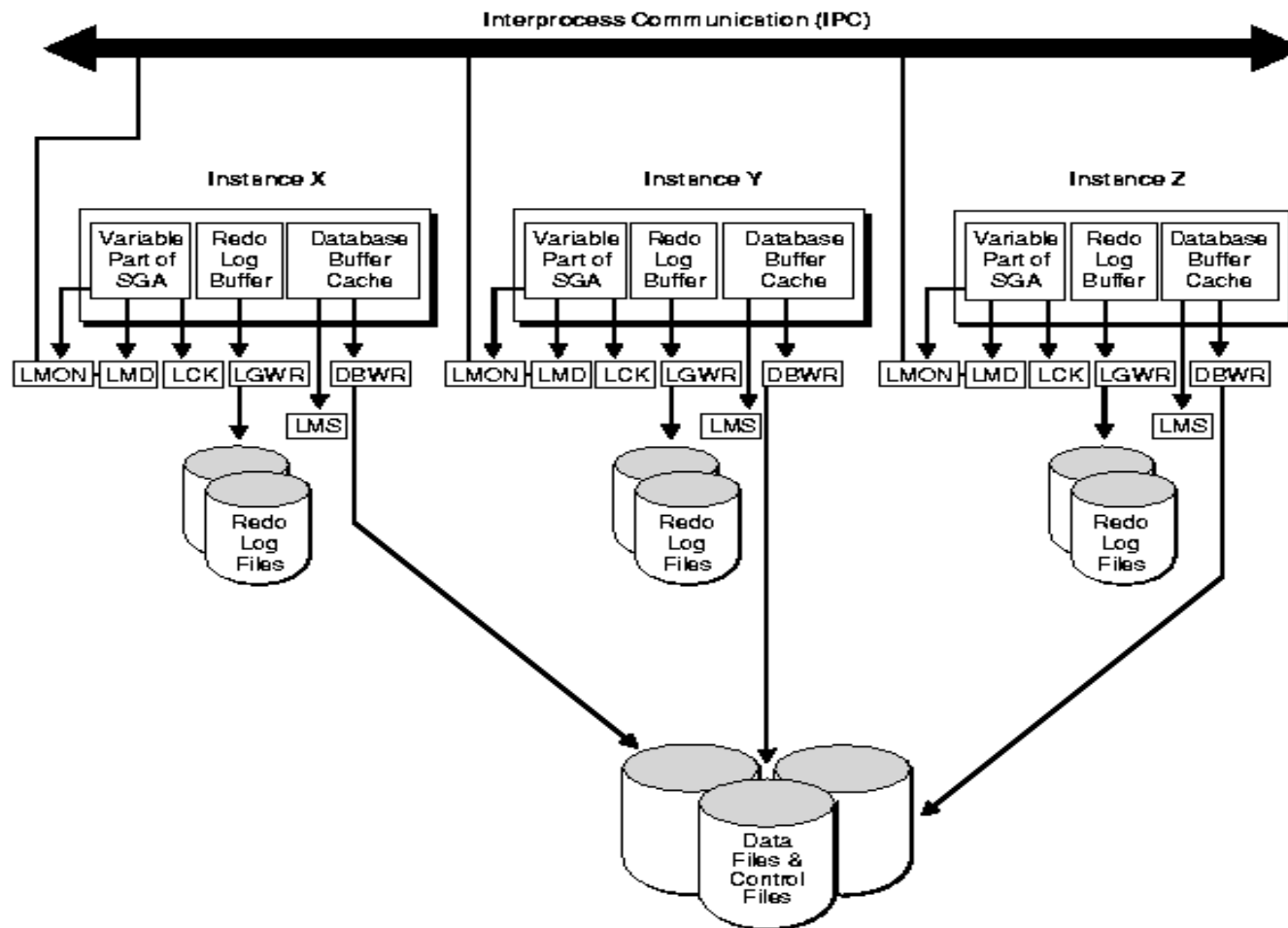
Latency or throughput?

- Latency
 - important for OLTP and small data movement
- Throughput
 - important for DSS and bulk data movement
- Use **snoop** to analyse network traffic
- Consider using Oracle MTS to avoid dedicated server creation/deletion

Agenda

- Capacity Planning Concepts
- Handling Performance Data
- **RAC Design and Performance**
- Questions?

