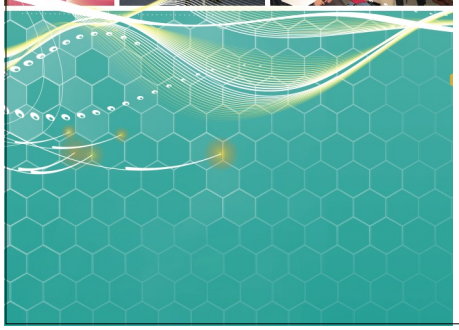
Today's Topic:



Resolving the Free Buffer Waits Event

Craig A. Shallahamer
OraPub, Inc.
craig@orapub.com

OraPub presentations at OOW.

- Sunday, 3pm
  - Resolving the Free Buffer Waits Event
    - S318617, Moscone West, L2, room 2005
- Monday, 11am
  - Optimizing Internal Serialization Control
    - Unconference, Hotel Parc 55, Lombard room
- Wednesday, 1pm
  - Evaluating the Anticipated Effect of Performance Sols.
    - Unconference, Hotel Parc 55, Lombard room
- Thursday, 1:30pm
  - Quantifying Oracle Database System Performance
    - S315724, Moscone South, room 302

 Resolving FBW

OraPub is about Oracle performance.

- OraPub is all about Oracle performance management; monitoring, firefighting, quantitative and predictive analysis.
- Web site started in 1995 and the company was founded in 1998 by Craig Shallahamer.
- OraPub has always been about disseminating Oracle database centric technical information.
- Consulting, training, books, papers, and products are now being offered.
- We have been on-site in 24 countries and our resources have been received in probably every country where there are DBAs.

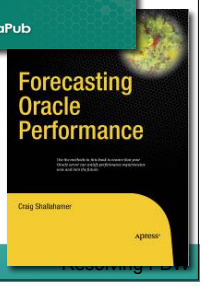
Resources

- Free Tools
- Free Papers
- Books
- Products
- Consulting
- Training

 Resolving FBW

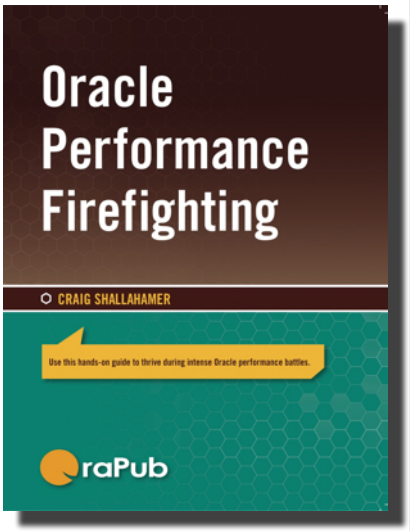
Short resume.

- Studies economics, mathematics, and computer science at university in California, US.
- Started working with Oracle technology in 1989 as a Forms 2.3 developer on Oracle version 5.
- Soon after started performance firefighting...daily!
- Co-found both Oracle's Core Technology and System Performance Groups.
- Left Oracle to start OraPub, Inc. in 1998.
- Authored 24 technical papers and worked in 24 countries.
- Authors and also teaches his classes ***Oracle Performance Firefighting***, ***Adv Oracle Performance Analysis***, and ***Oracle Forecasting & Predictive Analysis***.
- Authored the books, *Forecasting Oracle Performance*, and his new book, ***Oracle Performance Firefighting***.



If you want my firefighting book…

- Retail $49.95
- Amazon $69.95
- OraPub normal $39.95
- “OOW” disc. $29.96
- Cash in hand $30.00



raPub

 Resolving FBW

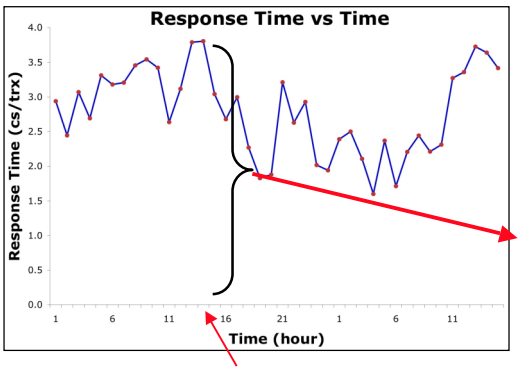
Our real objective tonight.

Learn how to resolve the “perfect storm” wait event “free buffer waits” by delving into Oracle internals and applying a rigorous time-based diagnostic and analysis method.

