Practical Performance Management
for Oracle RAC

Barb Lundhild RAC Product Management
Michael Zoll   RAC Development, Performance
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

- Oracle RAC Fundamentals and Infrastructure
- Common Problems and Symptoms
- Application and Database Design
- Diagnostics and Problem Determination
- Appendix
OBJECTIVE

• Convey a few simple and fundamental concepts of Oracle RAC performance
• Provide some simple sizing hints
• State that general system and SQL analysis and tuning experience is practically sufficient for Oracle RAC
• Note that problems can be identified with a minimum of metrics and effort
• Builds on similar presentation from OOW 2007

RAC Fundamentals and Infrastructure
**Oracle RAC Architecture**

- **Node 1**
  - Service
  - Listener
  - Instance 1
  - ASM
  - Oracle Clusterware
  - Operating System

- **Node 2**
  - Service
  - Listener
  - Instance 2
  - ASM
  - Oracle Clusterware
  - Operating System

- **Node n**
  - Service
  - Listener
  - Instance n
  - ASM
  - Oracle Clusterware
  - Operating System

- **Public Network**

- **Shared Storage**
  - Managed by ASM
    - Redo / Archive logs all instances
    - Database / Control files

- **RAW Devices**
  - OCR and Voting Disks

---

**Global Cache and Global Enqueue Service:**

**Functions and Processes**

- **SGA**
  - Library Cache
  - Dictionary Cache
  - Buffer Cache

- **Global Resource Directory**
  - Global Enqueue Service
  - Global Cache Service

- **Log buffer**

- **Cluster Private High Speed Network**

- **Oracle**

---
Global Buffer Cache

1. Local buffer cache scan
2. In local buffer cache? (Yes/No)
   - Yes: Access grant and Disk read
   - No: Global buffer cache scan
3. In global buffer cache? (Yes/No)
   - Yes: Transfer buffer
   - No: Access grant and Disk read

Data in Remote Cache: 2-way

1. Local Cache Miss
2. Datablock Requested
3. Remote Cache Hit
4. Datablock Returned
5. 2 hops
Data in Remote Cache: 3-way

- Local Cache Miss
- Datablock Requested
- 3 hops
- Datablock Returned
- Request forwarded
- Master/Directory Node
- Remote Cache Hit

Data On Disk

- Local Cache Miss
- Datablock Requested
- Disk Read
- Grant Returned
- Remote Cache Miss
Performance Model: Immediate Block Access

Message: ~200 bytes
200 bytes/(1 Gb/sec)

Block: e.g. 8K
8192 bytes/(1 Gb/sec)

Shadow process:
Initiate send and wait

1 Receive
2 LMS: Receive
3 Process block
4 Send

Total access time: e.g. ~360 microseconds (UDP over GBE)
- Network propagation delay: minor factor for roundtrip time if sufficient bandwidth
- Message processing: CPU and scheduling in Oracle and OS

Performance Model: Block Access with contention ("busy block")

Message: ~200 bytes
200 bytes/(1 Gb/sec)

Block: e.g. 8K
8192 bytes/(1 Gb/sec)

Shadow process:
Initiate send and wait

1 Receive
2 LMS
3 LGWR: Flush redo
4 Send

Total access time: e.g. ~360 microseconds transfer (UDP over GBE) +
- time for log flush and context switch, e.g. 3ms = 3.36 ms

ORACLE
Fundamentals: Minimum Latency (*), UDP/GbE and RDS/IB (2-way/3-way)

<table>
<thead>
<tr>
<th>Block size</th>
<th>2K</th>
<th>4K</th>
<th>8K</th>
<th>16K</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDP/GE</td>
<td>0.30</td>
<td>0.31</td>
<td>0.36</td>
<td>0.46</td>
</tr>
<tr>
<td>RDS/IB</td>
<td>0.12</td>
<td>0.13</td>
<td>0.16</td>
<td>0.20</td>
</tr>
</tbody>
</table>