Oracle RAC Architecture



Oracle RAC on Sun Cluster

- Solaris 9 / Sun Cluster 3.1
- Cluster interconnects and their hardware latencies
 - 100BaseT Ethernet 200 us
 - Gigabit Ethernet 60 us
 - SCI 4 us
 - Sun FireLink 1 us

Network Protocols

- **UDP** (default)
 - Supported on all interconnects
- Remote Shared Memory (**RSM**)
 - SCI & Sun FireLink
 - `init.ora: _disable_sun_rsm = FALSE`
- **uDAPL**
 - Coming soon in Solaris 10 / Oracle 10g

UDP/IP

- Light-weight connectionless protocol
- **No reliability guarantee**
 - Can drop packets under heavy load (buffer full)
 - Requires additional reliability layer
 - side channel messages
 - `init.ora: _side_channel_batch_timeout`
- Flow control necessary
- Limited scalability

RSM

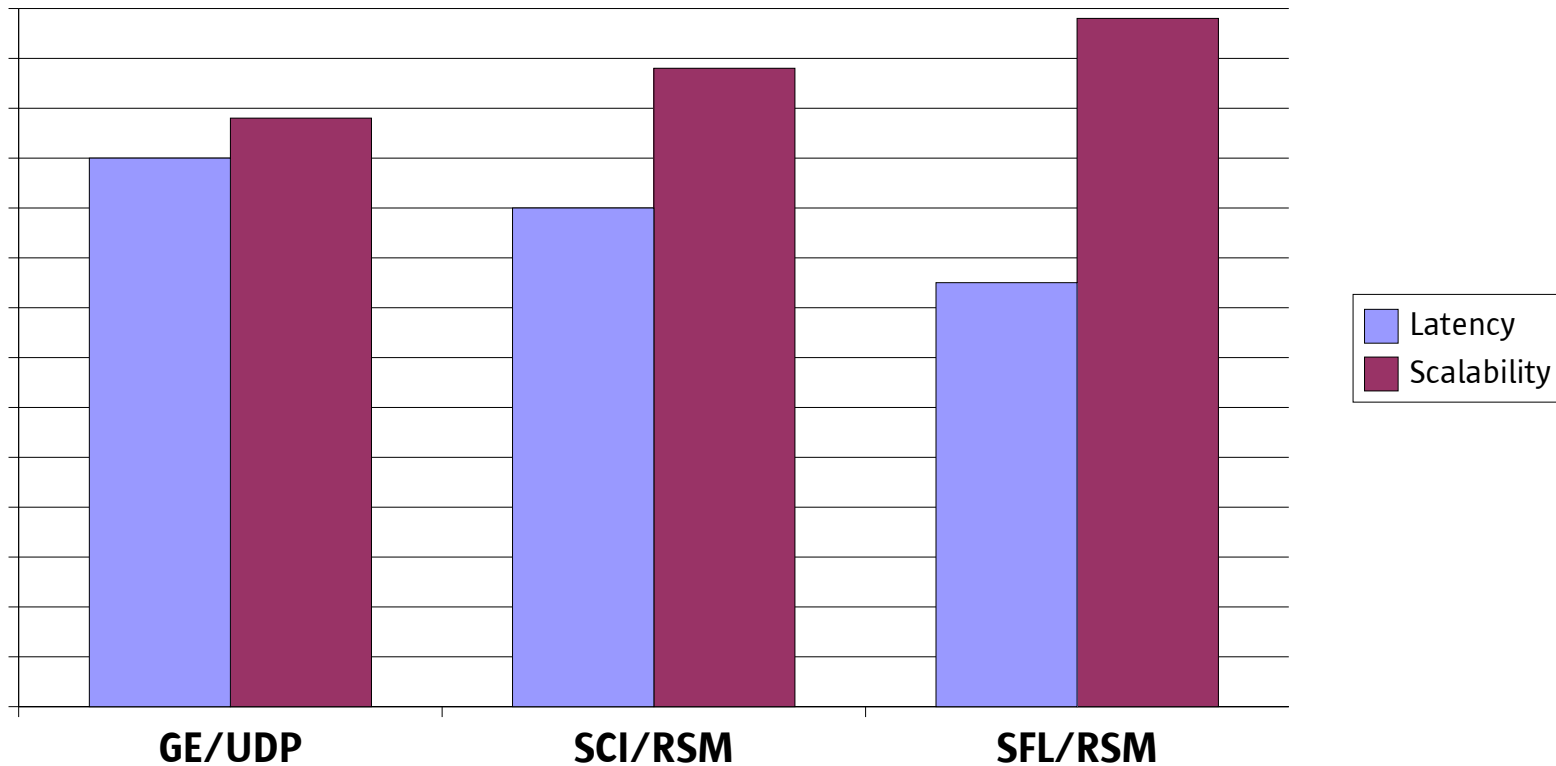
- **Memory shared between nodes**
 - exported by each node
 - mapped by other nodes
- **Bypasses operating system kernel**
 - user land data transfer
 - copy to remote memory
- Low overhead
- Better scalability
- Reliable protocol

Tuning for RSM

- Disable additional reliability layer
 - `init.ora: _reliable_block_sends = TRUE`
- Adjust the number of LMS processes
 - `init.ora: _lm_lms`
- Give increased priority to LMS
 - Scheduling delay contributor to latency
 - ```
priocntl -s -c FX -m 60 -p 60 \
`pgrep -f ora_lms`
```