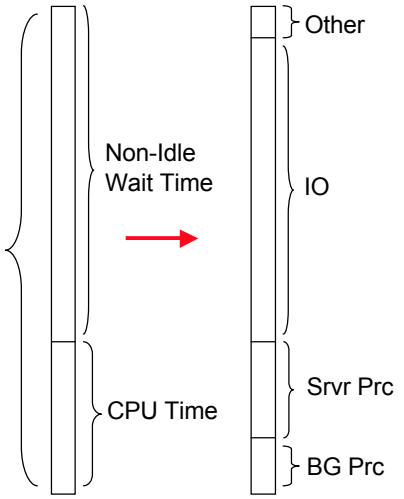
raPub

 Resolving FBW

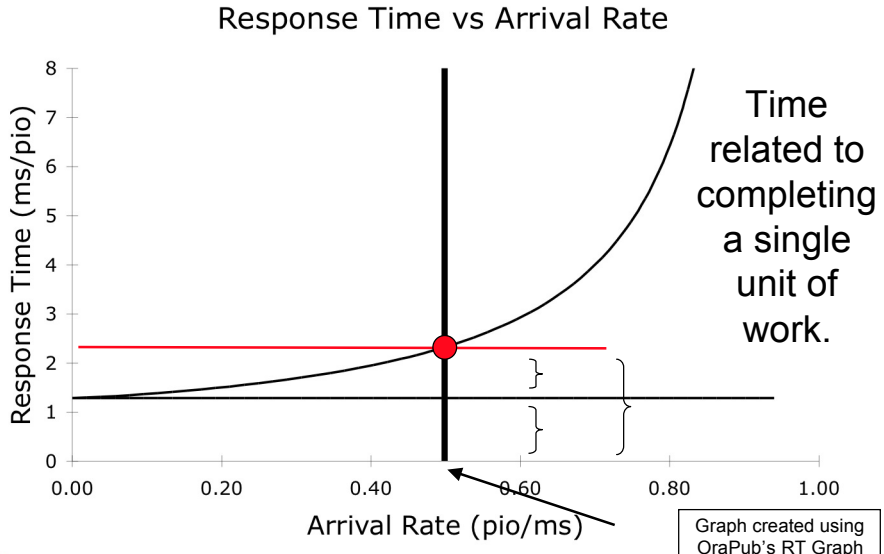
Representing meaningful time.



Time related to **all** the processed work over a specific period of time.



The response time curve.



Oracle RTA numeric classification.

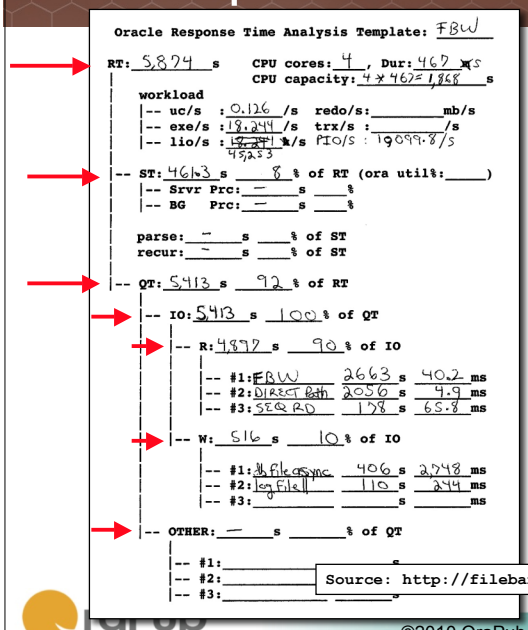
- While not visually as dazzling, numerically classifying response time provides many benefits, such as:
  - The entire performance situation is quantified providing inputs into advanced analysis.
  - The classification naturally results in diagnosis.
  - The performance problem is highlighted.
  - Provides strong clues as to the operating system situation.
  - Makes *before* and *after* analysis straightforward.
  - Lends itself towards systematic diagnosis.
  - Allows for intelligent classification of time.
  - Can be used for casual and introductory work as well as detailed and advanced analysis.

How do you know?

```
SQL> @rtpctx
Database: prod18                                          04-MAY-10 11:22am
Report:   rtpctx.sql          OSM by OraPub, Inc.          Page         1
                  Response Time Activity (467 sec interval)

                                          Time            Avg Time     Wait
RT Component                              (sec)    % RT Waited (ms) Count(k)
----------------------------------- ---------- ------- ----------- --------
free buffer waits                     2663.010   37.82      40.218       66
direct path read                      2055.780   29.20       4.881      421
CPU consumption: Oracle SP + BG procs  461.302    6.55       0.000        0
db file async I/O submit               406.800    5.78    2748.649        0
db file sequential read                216.220    3.07       5.772       37
read by other session                  178.170    2.53      65.818        3
log file parallel write                110.440    1.57     244.336        0
. . .
Instance Workload Statistic          Interval Work Interval Work/Sec
-------------------------------------- ------------- -----------------
user calls                                       59             0.126
session logical reads                      21132973         45252.619
physical reads                              4716617         10099.822
physical read IO requests                    327596           701.490
execute count                                  8520            18.244
```

Source: OSM Toolkit

A simple Oracle RTA template.



Notice the response time components are ready for entry based upon an Oracle performance statistics.

Hence the term, Oracle Response Time Analysis or ORTA for short.

