Maximizing ORACLE CLOUD Buffer Cache Throughput

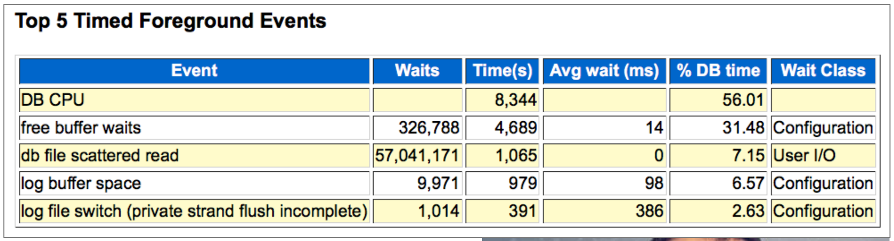
Craig Shallahamer
craig@orapub.com

ORACLE ACE Director

ORAPUB

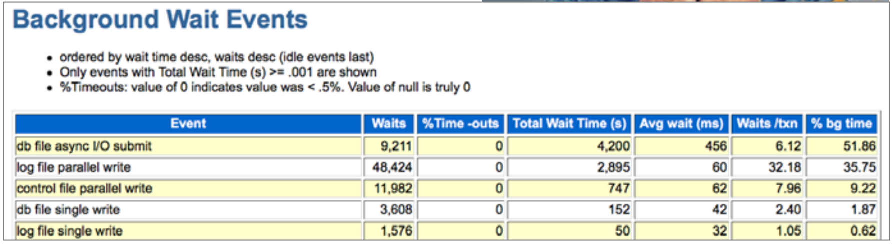
BGOUG
BULGARIAN ORACLE USER GROUP

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



Top 5 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
DB CPU		8,344		56.01	
free buffer waits	326,788	4,689	14	31.48	Configuration
db file scattered read	57,041,171	1,065	0	7.15	User I/O
log buffer space	9,971	979	98	6.57	Configuration
log file switch (private strand flush incomplete)	1,014	391	386	2.63	Configuration



Background Wait Events

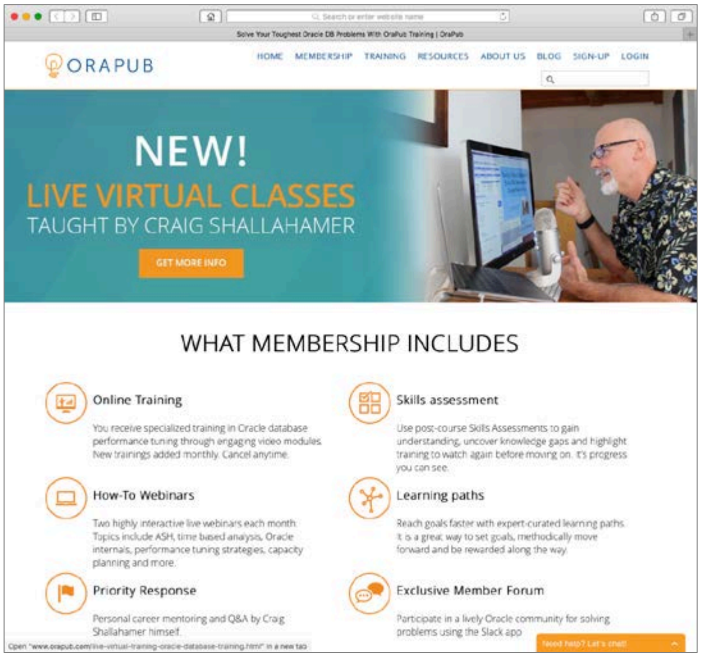
- ordered by wait time desc, waits desc (idle events last)
- Only events with Total Wait Time (s) >= .001 are shown
- %Timeouts: value of 0 indicates value was < .5%. Value of null is truly 0

Event	Waits	%Time -outs	Total Wait Time (s)	Avg wait (ms)	Waits /txn	% bg time
db file async I/O submit	9,211	0	4,200	456	6.12	51.86
log file parallel write	48,424	0	2,895	60	32.18	35.75
control file parallel write	11,982	0	747	62	7.96	9.22
db file single write	3,608	0	152	42	2.40	1.87
log file single write	1,576	0	50	32	1.05	0.62

About Me...

- Long time Oracle DBA
- Specialize in Oracle Database performance and predictive analysis
- Performance researcher
- Blogger: A Wider View About Oracle Performance Tuning
- Author: Oracle Performance Firefighting and Forecasting Oracle Performance.
- Conference speaker
- Teacher and mentor
- Oracle ACE Director
- IOUG DBA Track Manager

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.


Search or enter website name
Solve Your Toughest Oracle DB Problems With OraPub Training | OraPub
ORAPUB
HOME
MEMBERSHIP
TRAINING
RESOURCES
ABOUT US
BLOG
SIGN-UP
LOGIN
NEW!
LIVE VIRTUAL CLASSES
TAUGHT BY CRAIG SHALLAHAMER
GET MORE INFO
Need help? Let's chat!

WHAT MEMBERSHIP INCLUDES

Online Training

You receive specialized training in Oracle database performance tuning through engaging video modules. New trainings added monthly. Cancel anytime.

Skills assessment

Use post-course Skills Assessments to gain understanding, uncover knowledge gaps and highlight training to watch again before moving on. It's progress you can see.

How-To Webinars

Two highly interactive live webinars each month. Topics include ASH, time based analysis, Oracle internals, performance tuning strategies, capacity planning and more.

Learning paths

Reach goals faster with expert-curated learning paths. It is a great way to set goals, methodically move forward and be rewarded along the way.

Priority Response

Personal career mentoring and Q&A by Craig Shallahamer himself.

Exclusive Member Forum

Participate in a lively Oracle community for solving problems using the Slack app

Open "www.orapub.com/live-virtual-training-oracle-database-training.html" in a new tab

OraPub works closely with DBAs who want to take their Oracle tuning skills to the next level.

So, you can solve the toughest Oracle DB problems.

How connect with Craig and OraPub

slack Community with a common goal

craig@orapub.com

@OraPubInc

ORAPUB OraPub.Com: Everything starts here!

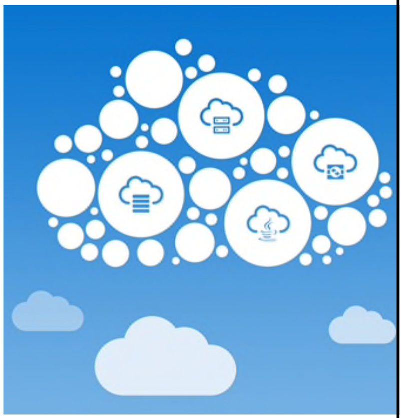
LinkedIn Connect and network

This presentation was given by Craig Shallahamer (craig@orapub.com) at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Key Topics

- Key buffer cache memory structures
- LRU list internals and algorithm
- LRU processing algorithm interesting observations
- How free buffer waits occur
- Free buffer wait solutions
- Going deeper

Key Buffer Cache Memory Structures



This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Free, Dirty, and Pinned…