(*) roundtrip, AWR and Statspack reports would report averages as if they were normally distributed, the session wait history which is included in Statspack in 10.2 and AWR in 11g will show the actual quantiles

The minimum values in this table are the optimal values for 2-way and 3-way block transfers, but can be assumed to be the expected values (I.E. 10ms for a 2-way block would be very high)

Sizing and Configuration Guidelines
**Private Interconnect**

- Network between the nodes of a RAC cluster MUST be private
- Bandwidth requirements depend on several factors
  - # of CPUs and CPU speed per node
  - # of nodes
  - Type of Workload: OLTP or DSS
- Large (Jumbo) Frames for GbE recommended
  - Avoids fragmentation and reassembly
- Interconnect bandwidth should be tested with non-Oracle utilities (e.g. iPerf)
  - No packet loss at 75% - 80% of bandwidth utilization

**Interconnect Bandwidth**

- Prediction based on upper bound heuristics
  - Large data messages (“blocks”) received can range from 1-8% of logical reads/sec
  - Small message rate approx.
    \[ ((\text{Total physical reads per sec}) \times (N-1)/N) \times 4 + \text{blocks per sec} \times 3 \]

- EXAMPLE for bandwidth calculation
  - Predicted database load per node
    - Physical reads per node: 10000/sec
    - Logical reads per node: 100000/sec
  - Estimated messages per node
    - 1000 – 8000 blocks/sec * 2 (receive and send)
    - 33000 – 54000 messages/sec
    - 35000 – 70000 total messages/sec
  - Estimated bandwidth utilization
    - 22.6MB/sec – 74.8 MB/sec
**Interconnect Bandwidth**

- Generally, 1Gb/sec sufficient for performance and scalability in OLTP.
  - Typical link utilization approx. 10-30% in OLTP
  - NIC bonding and load balancing recommended
  - 10GbE possible, but not much experience with RAC
- DSS/DW systems should be designed with > 1Gb/sec capacity
- Buffer overflows, congested links and flow control can have severe consequences for performance
- A sizing approach with rules of thumb is described in
  - Project MegaGrid: Capacity Planning for Large Commodity Clusters ([http://otn.oracle.com/rac](http://otn.oracle.com/rac))

**IPC configuration**

```
Applications: Oracle
Protocol processing: IP, UDP
Device Drivers
NIC1
NIC2
Switch
Queues
Ports
```

- Socket buffer overflows: rmem_default, rmem_max
- Packet reassembly failures
- STREAMS queues overflows: sq_max
- Transmission and receive queue discards
- Memory allocation issues: netdev_max_backlog
Configuration Hints

- On systems with a large number of CPUs or NUMA systems,
  - Direct NIC interrupts to specific CPUs
  - Bind LMS to those CPUs (or NUMA boards)
- Important Link, NIC and IPC Settings
  - Negotiated top bit rate and full duplex mode
  - UDP receive and send buffers
- CVU does checking

Block Server (LMSx) Processes

- Block access latencies can increase when CPU(s) are busy and run queues are long
- Fewer and busier LMS processes may be more efficient.
  - monitor their CPU utilization
  - the default is good for most requirements
  - LMS processes usually top CPU consumers
    - 70-80% busy per LMS is ok
- Higher priority for LMS is default
  - The implementation is platform-specific
Block Server (LMSx) Processes

- Oracle RAC 10g (10.2.0.4)
  - # of CPUs | # of LMS
  - <=2 | 1
  - <=12 | 2
  - >=12 | #cpu/4

- Oracle RAC 11g
  - # of CPUs | # of LMS
  - <=4 | 1
  - <=16 | 2
  - >=16 | #cpu/4

- An additional LMS for every 32 CPUs

- gcs_server_processes overrides defaults
- Defaults are good for most requirements