Source: http://filebank.orapub.com/perf_stats/RTA_Template.pdf

How do you know, with `rtsysx.sql`?

```
SQL> @rtsysx 120 50

OraPub's Response Time Analysis (RTA) interactive system level delta report

Initializing response time delta objects...
Sleeping and probing active SQL for next 120 seconds...
Done sleeping...gathering and storing current values...

*** Response Time Ratio and Workload Metrics

RT Ratio  Ora Trx/s Block Changes/s User Calls/s      Execs/s
-------- ---------- ---------------- ------------ ------------
   0.967       1.30         12051.93         0.57        38.18

*** Response Time System Summary (delta - interactive - system level)

   Tot CPU    CPU SP    CPU BG CPU Parse CPU Recur          Tot Wait  IO Wait Other Wait
      Time      Time      Time      Time      Time Ora CPU      Time      Time       Time         %          %
     (sec)     (sec)     (sec)     (sec)     (sec)  Util %     (sec)     (sec)      (sec) IO Wait Other Wait
---------- --------- --------- --------- --------- ------- --------- --------- ---------- ------- ----------
       168       165         2         0       199    34.9     4,896     4,775        121      98          2
```

How do you know, with rtsysx.sql?

```
*** I/O Wait Time Summary w/Event Details (delta - interactive - system level)

 IO Wait  IO WRITE    IO READ
    Time Wait Time  Wait Time  % IO % IO
   (sec)      (sec)      (sec) Write Read
-------- --------- ---------- ----- ----
   4,775     2,891      1,887    61   40

                                                                Tot Call    Avg Call
                                                               Wait Time   Wait Time
IO Wait Event                                          R,W     %     (sec)        (ms) Tot Waits
------------------------------------------------------------ --- ----- ---------- ---------- ---------
free buffer waits                                      W      49    2,326.90       29.56    78,722
direct path read                                       R      35    1,668.53        8.70   191,708
db file async I/O submit                               W       6      308.23     1802.51       171

*** Other Wait Time (non-I/O) Event Detail (delta - interactive - system level)

                                                                    Tot Call    Avg Call
                                                                   Wait Time   Wait Time
Non IO (other) Wait Event                                    %        (sec)        (ms) Tot Waits
------------------------------------------------------------ ----- ---------- ---------- ---------
os thread startup                                             40       48.53   1,6176.67         3
enq: RO - fast object reuse                                   25       30.00     7500.00         4
buffer busy waits                                             24       29.08      605.83        48
cr request retry                                               9       11.22        0.14    77,906
latch: enqueue hash chains                                     3        3.76      250.67        15
```


How do you know, with rtsysx.sql?

```
*** SQL Activity Details During Probe

                          Phys Rds  Log Rds  Tot Time   CPU Time                          Rows        Stmt
SQL ID             Sec/EXE      (k)      (k)      (sec)      (sec) Sec/PIO Sec/LIO     Runs      (k)  Sorts Type
---------------- --------- -------- -------- --------- ---------- ------- ------- -------- -------- ------ -----
cvb1dphpubtfw        18.43       24    3,874     571.5      100.0   0.024   0.000       31        0 ###### SELEC
2rnnuhxkj3y2q        10.76        1      149     172.1        1.3   0.185   0.001       16      160      0 UPDAT
aqrakyptx32j0         9.88        1      116     167.9        0.9   0.183   0.001       17      170      0 UPDAT
9gm93sw7hq936        11.19        1      116     167.9        1.0   0.163   0.001       15      150      0 UPDAT
fm7q0unp5a0by         9.87        1      143     157.9        1.2   0.136   0.001       16      160      0 UPDAT
50q9gvxbrym8k        75.30      153      300     150.6        2.4   0.001   0.001        2        0      0 SELEC
c41xqj15aa6t6        74.72      188      369     149.4        2.9   0.001   0.000        2        0      0 SELEC
1kmn18p5vqmkh        70.55      187      366     141.1        2.9   0.001   0.000        2        0      0 SELEC
9bmhnvbndt693        69.88      172      337     139.8        2.6   0.001   0.000        2        0      0 SELEC
gcs0nwp0yudyq        69.37      189      371     138.7        3.0   0.001   0.000        2        0      0 SELEC
98kndhdunkf0r        67.93      190      372     135.9        3.0   0.001   0.000        2        0      0 SELEC
acz1t53gkwa12         7.86        1      113     133.6        0.9   0.156   0.001       17      170      0 UPDAT
ddjn9szuw8n2r        57.14      161      312     114.3        2.5   0.001   0.000        2        0      0 SELEC
5qbu1vnjk7jkn         9.35      384      748      46.7        4.2   0.000   0.000        5        0      0 SELEC
a0n2bkax2r0k6         8.68      319      623      34.7        3.3   0.000   0.000        4        0      0 SELEC
gz5bfrcjq060u         0.79        0        0      15.9        0.5   3.963   0.089       20        0     19 INSER
7msxd2qjtsw05         2.76        0        0       2.8        0.0   0.921   0.691        1        0      0 SELEC
```