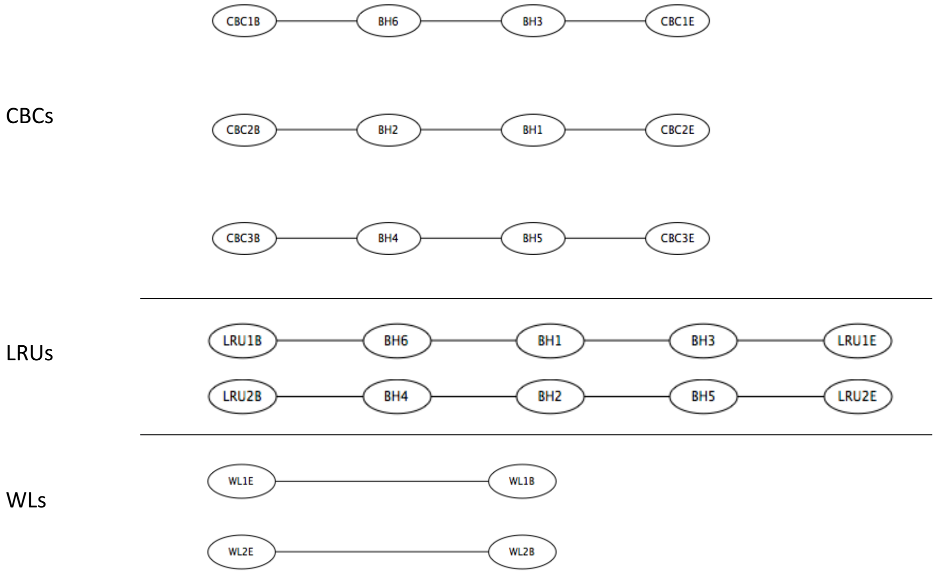
- Free/mirrored. When a block's cached image mirrors the actual block on disk. The cached block (I.e., the buffer) can be replaced by another block and it can be referenced by any Oracle process. When the contents of a dirty buffer has been written to disk, it is once again a mirror of the actual block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

There are three *key* buffer cache lists.

- **Hash/Cache buffer chains** are used to quickly determine if a block is in the buffer cache.
- **LRU (LRU) chains** are used to keep popular buffers in the cache and to find free buffers.
- **Write (LRU-W) lists** are used by the DBWR to batch write dirty buffers. The write list is also called the *dirty list*.
- **Buffer headers** contain pointers to the physical block, the buffer cache location, and its position on its LRU, write, and hash chain list.

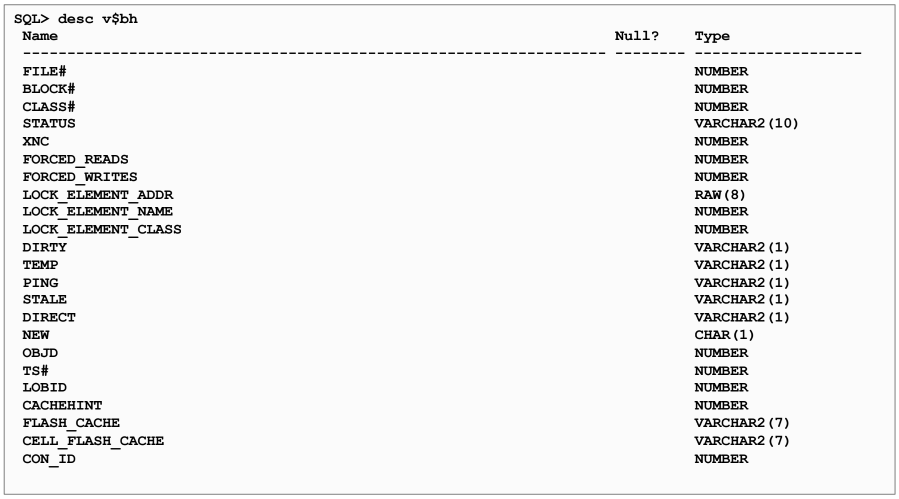
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Buffer headers are on many lists.



The buffer head view, v$bh in 12c.

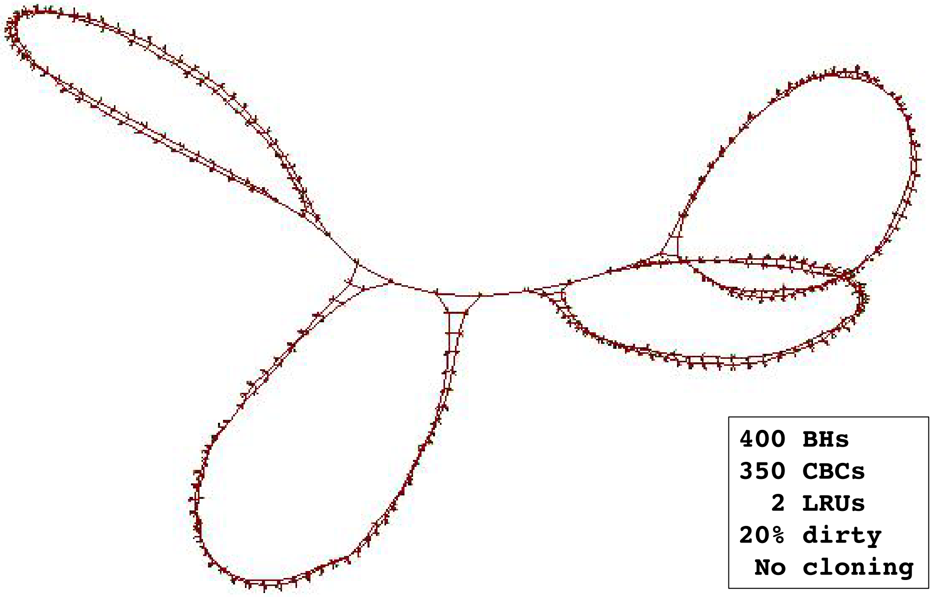
No changes in x$bh. TCH column still only in x$bh.



```
SQL> desc v$bh
 Name                                                              Null?    Type
 ----------------------------------------------------------------- -------- --------------------
 FILE#                                                                      NUMBER
 BLOCK#                                                                     NUMBER
 CLASS#                                                                     NUMBER
 STATUS                                                                     VARCHAR2(10)
 XNC                                                                        NUMBER
 FORCED_READS                                                               NUMBER
 FORCED_WRITES                                                              NUMBER
 LOCK_ELEMENT_ADDR                                                          RAW(8)
 LOCK_ELEMENT_NAME                                                          NUMBER
 LOCK_ELEMENT_CLASS                                                         NUMBER
 DIRTY                                                                      VARCHAR2(1)
 TEMP                                                                       VARCHAR2(1)
 PING                                                                       VARCHAR2(1)
 STALE                                                                      VARCHAR2(1)
 DIRECT                                                                     VARCHAR2(1)
 NEW                                                                        CHAR(1)
 OBJD                                                                       NUMBER
 TS#                                                                        NUMBER
 LOBID                                                                      NUMBER
 CACHEHINT                                                                  NUMBER
 FLASH_CACHE                                                                VARCHAR2(7)
 CELL_FLASH_CACHE                                                           VARCHAR2(7)
 CON_ID                                                                     NUMBER
```

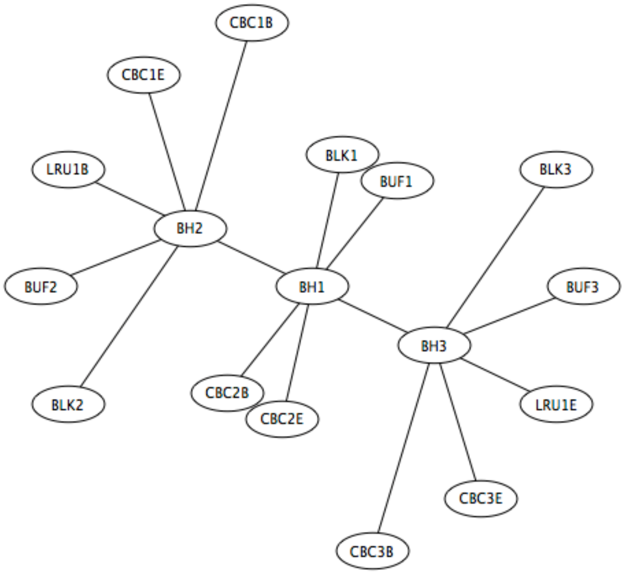
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

The “real” stars of the buffer cache!