Block Server (LMSx) Processes

- SCHED_RR on most platforms
  - Real time priority
  - SCHED_NOAGE = 178 on HP/UX (10.2.0.4)
- High priority can cause starvation when the number of LMS processes exceeds the number of available CPUs
  - Multiple databases per cluster
  - ASM instance
- _os_sched_high_priority = 0
  - Turns off SCHED_RR, i.e., real-time priority for LMS
Application and Database Design for Scalability in RAC

General Principles

- No fundamentally different design and coding practices for RAC
- BUT: flaws in execution or design have higher impact in RAC
  - Performance and scalability in RAC will be more sensitive to bad plans or bad schema design
- Standard SQL and schema tuning solves > 80% of performance problems
Scaling and Workloads

- DML intensive
  - Scale well, if care is taken to avoid bottlenecks
- Read-intensive
  - Scale well

DML Contention and Serialization

```
INSERT ... WHERE Key = sequence
Other DML/modification intensive operations on small set of
cached blocks
```

"busy blocks"
Read-intensive

Eventually all blocks cached, Larger read cache

No messages in 11g

Scalability: Hot Spots

- Serialization on frequently modified blocks has higher cost in RAC
  - INSERT into indexes with right growing characteristics
  - UPDATES of data in the same block
  - Segments without ASSM or Freelist Groups
- A small set of blocks is concurrently modified by all instances
  - Processes queue for “busy” buffers
  - Redo may have to be written before sending data
- Application Data Partitioning (Range, Hash) and routing per partitioning key in application server tier
  - For high performance
Best Practices for Predictable Scalability

- Automatic Segment Space Management required
- Indexes with right-growing characteristics have monotonically increasing keys E.G. DATE columns or keys generated by sequence numbers
  - Cache sequence numbers per instance
  - Hash or range partitioned table with LOCAL indexes
- Frequent updates of “small” and compact tables
  - “small”=fits into a single buffer cache
  - Reduce block size (2K) and row density of blocks (PCTFREE 99)
- Application Modules with such characteristics could also be directed to particular nodes
  - Use cluster managed services as early as possible

Scalability: CPU cost for remote block copies

- A small set of blocks is randomly modified and frequently read
  - UPDATE followed by frequent SELECT
  - INSERTS into index with non-sequential keys
- Traffic across the interconnect higher
- CPU cost for messaging increases but latency low
- Scales as long as CPU is available
- In 11.1, CPU cost is further reduced by data compression and aggregation of message
**Migration to RAC and RAC scalability**

- Rough database design review points out bottlenecks
  Look for:
  - Indexes with right-growing characteristics
  - Monotonic, ordered sequences
  - Frequently updated and read “small” tables with high row density
  - Single instance AWR or Statspacks report
    - “buffer busy” waits indicate contention, may not scale in RAC
    - Index contention waits in segment statistics can point to hot spots in index
    - Row lock waits in segment statistics identify design that may not scale

**Load Control**

*Ensure capacity when failures*

<table>
<thead>
<tr>
<th>Instance failure</th>
<th>Instance recovery</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous load level before failure</td>
<td>New load level after failure</td>
</tr>
</tbody>
</table>

Listener throttling can mitigate connection storms via rate control
Leverage Connection Pools
Oracle UCP

Performance Health Checks:
Metrics and Method using AWR
Quick Primer of Global Cache Events

- Time-based events for global cache (gc)
  - Gc cr/current block 2-way/3-way
    - Single immediate block get via interconnect
  - Gc cr/current block busy
    - Single block get via interconnect delayed by log flush
  - Gc cr multi block request
    - Contiguous read via interconnect
  - Gc buffer busy
    - Waiting behind one of the foregoing events
  - Gc cr/current grant 2-way
    - Access granted to data, followed by disk read
  - Gc cr/current block lost
    - Message was dropped
- Often related to other events in terms of causality
  - E.g. gc current block busy -> log file sync/log file parallel write