©2010 OraPub, Inc. Resolving FBW

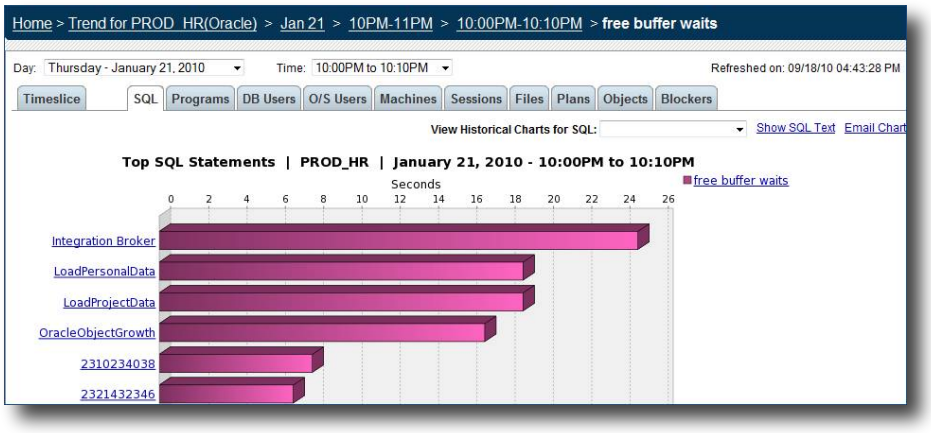
How do you know, using ASH?

```
SQL> @ashrt 10 %

Database: prod18                                          04-MAY-10 11:57am
Report:   ashrt.sql            OSM by OraPub, Inc.           Page         1
            ASH Response Time Analysis Report (last 10 min, SQL_ID=%)

% Service Time % Queue Time  % Qt IO % Qt Other
          (cpu)       (wait)   (wait)     (wait)
-------------- ------------ -------- ----------
          7.64        92.36    99.28       0.72

Activity Detail                        % R Time
-------------------------------------- --------
free buffer waits                         52.37
direct path read                          24.78
db file scattered read                     8.31
db file sequential read                    4.38
log file switch completion                 1.86
```


How do you know, using ASH?

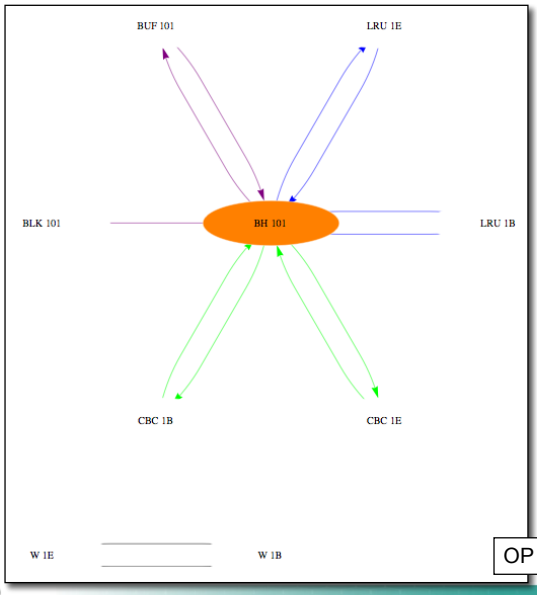
```
SQL> @ashsqlpcte 10 free%buffer

Database: prod18                                          04-MAY-10 11:58am
Report:   ashsqlpcte.sql           OSM by OraPub, Inc.          Page         1
                  System Level ASH: Find SQL by event (last 10 min)

                                                                  % Time
Wait Event                           SQL_ID        ADDRESS  HASH_VALUE  Waited
------------------------------------ ------------- -------- ---------- -------
free buffer waits                    cvb1dphpubtfw 3322B8E0  732292572   27.76
free buffer waits                    2rnnuhxkj3y2q 2FF59E34 1695676502   11.03
free buffer waits                    9gm93sw7hq936 30757B08  252388454   10.82
free buffer waits                    acz1t53gkwa12 2FF9AFD8 3744344098   10.75
free buffer waits                    aqrakyptx32j0 2FF5E278 1943112224   10.51
free buffer waits                    fm7q0unp5a0by 30767B78  710214014   10.44

SQL> @ashsqlpctcpu 10

Database: prod18                                          04-MAY-10 11:59am
Report:   ashsqlpctcpu.sql         OSM by OraPub, Inc.          Page         1
                  System Level ASH: Find SQL by event (last 10 min)

                                  % Time
SQL_ID        ADDRESS  HASH_VALUE     CPU
------------- -------- ---------- -------
cvb1dphpubtfw 3322B8E0  732292572   65.69
7bkgng9mb666d 30079BFC 1723013325    1.81
a0n2bkax2r0k6 307CA920 3123413574    1.67
1kmn18p5vqmkh 30730024 1270566480    1.53
6gvch1xu9ca3g 364D3AA8 1955997807    1.53
```

Using a professional product.



This is a quick drilldown screen shot from Confio's Ignite product.

Buffer headers : the map.