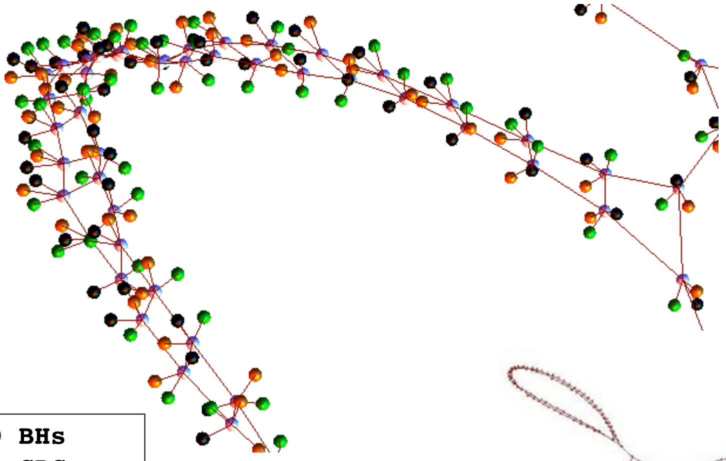
```
400 BHs
350 CBCs
  2 LRUs
20% dirty
 No cloning
```

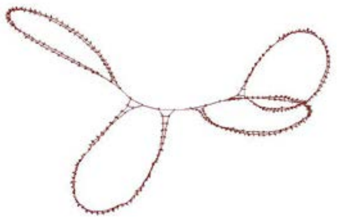
It's the lists and the buffer headers.



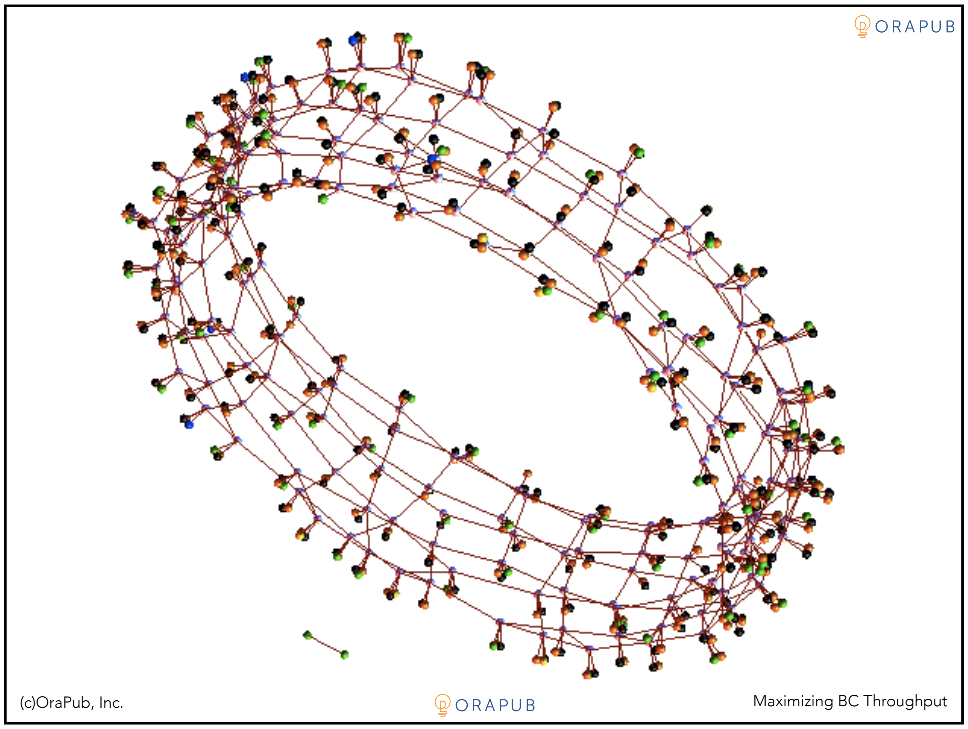
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Visualizing 400 BH fully connected.





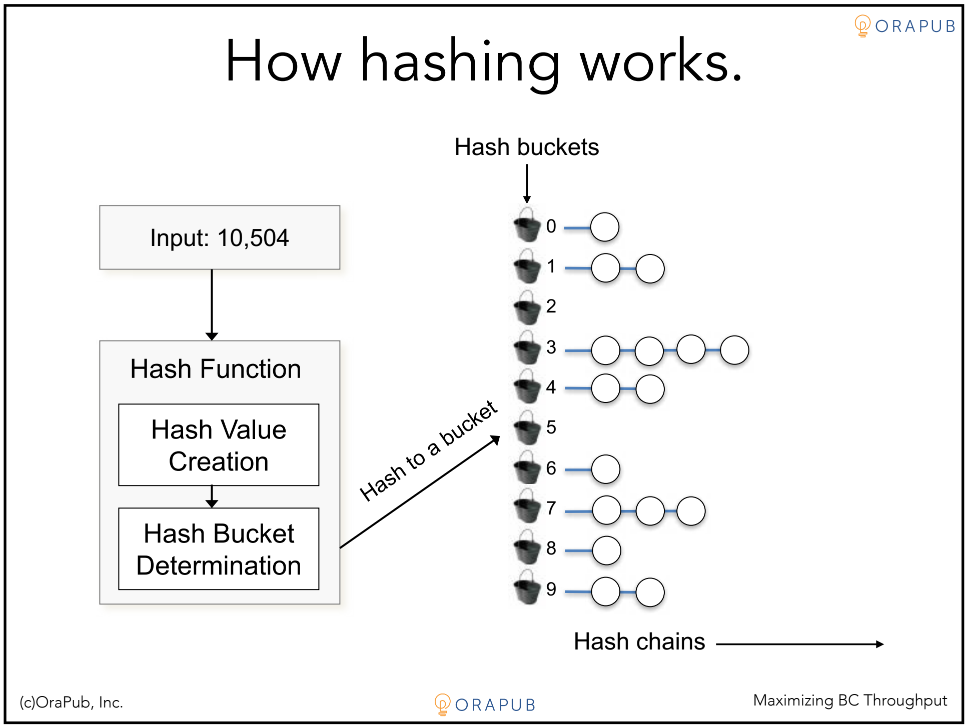
```
400 BHs
350 CBCs
  2 LRUs
20% dirty
 No cloning
```



ORAPUB

(c)OraPub, Inc. ORAPUB Maximizing BC Throughput

This presentation was given by Craig Shallahamer (craig@orapub.com) at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



ORAPUB

How hashing works.