Global View of Metrics

Note: Cause and effect are distributed
Example: Review AWR

Top 5 Timed Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Waits</th>
<th>Time(s)</th>
<th>AVG (ms)</th>
<th>%Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>db file sequential read</td>
<td>2,627,295</td>
<td>21,808</td>
<td>8</td>
<td>43.2%</td>
</tr>
<tr>
<td>CPU time</td>
<td>9,156</td>
<td></td>
<td></td>
<td>18.2%</td>
</tr>
<tr>
<td>gc current block 3-way</td>
<td>3,289,371</td>
<td>4,019</td>
<td>8</td>
<td>8.0</td>
</tr>
<tr>
<td>gc buffer busy</td>
<td>373,777</td>
<td>3,272</td>
<td>9</td>
<td>6.5</td>
</tr>
<tr>
<td>gc current block 2-way</td>
<td>3,982,284</td>
<td>3,192</td>
<td>9</td>
<td>6.3</td>
</tr>
</tbody>
</table>

Data received from other instances: Faster than IO

Example: Performance Health Check

- IO time for random IOs (43% of wait time)
  - Avg read time: ok -> NO ACTION
- Global Cache (20% of wait time)
  - Avg block access time: ok -> NO ACTION
  - Some serialization, but not a big problem -> INVESTIGATE
Example: Investigate Serialization

<table>
<thead>
<tr>
<th>gc buffer busy</th>
<th>373,777</th>
<th>3,272</th>
<th>9 ms</th>
</tr>
</thead>
</table>

Waits for

gc current block busy | 125,595 | 2,931 (Not OK!)

Global Cache Transfer Stats

<table>
<thead>
<tr>
<th>Inst Block</th>
<th>Blocks</th>
<th>%</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Class</td>
<td>Received</td>
<td>Immed</td>
<td>Busy</td>
</tr>
<tr>
<td>4 data block</td>
<td>114,426</td>
<td>95.9</td>
<td>4.1</td>
</tr>
<tr>
<td>7 data bloc</td>
<td>162,630</td>
<td>76.6</td>
<td>23.4</td>
</tr>
</tbody>
</table>

Log file IO

Avg global cache current block flush time (ms): 3.7

Example: Analysis

- Serialization is likely to be due to log flush time on another node
  - 3-4 ms for log file IO
  - Some data is modified frequently and changes may not have been logged before data blocks are “pinged” out of cache

- Reason for serialization (“busy”) is normally
  - Block transfers which require a log flush by the server
  - Immediate block transfers which take a long time
  - A queue of waiters forms while the transfer is in progress
  - Analogous to buffer waits while disk IO is in progress
Example: Segment Statistics

Segments by Global Cache Buffer Busy
ES_BILLING    TABLE    97.41 %

Segments by Current Blocks Received
ES_BILLING    TABLE    85.81 %

ANALYSIS: TABLE ES_BILLING is frequently read and modified on all nodes. The majority of global cache accesses and serialization can be attributed to this.

Performance Health Check: Method

• Check events with highest % of wait time ( "Top 5")
• Check avg access times
  • If higher than normal, look for reasons
• Check for serialization ( "busy" events )
  • Find instance(s) which transfer highest % of busy blocks
  • Check avg log flush time and log sync time on those instances
  • Find tables and indexes with highest serialization
• Run advisories on the analyzed snapshots
Common Problems and Symptoms

- Interconnect or Switch Problems
- Slow or bottlenecked disks
- System load and scheduling
- Unexpectedly high latencies
### Symptoms of an Interconnect Problem

#### Top 5 Timed Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Waits</th>
<th>Time(s)(ms)</th>
<th>Time</th>
<th>Wait Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>log file sync</td>
<td>286,038</td>
<td>49,872</td>
<td>174</td>
<td>41.7</td>
</tr>
<tr>
<td>gc buffer busy</td>
<td>177,315</td>
<td>29,021</td>
<td>164</td>
<td>24.3</td>
</tr>
<tr>
<td>gc cr block busy</td>
<td>110,348</td>
<td>5,703</td>
<td>52</td>
<td>4.8</td>
</tr>
<tr>
<td>gc cr block lost</td>
<td>4,272</td>
<td>4,953</td>
<td>1159</td>
<td>4.1</td>
</tr>
<tr>
<td>cr request retry</td>
<td>6,316</td>
<td>4,668</td>
<td>739</td>
<td>3.9</td>
</tr>
</tbody>
</table>