OP Search: explore bc

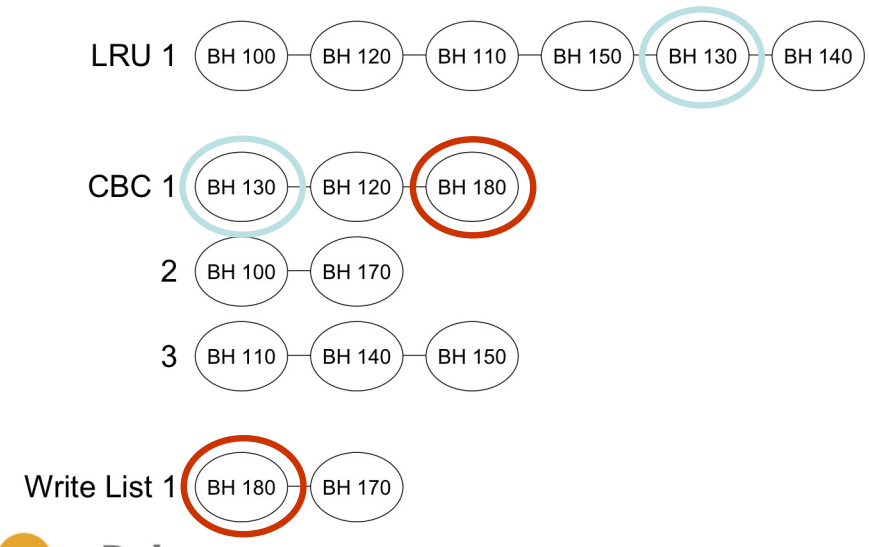
A buffer can be pinned, dirty, or free.

- **Free/mirrored.** When a block's cached image mirrors the actual block on disk. The cached block (I.e., the buffer) can be replaced by another block and it can be referenced by any Oracle process. When the contents of a dirty buffer has been written to disk, it is once again a mirror of the actual block and placed back on the LRU end of its LRU list. An empty buffer is also called a free buffer. There are multiple buffer *states* (**`x$bh.state`**) related to a free/mirrored buffer.
- **Pinned**. A buffer is pinned to keep it from being replaced. You don't want a block you're referencing to be replaced! Pinning can also be used to ensure serial access.
- **Dirty**. When a block's cached image is not the same as its database file image. After a block has been changed (I.e., was pinned) and has not yet been written back to its database file, it is deemed "dirty." (**`v$bh.dirty='Y'`**)
- v$bh and sys.x$bh contain buffer details.

There are three key lists.

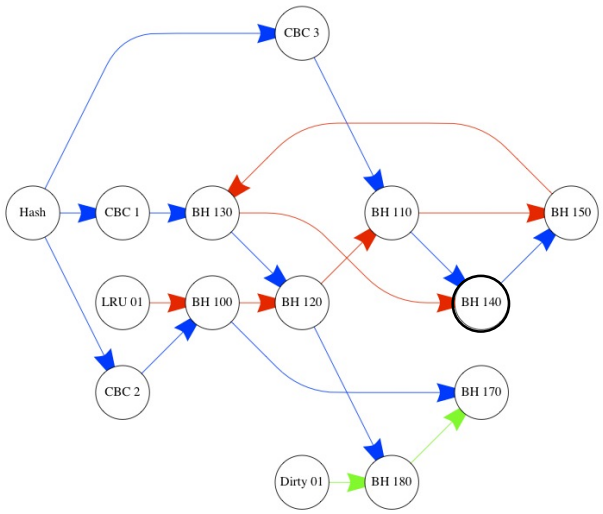
- **Hash buffer chains** are used to quickly determine if a block is in the cache.
- **LRU (LRU) chains** are used to keep popular buffers cached and to find free buffers.
- **Write (LRU-W) lists** are used by its DBWR to batch write dirty buffers. Also called the *dirty list*.

Buffer headers are on many lists.



 Resolving FBW

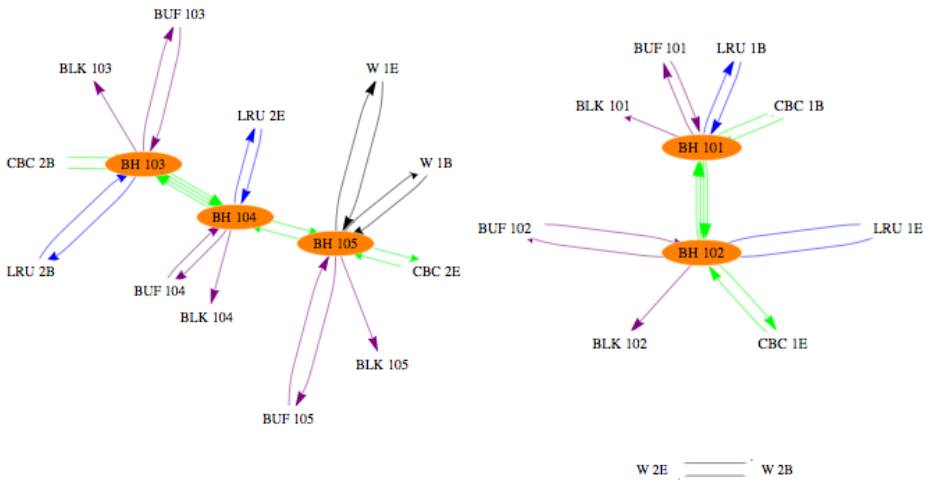
A more connected view.