Input: 10,504

Hash Function

Hash Value Creation

Hash Bucket Determination

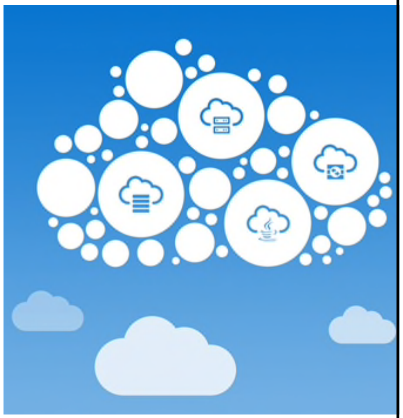
Hash to a bucket

Hash buckets

Hash chains

(c)OraPub, Inc. ORAPUB Maximizing BC Throughput

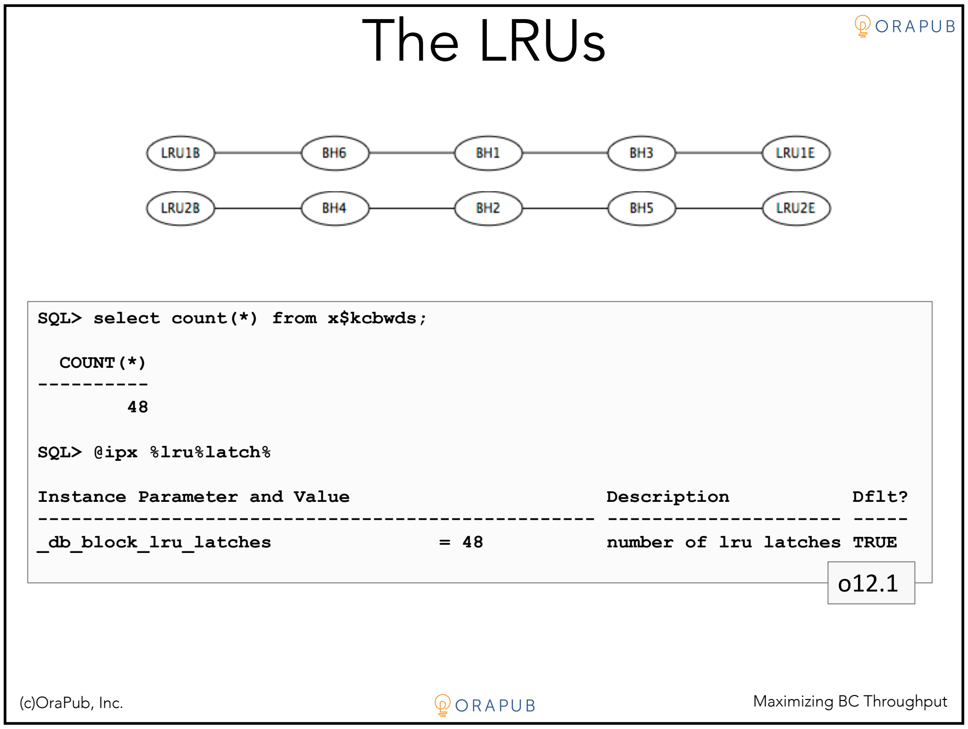
Least Recently Used List Internals and Algorithm



This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Least recently used chains; LRUs.

- LRUs are used to cache hot buffers and to quickly supply free buffers for replacement to server processes.
- Cached buffers are divided into working sets.
- Each working set has an associated LRU chain or list.
- LRUs are made up of cached buffer headers ordered by *popularity*, descending. (not really…)
- Each LRU can contain free and dirty buffer headers. And they can be pinned or not pinned.
- Each DBWR is associated with one or more LRUs.
- There appears to be one LRU latch (`_db_block_lru_latches`) for each LRU chain (`x$kcbwds`). A typical value is less than 100 LRUs.



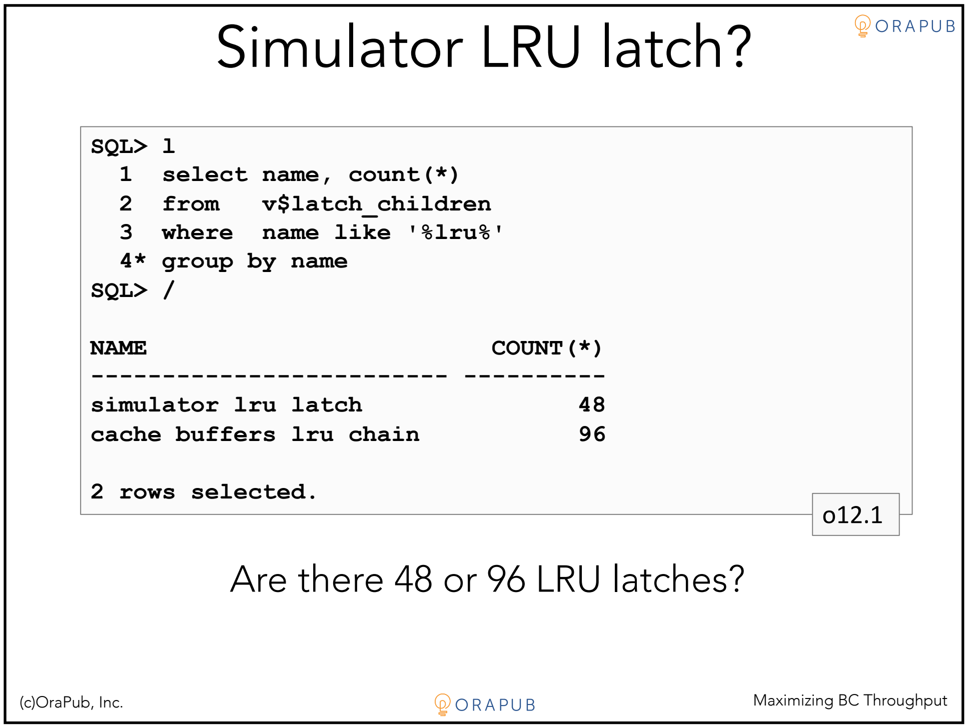
The LRUs

LRU1B
BH6
BH1
BH3
LRU1E
LRU2B
BH4
BH2
BH5
LRU2E

```
SQL> select count(*) from x$kcbwds;

  COUNT(*)
----------
        48

SQL> @ipx %lru%latch%

Instance Parameter and Value                                Description           Dflt?
--------------------------------------------------------    --------------------- -----
_db_block_lru_latches                    = 48               number of lru latches TRUE
```

o12.1

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

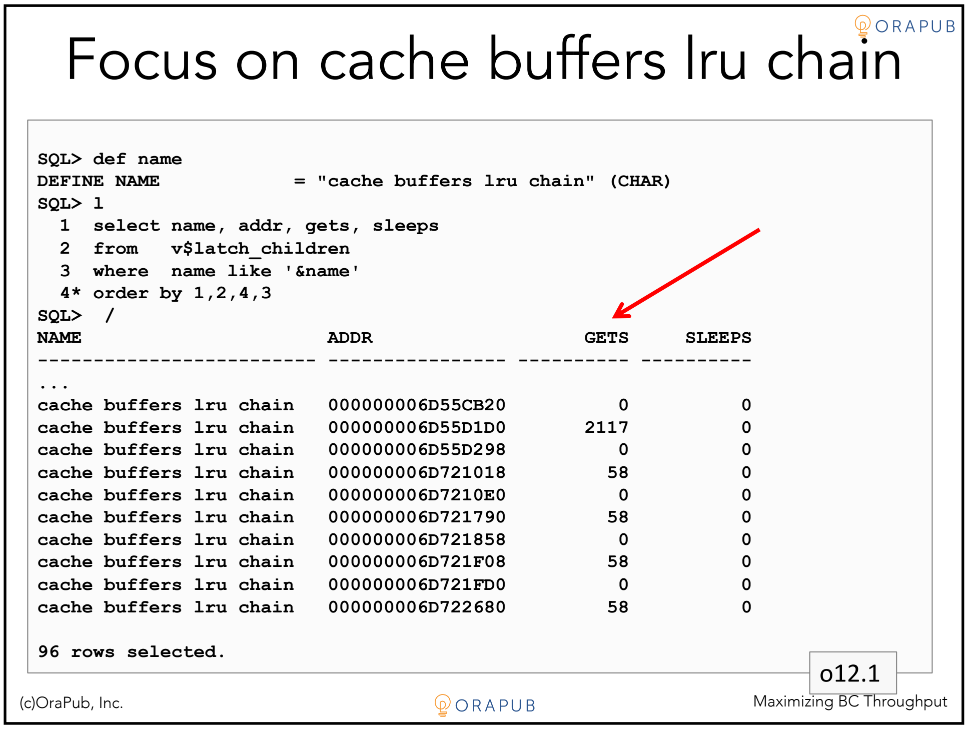


Simulator LRU latch?

```
SQL> l
  1  select name, count(*)
  2  from   v$latch_children
  3  where  name like '%lru%'
  4* group by name
SQL> /

NAME                           COUNT(*)
------------------------- ----------
simulator lru latch               48
cache buffers lru chain           96

2 rows selected.
```

o12.1

Are there 48 or 96 LRU latches?