*Should never be here*
*Always a severe performance problem*

### Interconnect or IPC problems

- **Application:** Oracle
- **Protocol processing:** IP, UDP
- **UDP:** Packet receive errors, Socket buffer overflows
- **IP:**
  - 1201 Fragments dropped after timeout
  - 467 Reassembly failure
  - 468 Incoming packets discarded
- **TX errors:** 135 dropped, overruns:
- **RX errors:** 0 dropped, 27 overruns
- **Below:**
  - **NIC:** NIC1, NIC2
  - **Ports**
  - **Switch**
  - **Queues**

---

*ORACLE*
Probable Causes of Interconnect Problems

- Network Congestion
  - discarded packets/fragments
  - Packet reassembly failures or timeouts
  - Buffer overflows
  - Flow control
- Checksum errors or corrupted headers

$netstat -s
Ip:
84884742 total packets received
1201 fragments dropped after timeout
3384 packet reassembles failed

Probable Causes of Interconnect Problems : NIC or driver errors

- Checksum errors or corrupted frames
- Memory allocation, Queues full

ifconfig -a:
eth0 Link encap:Ethernet  HWaddr 00:0B:DB:4B:A2:04
  inet addr:130.35.25.110  Bcast:130.35.27.255  Mask:255.255.252.0
  UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
  RX packets:21721236  errors:135  dropped:0  overruns:0  frame:95
  TX packets:273120  errors:0  dropped:27  overruns:0  carrier:0
  ...
Global Cache Lost Block

- Almost always an IPC configuration (buffering and flow control) or downstream (switch etc.) network problem
  - Downstream problems are hard to debug
- Get OS data immediately: ifconfig, netstat
- Detection Time in 10.2.0.4 and 11g reduced
  - 500ms (around 5 secs in 10.2.0.3)
  - can be lowered if necessary
- Cr request retry event related to lost blocks
  - It is highly likely to see it when gc cr blocks lost show up

Cluster-Wide Disk I/O Impact

Node 1

<table>
<thead>
<tr>
<th>Top 5 Timed Events</th>
<th>Avg %</th>
<th>Total</th>
<th>Event</th>
<th>Calls</th>
<th>Time(s)</th>
<th>Time(ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>log file sync</td>
<td>286,038</td>
<td>49,872</td>
<td>174</td>
<td>41.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gc buffer busy</td>
<td>177,315</td>
<td>24,021</td>
<td>164</td>
<td>24.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>gc cr block busy</td>
<td>110,348</td>
<td>5,703</td>
<td>52</td>
<td>4.8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. IO on disk group containing redo logs is slow
2. Block shipping for frequently modified blocks is delayed by log flush IO
3. Serialization builds up

CAUSE: Expensive Query in Node 2
Causes IO bottleneck
Node 2

Load Profile

<table>
<thead>
<tr>
<th>Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redo size:</td>
</tr>
<tr>
<td>Logical reads:</td>
</tr>
<tr>
<td>Physical reads:</td>
</tr>
</tbody>
</table>
**Node 2: An IO capacity problem**

<table>
<thead>
<tr>
<th>Event</th>
<th>Waits</th>
<th>Time(s) (ms)</th>
<th>Avg %Total</th>
<th>Wait Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>db file scattered read</td>
<td>3,747,683</td>
<td>368,301</td>
<td>98</td>
<td>User I/O</td>
</tr>
<tr>
<td>gc buffer busy</td>
<td>3,376,228</td>
<td>233,632</td>
<td>69</td>
<td>Cluster</td>
</tr>
<tr>
<td>db file parallel read</td>
<td>1,552,284</td>
<td>225,218</td>
<td>145</td>
<td>User I/O</td>
</tr>
<tr>
<td>gc cr multi block request</td>
<td>35,588,800</td>
<td>101,888</td>
<td>3</td>
<td>Cluster</td>
</tr>
<tr>
<td>read by other session</td>
<td>1,263,599</td>
<td>82,915</td>
<td>66</td>
<td>User I/O</td>
</tr>
</tbody>
</table>