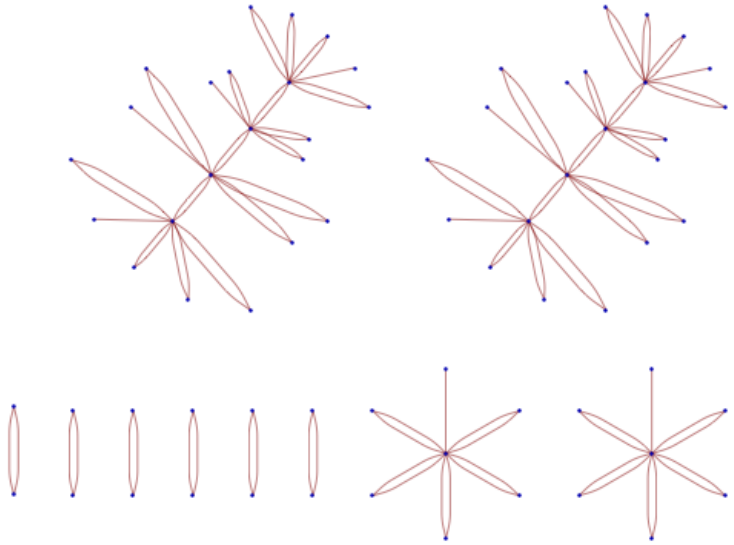
How many LRUs, write lists, CBC chains, and buffer headers are there?

 Resolving FBW

Always on multiple lists.

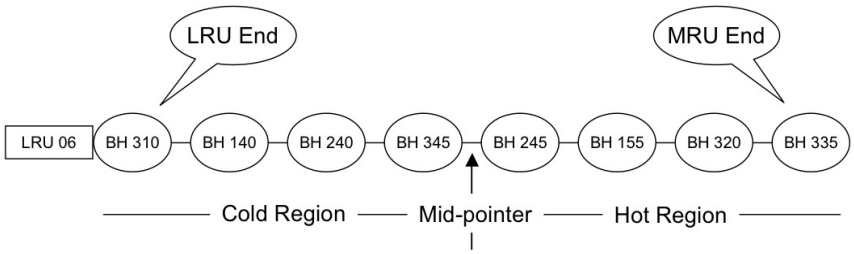


Less clutter…more information.



The key TC components.

- Midpoint insertion
- Touch count incrementation
- Buffer promotion
- Parameters for everything...



Resolving FBW

The write list and the DBWR.

- Lists of dirty buffer (headers) go by many names; dirty list, write list, LRU-W, checkpoint queue, etc.
- **Write list** is composed entirely of dirty buffer headers.
- Each LRU has an associated write list.
- Dirty buffers are moved to their associated write list by either a:
  - Server process, when its looking for free buffers, or its
  - DBWR when looking for something to do.
- The DBWR will write the dirty buffers, in small batches, to their "dbf" file making them free buffers again and then placing the newly free buffers on the LRU end of their LRU chain.

Resolving FBW

When the DBWR is active.

```
SQL> l
  1* select count(*) from v$bh where dirty='Y'
SQL> /

  COUNT(*)
----------
       219
SQL> /

  COUNT(*)
----------
       214
SQL> /

  COUNT(*)
----------
       287
SQL> /

  COUNT(*)
----------
       274
SQL> /

  COUNT(*)
----------
       212
```

The number of dirty buffers is constantly changing.

```
SQL> /

  COUNT(*)
----------
       236
SQL> /

  COUNT(*)
----------
       156
SQL> /

  COUNT(*)
----------
       148
SQL> /

  COUNT(*)
----------
       275
SQL> /

  COUNT(*)
----------
       261
```