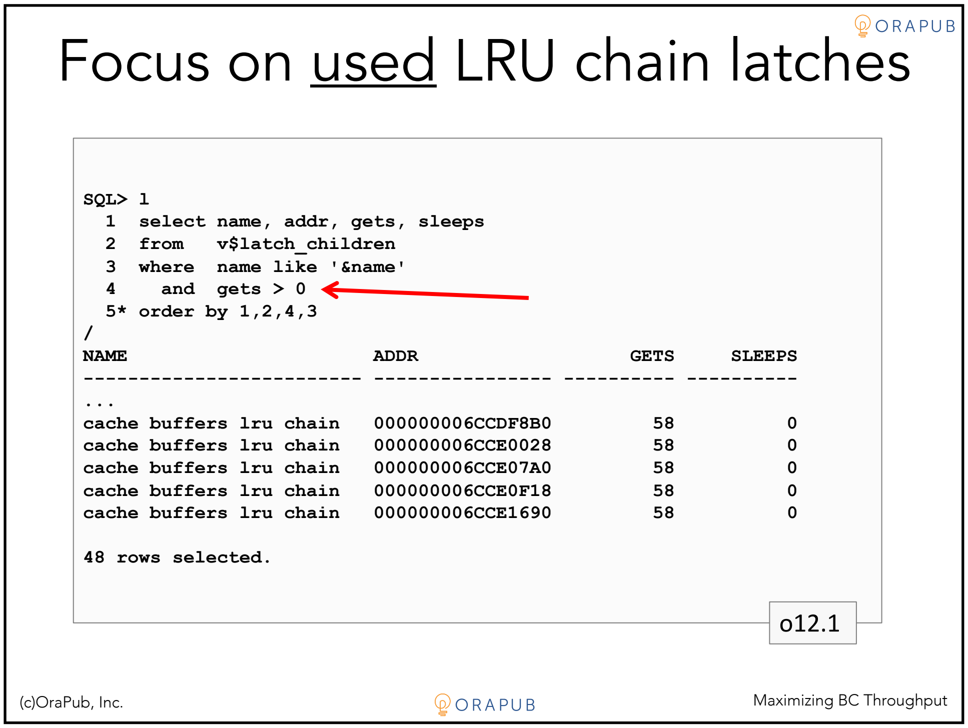


Focus on cache buffers lru chain

```
SQL> def name
DEFINE NAME             = "cache buffers lru chain" (CHAR)
SQL> l
  1  select name, addr, gets, sleeps
  2  from   v$latch_children
  3  where  name like '&name'
  4* order by 1,2,4,3
SQL>  /
NAME                      ADDR                   GETS     SLEEPS
------------------------- ---------------- ---------- ----------
...
cache buffers lru chain   000000006D55CB20          0          0
cache buffers lru chain   000000006D55D1D0       2117          0
cache buffers lru chain   000000006D55D298          0          0
cache buffers lru chain   000000006D721018         58          0
cache buffers lru chain   000000006D7210E0          0          0
cache buffers lru chain   000000006D721790         58          0
cache buffers lru chain   000000006D721858          0          0
cache buffers lru chain   000000006D721F08         58          0
cache buffers lru chain   000000006D721FD0          0          0
cache buffers lru chain   000000006D722680         58          0

96 rows selected.
```

o12.1

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



Focus on used LRU chain latches

```
SQL> l
  1  select name, addr, gets, sleeps
  2  from   v$latch_children
  3  where  name like '&name'
  4    and  gets > 0
  5* order by 1,2,4,3
/
NAME                      ADDR                   GETS     SLEEPS
------------------------- ---------------- ---------- ----------
...
cache buffers lru chain   000000006CCDF8B0         58          0
cache buffers lru chain   000000006CCE0028         58          0
cache buffers lru chain   000000006CCE07A0         58          0
cache buffers lru chain   000000006CCE0F18         58          0
cache buffers lru chain   000000006CCE1690         58          0

48 rows selected.
```

o12.1

Let's summarize what we see

- We can see the number LRUs defined:
  - select count(*) from **x$kcbwds**
- The number latches in use is set by the instance parameter _db_block_lru_latches
- Do not determine the number of LRU latches simply from v$latch_children entries.

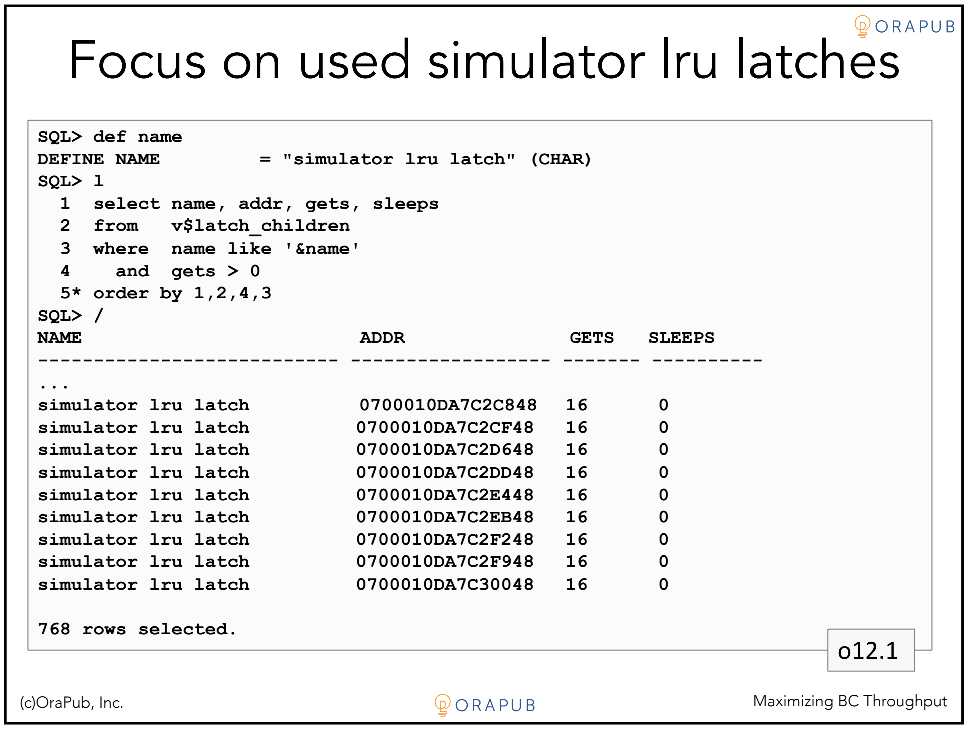
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Check our assumptions on a big production system – 35GB BC

```
SQL> l
  1  select name, count(*)
  2  from   v$latch_children
  3  where  name like '%lru%'
  4* group by name
SQL> /

NAME                            COUNT(*)
------------------------- ----------
cache buffers lru chain         1536
simulator lru latch              768

2 rows selected.
```

o12.1

Are there 1536 or 768 LRU latches?