# Latency vs Scalability

Latencies seen by Oracle vs resulting scalability



# RAC statistics

- From **Statspack**:
  - Global Cache Service (GCS) statistics
  - Global Enqueue Service (GES) statistics
  - Wait Events
  - Instance Activity statistics
  - Buffer Pool statistics
  - Load Profile

# GCS Statistics

| Statistic (in ms)             | GE   | SCI   | SFL  |
|-------------------------------|------|-------|------|
| CR block build time           | 0    | 0.1   | 0    |
| CR block flush time           | 0.1  | 0.15  | 0.1  |
| CR block send time            | 0.2  | 0.3   | 0.1  |
| CR block processing time      | 0.3  | 0.5   | 0.2  |
| CR block receive time         | 25.9 | 19.3  | 1.4  |
| Global cache get time         | 21.4 | 17.25 | 1.1  |
| Current block pin time        | 0.4  | 0.5   | 0.3  |
| Current block flush time      | 0    | 0.05  | 0    |
| Current block send time       | 0.2  | 0.3   | 0.1  |
| Current block processing time | 0.7  | 0.8   | 0.4  |
| Current block receive time    | 60.9 | 14.9  | 1.5  |
| Global cache convert time     | 37.4 | 26.7  | 1.25 |

# GES Statistics

| Statistic (in ms)        | GE  | SCI  | SFL  |
|--------------------------|-----|------|------|
| Global lock get time     | 3.7 | 3.05 | 0.35 |
| Global lock convert time | 0.1 | 0.15 | 0.1  |
| GES message process time | 0.1 | 0.15 | 0.1  |

# Message Statistics

| Statistic                     | GE   | SCI  | SFL  |
|-------------------------------|------|------|------|
| Message sent queue time       | 34.7 | 0.6  | 0.15 |
| % of flow controlled messages | 4.6  | 0.95 | 0.25 |
| GCS side channel messages     | 6.8  | 0    | 0    |

# RAC wait events

| Event                    | GE   | SCI  | SFL  |
|--------------------------|------|------|------|
| Global cache CR request  | 8750 | 6418 | 420  |
| Global cache s to x      | 3709 | 5020 | 277  |
| Buffer busy global cache | 284  | 302  | 65.5 |
| Enqueue                  | 572  | 618  | 31   |
| Global cache open x      | 134  | 701  | 40   |
| Global cache null to x   | 229  | 416  | 136  |
| Global cache busy        | 31   | 83   | 15   |
| Global cache null to s   | 42   | 53   | 11   |
| Buffer busy global CR    | 257  | 174  | 9    |
| Wait for msg sends       | 3016 | 8.5  | 4    |

# Workload Characteristics

## Reads vs Writes

| Per Transaction           | W1   | W2    | W3   | W4   | W5     |
|---------------------------|------|-------|------|------|--------|
| Logical Reads             | 196  | 90    | 2323 | 425K | 8K     |
| Block Changes             | 17   | 24    | 142  | 0.16 | 3.8K   |
| % Blocks changed per read | 8.72 | 26.84 | 6.11 | 0    | 47.23  |
| Physical Reads            | 0.32 | 1.24  | 1.72 | 6.78 | 0      |
| Physical Writes           | 0.22 | 1.69  | 1.73 | 0.02 | 193.25 |

# Buffer Cache Statistics

|                        | W1    | W2    | W3    | W4  | W5  |
|------------------------|-------|-------|-------|-----|-----|
| Buffer Hit % (Local)   | 99.8  | 97.6  | 99.9  | 100 | 100 |
| Global Cache Hit Ratio | 6.6   | 3.7   | 1.3   | 0.1 | 1.3 |
| Buffer Hit % (Overall) | 99.93 | 98.62 | 99.93 | 100 | 100 |

# Locality of Access

|                               | W1   | W2  | W3  | W4   | W5    |
|-------------------------------|------|-----|-----|------|-------|
| % msgs for buffer gets        | 6.35 | 2.7 | 1.2 | 0.1  | 3.9   |
| % remote buffer gets          | 3.7  | 1.1 | 1.1 | 0.1  | 0     |
| Ratio of local vs remote work | 0.85 | 2.5 | 0.2 | 0.3  | 19532 |
| Ratio of I/O for coherence    | 4.1  | 1.4 | 1.5 | 15.1 | 72    |

# Things to Consider

- Message latencies
- RSM vs UDP
- Sun Firelink vs SCI vs GE
- Data partitioning / locality
- Read vs write access
- Buffer hit ratio

*Total throughput matters more than scalability*

# Agenda

- Capacity Planning Concepts
- Handling Performance Data
- RAC Design and Performance
- Questions?

# For more information ...

phil.harman@sun.com  
glenn.colaco@sun.com  
biswadeep.nag@sun.com

*Please do fill out your session survey!*