A very active DBWR…

```
[oracle@fourcore ~]$ ps -eaf|grep dbw
oracle   11455     1  0 09:01 ?        00:00:02 ora_dbw0_prod18
oracle   12063 12023  0 10:08 pts/3    00:00:00 grep dbw
[oracle@fourcore ~]$ strace -rp 11455
Process 11455 attached - interrupt to quit
     0.000035 times({tms_utime=103, tms_stime=167, tms_cutime=0, tms_cstime=0}) = 488524973
     0.000036 semtimedop(… {2, 770000000}) = -1 EAGAIN (Resource temporarily unavailable)
     2.771256 gettimeofday({1264529314, 633792}, NULL) = 0
     0.000026 getrusage(RUSAGE_SELF, {ru_utime={1, 31843}, ru_stime={1, 672745}, ...}) = 0
     0.000030 gettimeofday({1264529314, 636133}, NULL) = 0
     0.000030 pwrite64(22, "\35\242\0\0\2\0\300\0\262\237\35…=\1\0"..., 16384, 16384) = 16384
     0.009033 gettimeofday({1264529314, 645200}, NULL) = 0
     0.000028 pwrite64(22, "&\242\0\0\0\1\300\0\276\235\35…"..., 8192, 2097152) = 8192
     0.000393 gettimeofday({1264529314, 646163}, NULL) = 0
     0.000029 pwrite64(22, "\2\242\0\0\3413\301\0\30\236\35…"..., 253952, 645668864) = 253952
     0.001805 gettimeofday({1264529314, 648026}, NULL) = 0
     0.000028 pwrite64(22, "\2\242\0\0\0;\301\0=\235\35….., 565248, 660602880) = 565248
     0.005447 gettimeofday({1264529314, 655395}, NULL) = 0
     0.000029 pwrite64(22, "\2\242\0\0\200;\301\0\"\236\35…"..., 131072, 661651456) = 131072
     0.001584 gettimeofday({1264529314, 657043}, NULL) = 0
     0.000029 pwrite64(23, "\6\242\0\0\311\0\0\1>\240\35…"..., 122880, 1646592) = 122880
     0.001072 gettimeofday({1264529314, 659263}, NULL) = 0
     0.000036 gettimeofday({1264529314, 659297}, NULL) = 0
     0.000029 pwrite64(23, "\6\242\0\0\351\0\0\1>\240\35…"..., 122880, 1908736) = 122880
     0.001134 gettimeofday({1264529314, 661674}, NULL) = 0
     0.000028 pwrite64(23, "\6\242\0\0\371\0\0\1>\240\35…"..., 40960, 2039808) = 40960
     0.001136 gettimeofday({1264529314, 662870}, NULL) = 0
<unfinished ...>
Process 11455 detached
```

o11.2

An inactive DBWR.

```
[oracle@fourcore ~]$ ps -eaf|grep dbw
oracle   12300     1  0 10:21 ?        00:00:00 ora_dbw0_prod18
oracle   12350 12023  0 10:21 pts/3    00:00:00 grep dbw
[oracle@fourcore ~]$ strace -rp 12300
Process 12300 attached - interrupt to quit
. . .
     0.000045 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.000675 gettimeofday({1264530117, 135109}, NULL) = 0
. . .
     0.000042 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.000698 gettimeofday({1264530120, 136949}, NULL) = 0
. . .
     0.000039 getrusage(RUSAGE_SELF, {ru_utime={0, 10998}, ru_stime={0, 29995}, ...}) = 0
     0.000080 gettimeofday({1264530120, 137160}, NULL) = 0
. . .
     0.000039 times({tms_utime=1, tms_stime=2, tms_cutime=0, tms_cstime=0}) = 488605789
. . .
     0.000042 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.000708 gettimeofday({1264530123, 138798}, NULL) = 0
. . .
     0.000043 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.000702 gettimeofday({1264530126, 140641}, NULL) = 0
. . .
     0.000045 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.001700 gettimeofday({1264530132, 146335}, NULL) = 0
. . .
     0.000042 semtimedop(11239424, x, 1, {3, 0}) = -1 EAGAIN (Resource temporarily unavailable)
     3.001735 gettimeofday({1264530135, 149232}, NULL) = 0
Process 12300 detached
```

o11.2

What is free buffer wait contention?

- The DBWR can't **pull** dirty blocks out of the buffer cache fast enough to make them free. It's not a **push** to disk issue, it's a **pull** from cache issue.
- Two scenarios are common; when a server process is scanning its LRU looking for free buffers and:
  - It has scanned "too many" buffers (pinned or dirty) before it finds a free buffer. The wait event "free buffer wait" is posted, the DBWR is told to write the related LRU-W list, and the server process waits no more than one second. Upon wake-up the server process starts looking for a free buffer again at the LRU end of its LRU.
  - Stumbles upon a dirty buffer and tries to move a buffer to the write list (LRU-W) but can't (perhaps the DBWR is writing it to disk). The server process posts a free buffer wait event and waits until the DBWR is finished.

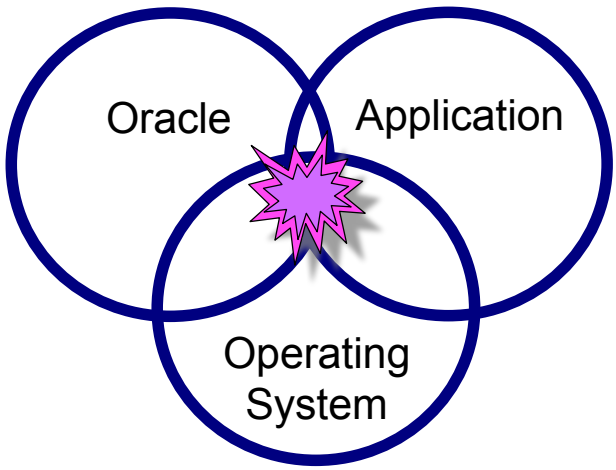
Free buffer wait, wait times.