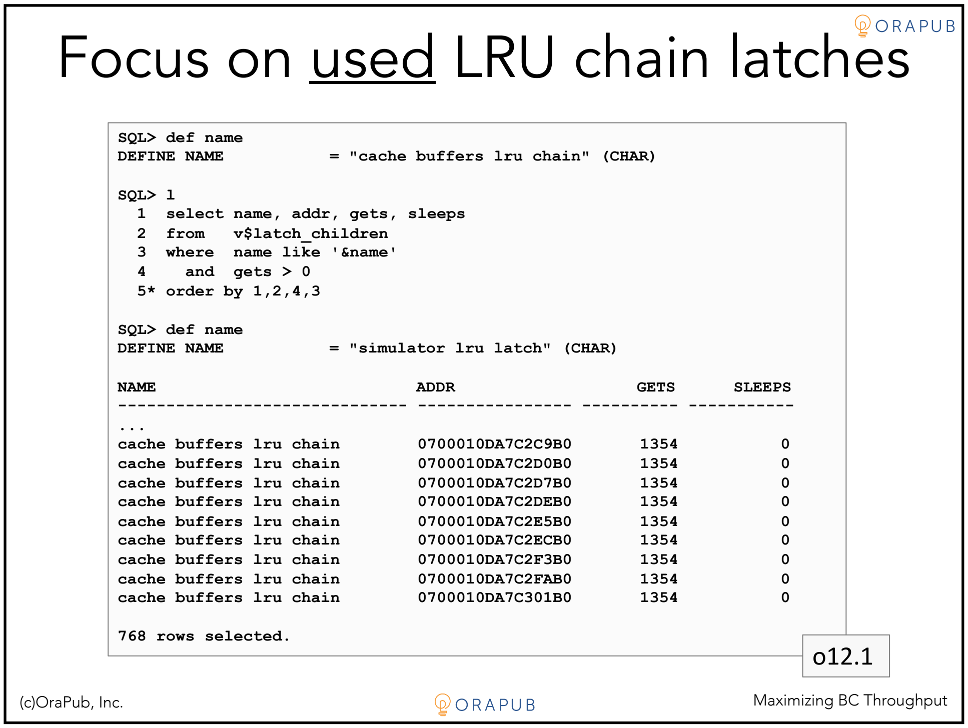
ORAPUB

Focus on used simulator lru latches

```
SQL> def name
DEFINE NAME         = "simulator lru latch" (CHAR)
SQL> l
  1  select name, addr, gets, sleeps
  2  from   v$latch_children
  3  where  name like '&name'
  4    and  gets > 0
  5* order by 1,2,4,3
SQL> /
NAME                         ADDR                GETS   SLEEPS
--------------------------- ------------------ ------- ----------
...
simulator lru latch          0700010DA7C2C848   16       0
simulator lru latch          0700010DA7C2CF48   16       0
simulator lru latch          0700010DA7C2D648   16       0
simulator lru latch          0700010DA7C2DD48   16       0
simulator lru latch          0700010DA7C2E448   16       0
simulator lru latch          0700010DA7C2EB48   16       0
simulator lru latch          0700010DA7C2F248   16       0
simulator lru latch          0700010DA7C2F948   16       0
simulator lru latch          0700010DA7C30048   16       0

768 rows selected.
```

o12.1

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



ORAPUB

Focus on <u>used</u> LRU chain latches

```
SQL> def name
DEFINE NAME           = "cache buffers lru chain" (CHAR)

SQL> l
  1  select name, addr, gets, sleeps
  2  from   v$latch_children
  3  where  name like '&name'
  4    and  gets > 0
  5* order by 1,2,4,3

SQL> def name
DEFINE NAME           = "simulator lru latch" (CHAR)

NAME                           ADDR                   GETS      SLEEPS
------------------------------ ---------------- ---------- -----------
...
cache buffers lru chain        0700010DA7C2C9B0       1354           0
cache buffers lru chain        0700010DA7C2D0B0       1354           0
cache buffers lru chain        0700010DA7C2D7B0       1354           0
cache buffers lru chain        0700010DA7C2DEB0       1354           0
cache buffers lru chain        0700010DA7C2E5B0       1354           0
cache buffers lru chain        0700010DA7C2ECB0       1354           0
cache buffers lru chain        0700010DA7C2F3B0       1354           0
cache buffers lru chain        0700010DA7C2FAB0       1354           0
cache buffers lru chain        0700010DA7C301B0       1354           0

768 rows selected.
```

o12.1

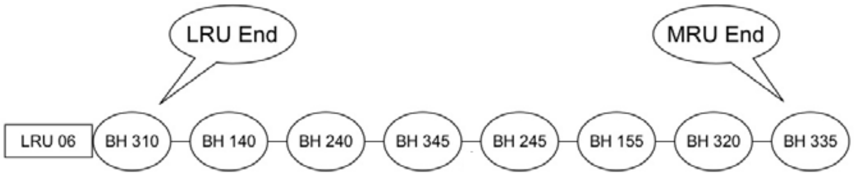
What's the situation on your system?

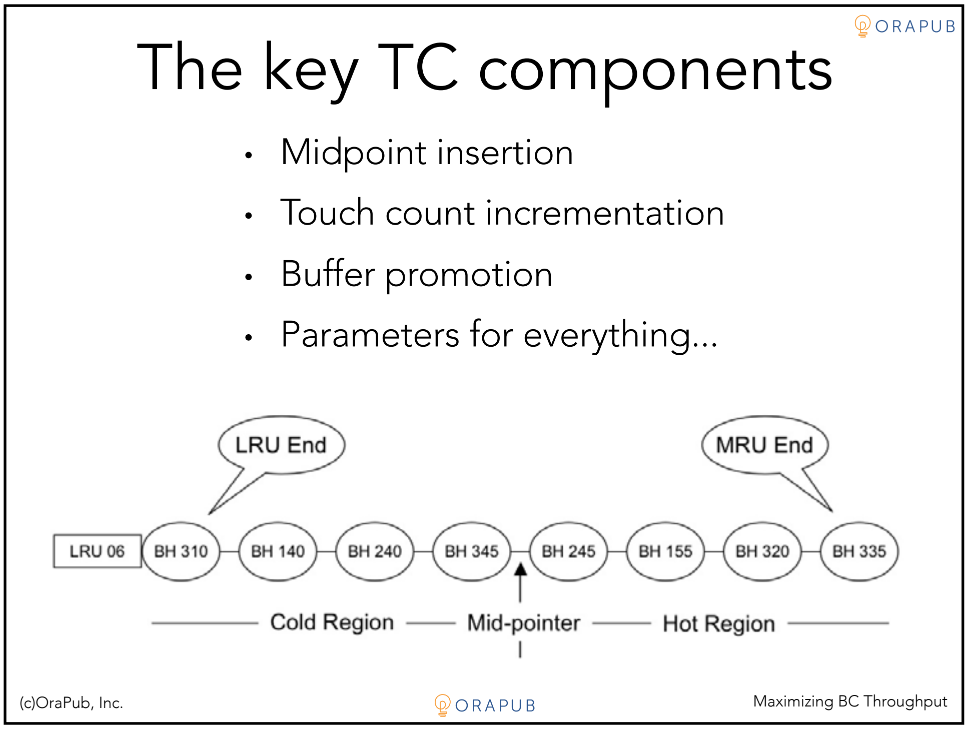
```
col name format a25

select name, count(*)
from   v$latch_children
where  name like '%lru%'
group by name;

def name='cache buffers lru chain'
def name='simulator lru latch'

select name, addr, gets, sleeps
from   v$latch_children
where  name like '&name'
order by 1,2,4,3;

select name, addr, gets, sleeps
from   v$latch_children
where  name like '&name'
  and  gets > 0
order by 1,2,4,3;
```

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Oracle' s caching struggle…

- Standard least recently used algorithm
- Modified least recently used algorithm
- Touch-Count algorithm