**Symptom of Full Table Scans**

- Disk IO slow due to rogue query
- Slow concurrent scans causes buffer cache contention
- Multiple symptoms for the same cause

---

**CPU Saturation or Long Run Queues**

<table>
<thead>
<tr>
<th>Event</th>
<th>Waits</th>
<th>Time(s)</th>
<th>Avg %Total</th>
<th>Wait Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>db file sequential read</td>
<td>1,312,840</td>
<td>21,590</td>
<td>16</td>
<td>User I/O</td>
</tr>
<tr>
<td>gc current block congested</td>
<td>275,004</td>
<td>21,054</td>
<td>77</td>
<td>Cluster</td>
</tr>
<tr>
<td>gc cr grant congested</td>
<td>177,044</td>
<td>13,495</td>
<td>76</td>
<td>Cluster</td>
</tr>
</tbody>
</table>
### High Latencies

<table>
<thead>
<tr>
<th>Event</th>
<th>Waits</th>
<th>Time (s)</th>
<th>AVG (ms)</th>
<th>% Call Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>gc current block 2-way</td>
<td>201,663</td>
<td>4,063</td>
<td>20</td>
<td>13.4</td>
</tr>
<tr>
<td>gc cr block 2-way</td>
<td>317,062</td>
<td>5,767</td>
<td>18</td>
<td>19.0</td>
</tr>
</tbody>
</table>

Unexpected: To see > 1 ms  (AVG ms should be around 1 ms)

- Check network configuration (private? bandwidth?)
- Check for high CPU consumption
  - Runaway or spinning processes

### Hints and Tips

Look for:
- Unexpected Events
- Unexpected “Hints” (“busy”, “congested”)
- Unexpected high avg
- Maintain “global” perspective
  - Symptom and cause are often not on the same instance
- Execute top down and assume simplest case:
  - The top wait is usually the suspect
  - Gc events look scary but are often secondary effects.
Transient Problems and Hangs

Temporary Slowness and Hang

- Can affect one or more instances in cluster
- Can be related
  - IO issues at log switch time (checkpoint or archiver slow)
  - Process stuck waiting for IO
  - Connection storm
- Hard to establish causality with AWR statistics
- Active Session History
Active Session History

- In 10.2, can identify local blocker for a hang
- In 11g, can identify global blocker

Accessing ASH data

- Dump to trace file
- V$ACTIVE_SESSION_HISTORY
- DBA_HIST_ACTIVE_SESS_HISTORY
- ASH report
- EM Diagnostic Pack
Dumping ASH to file

> oradebug setmypid
> oradebug dump ashdump 10

> alter session set events 'immediate trace name ashdump level 10';

- 10 ==> minutes of history you want to dump
- Caveat: no global dump for cluster
- Generated file can be loaded into database using supplied control file rdbms/demo/ashldr.ctl

Temporary Slowness or Hang

Slowdown from 5:00-5:30
Spike in waiting sessions

Active Session History Dump Queried

Sessions begin to slow down waiting for space
Active Session History Dump Queried

All instances are affected similarly

Procedures for transient slowdowns and hangs

- On all instances
  - Produce Active Session History (ASH) report
  - Dump in-memory ASH data and load into database
  - Narrow down to time interval with slowdown
Conclusions

• Anyone can deal with Oracle RAC
• Use Occam’s razor:
  • SQL and Schema tuning solves most issues
  • Efforts in testing, configuring and sizing pay off
  • Load control is important
• Any node could be the culprit for a problem, follow the trail the AWR reports
QUESTIONS

ANSWERS

ORACLE

ORACLE