Wait time is nearly always 16m or less.
rtpctx shows average wait time is 30 to 40 ms.

```
SQL> @swhist free%buffer%


Database: prod18
Report:    swhist.sql           OSM by Or
               Wait Event Activity Histo
```

Wait Event	ms Wait <=	Occurs	Running Occurs %
free buffer waits	1	12449	1.49
free buffer waits	2	6093	2.22
free buffer waits	4	8170	3.20
free buffer waits	8	10318	4.44
free buffer waits	16	789451	99.05
free buffer waits	32	1582	99.24
free buffer waits	64	1224	99.39
free buffer waits	128	1294	99.55
free buffer waits	256	1520	99.73
free buffer waits	512	1052	99.85
free buffer waits	1024	276	99.89
free buffer waits	2048	252	99.92
free buffer waits	4096	230	99.94
free buffer waits	8192	280	99.98
free buffer waits	16384	296	100.01
free buffer waits	32768	209	100.04
free buffer waits	65536	50	100.04
free buffer waits	######	3	100.05

OraPub's 3-Circle Analysis



3-Circle method benefits.

- **Speed.** It's applied parallelism! Why tune sequentially when can tune in parallel? Get everyone involved and working towards a common goal.
- **Structure**. It naturally creates a methodical diagnosis that everyone understands.
- **Communication**. It's natural structure allows for straightforward communication.
- **Confirmation**. The multiple perspective crossover diagnosis results in a pinpoint diagnosis with built-in checks.
- **Cooperation**. Avoids finger-pointing by articulating a common goal where usually, everyone can contribute to the solution.
- **Open**. No specific tool, product, or Oracle release is required. Some product naturally embrace OraPub's 3-Circle method.

 Resolving FBW

Resolving free buffer contention.

- The DBWR can not get dirty buffers to disk fast enough, so increase DBWR *pull power* (e.g., more DBWRs) and perhaps some I/O write throughput improvement…
- The LRU has too many dirty buffers, so:
  - Increase the buffer cache to increase the likelihood of finding a free buffer.
  - Tune SQL to reduce PIO. The server process must find a free buffer to place the block into the cache. By reducing PIO, you reduce the number of times a free buffer must be found.
  - Find DML SQL to understand why (and perhaps reduce) there are so many dirty buffers. This may be an application issue or perhaps a workload balancing issue.
  - Lots of cloned buffers force the need for lots of free buffers. Identify high consistent read SQL and analyze the situation.
  - Encourage the DBWR to write more often, but with smaller batches, by decreasing `_db_large_dirty_queue`.
- Be more patient, search longer, and give the DBWR more time to write its batch by increasing `_db_writer_max_scan_pct`.

 Resolving FBW

Want to dig deeper?

- **Training** from OraPub
  - Oracle Performance Firefighting (I)
  - Adv Oracle Performance Analysis (II)
  - Oracle Forecasting & Predictive Analysis
  - OraPub 1-Day 2010 Perf Seminar
- **Books**
  - Oracle Performance Firefighting (C. Shallahamer)
  - Forecasting Oracle Performance (C. Shallahamer)
- **Products**
  - **OraPub Stress Identifier** now available.
- **Papers/Tools** at www.orapub.com
  - Scientifically Evaluating Alternative Performance Solutions
  - Introduction To Oracle Performance Optimization
  - **Exploring Oracle's [Buffer Cache, Library Cache]**
- **Craig's Blog** at blogspot.com

Philadelphia Oct 18-
Costa Mesa, CA Nov 15-

Arizona/ Oct 4
Florida/ Oct 7

raPub ©2010 OraPub, Inc. Resolving FBW

Visit the IOUG Booth This Week

- Located in the User Group Pavilion - Moscone West, 2nd Floor
- Learn why over 23,000 have joined IOUG and what it can do for you
- Chat with the IOUG Board of Directors
- Hear about new regional IOUG BI user communities
- Find out how to submit an abstract for COLLABORATE 11 – IOUG Forum
- Enter for a chance to win a COLLABORATE 11 registration

IOUG independent oracle users group
For the Complete Technology & Database Professional

COLLABORATE 11
Technology and Applications Forum for the Oracle Community

raPub ©2010 OraPub, Inc. Resolving FBW

Thank You!

OraPub presentations at OOW.

- Sunday, 3pm
  - Resolving the Free Buffer Waits Event
    - S318617, Moscone West, L2, room 2005
- Monday, 11am
  - Optimizing Internal Serialization Control
    - Unconference, Hotel Parc 55, Lombard room
- Wednesday, 1pm
  - Evaluating the Anticipated Effect of Performance Sols.
    - Unconference, Hotel Parc 55, Lombard room
- Thursday, 1:30pm
  - Quantifying Oracle Database System Performance
    - S315724, Moscone South, room 302

OraPub

Resolving the Free Buffer Waits Event

Craig A. Shallahamer
OraPub, Inc.
craig@orapub.com