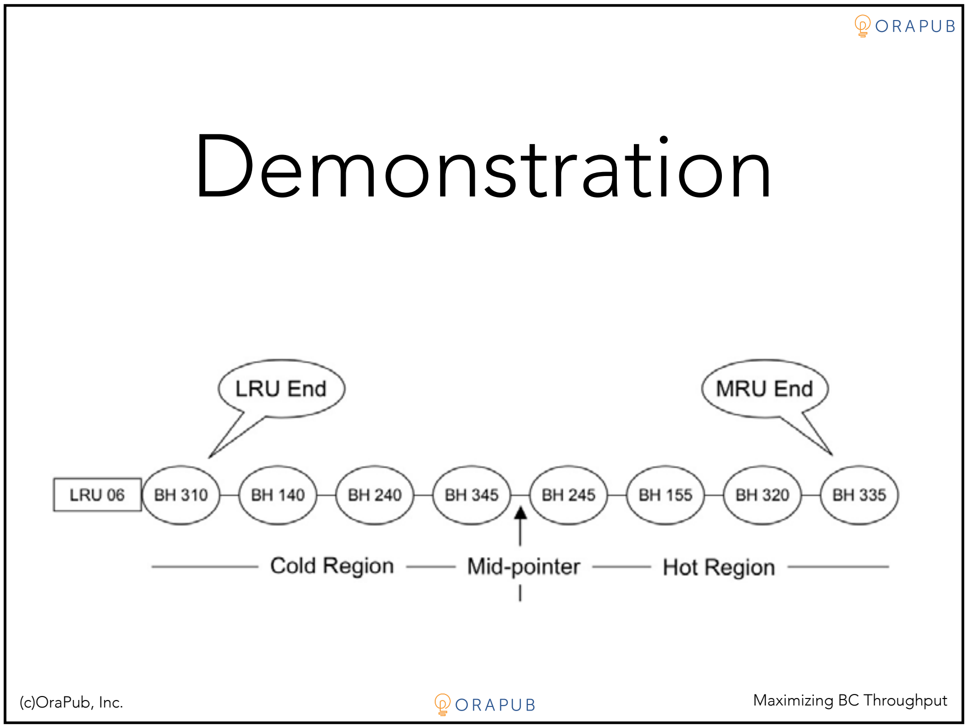




The key TC components

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

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



Demonstration

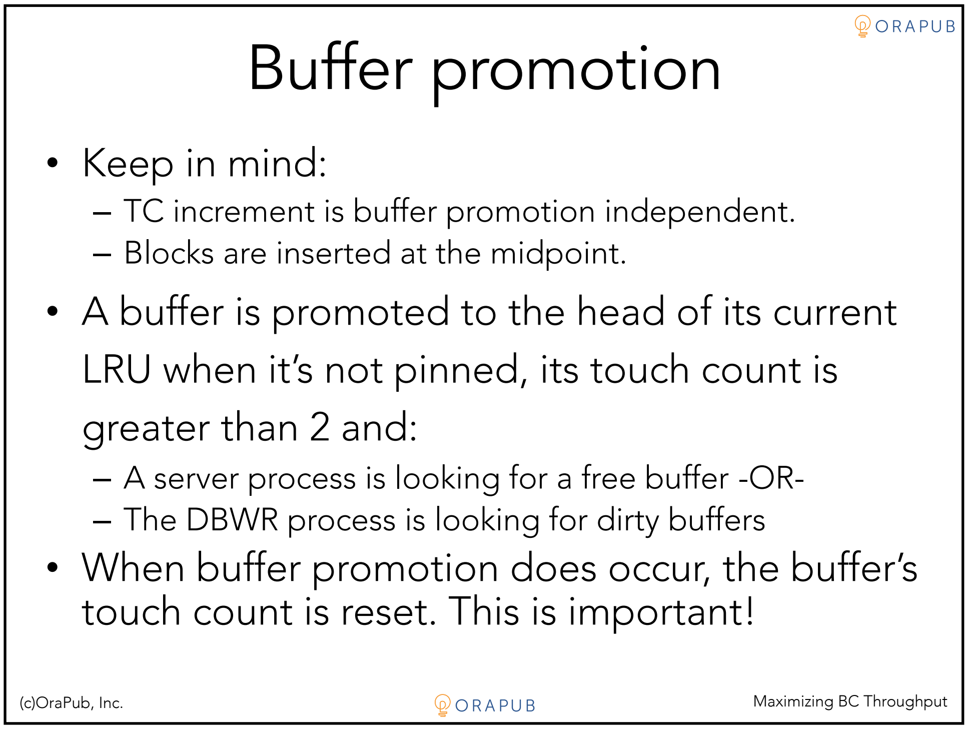
TC midpoint insertion details

- The buffer cache is divided into a *hot region* and a *cold region*.
- A *midpoint pointer* is maintained which *moves* to ensure proper regional block quantities.
- When a block is brought into the cache, it is placed in the “middle” of the LRU chain.
- A buffer naturally moves from the hot region into the cold region.

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Incrementing the touch count

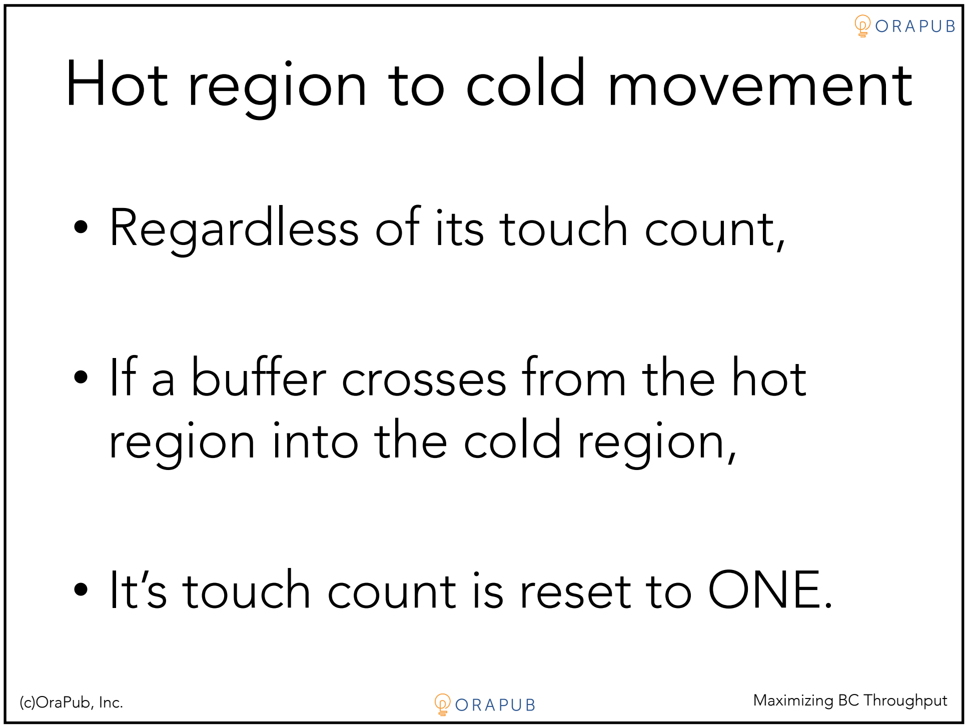
- In concept, whenever a buffer is touched its TC is incremented.
- In reality, it does not work this way.
- No TC latch exist, thereby eliminating latch contention.
- To reduce rapid TC increases, the touch count will only be incremented once every 3 seconds.
- Buffer promotion and touch count increments are independent events.



Buffer promotion

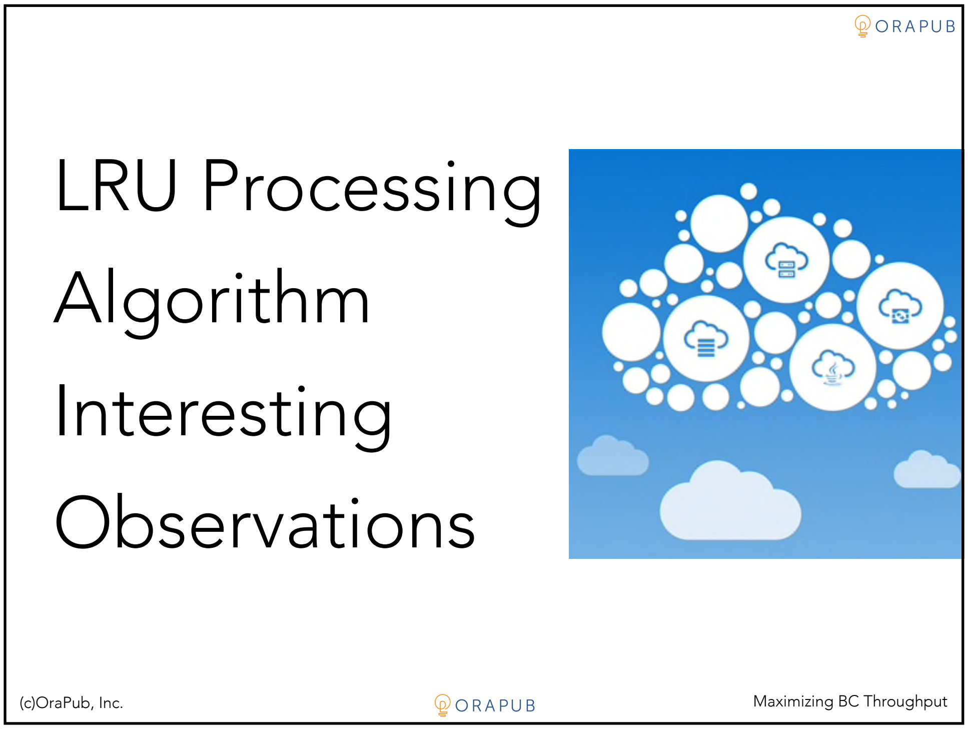
- Keep in mind:
  - TC increment is buffer promotion independent.
  - Blocks are inserted at the midpoint.
- A buffer is promoted to the head of its current LRU when it's not pinned, its touch count is greater than 2 and:
  - A server process is looking for a free buffer -OR-
  - The DBWR process is looking for dirty buffers
- When buffer promotion does occur, the buffer's touch count is reset. This is important!

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



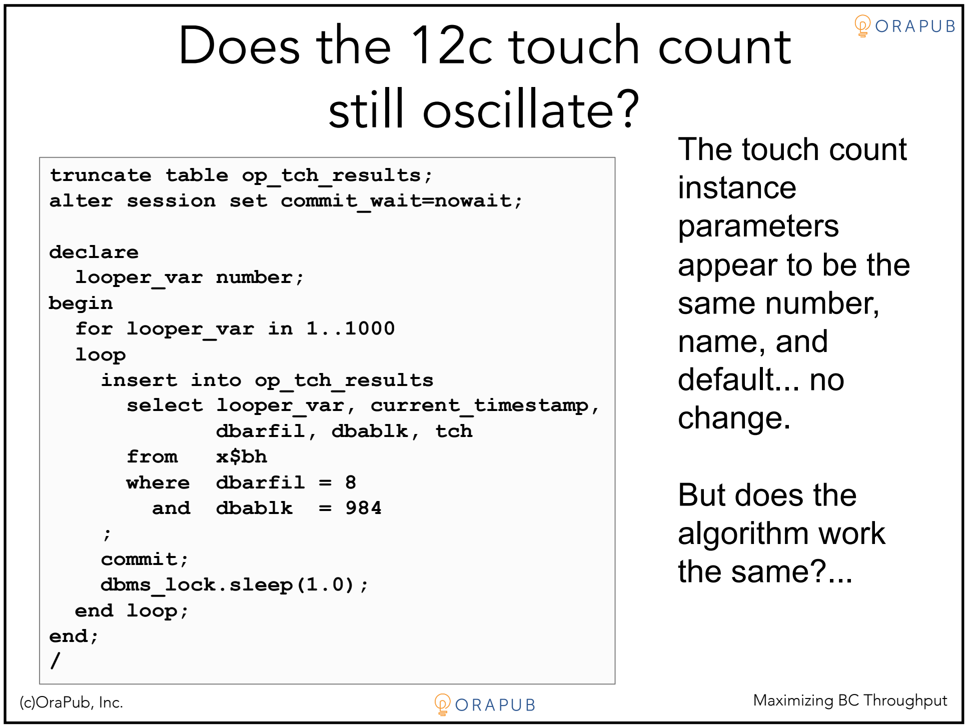
Hot region to cold movement

- Regardless of its touch count,
- If a buffer crosses from the hot region into the cold region,
- It's touch count is reset to ONE.



LRU Processing Algorithm Interesting Observations

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

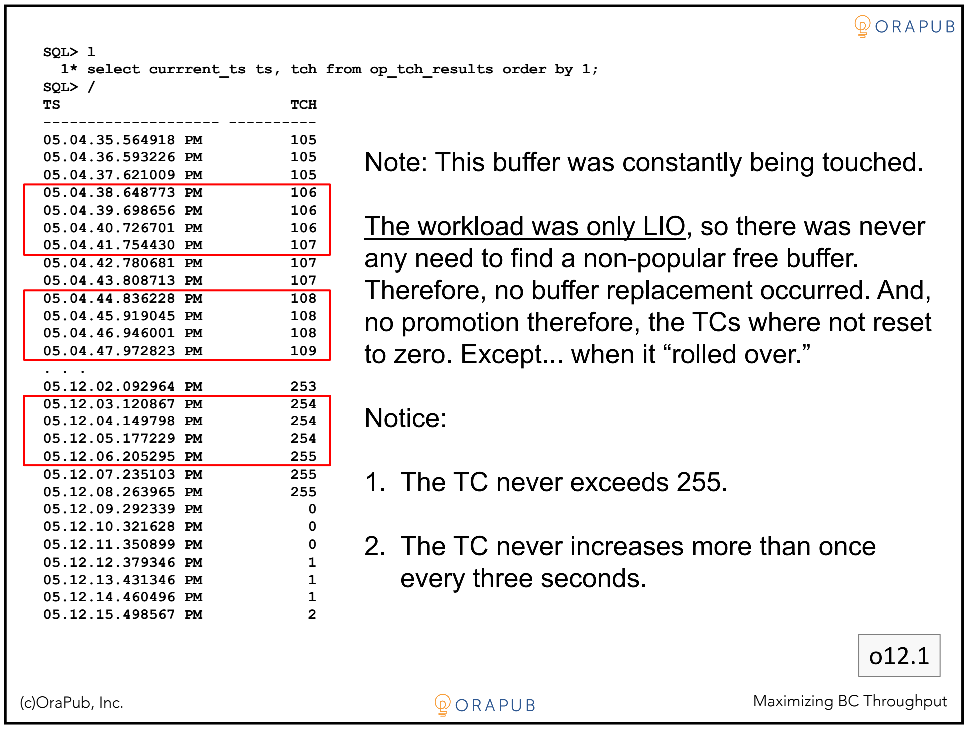


Does the 12c touch count still oscillate?

```
truncate table op_tch_results;
alter session set commit_wait=nowait;

declare
  looper_var number;
begin
  for looper_var in 1..1000
  loop
    insert into op_tch_results
      select looper_var, current_timestamp,
             dbarfil, dbablk, tch
      from   x$bh
      where  dbarfil = 8
        and  dbablk  = 984
    ;
    commit;
    dbms_lock.sleep(1.0);
  end loop;
end;
/
```

The touch count instance parameters appear to be the same number, name, and default... no change.

But does the algorithm work the same?...



```
SQL> l
  1* select currrent_ts ts, tch from op_tch_results order by 1;
SQL> /
TS                           TCH
-------------------- ----------
05.04.35.564918 PM           105
05.04.36.593226 PM           105
05.04.37.621009 PM           105
05.04.38.648773 PM           106
05.04.39.698656 PM           106
05.04.40.726701 PM           106
05.04.41.754430 PM           107
05.04.42.780681 PM           107
05.04.43.808713 PM           107
05.04.44.836228 PM           108
05.04.45.919045 PM           108
05.04.46.946001 PM           108
05.04.47.972823 PM           109
. . .
05.12.02.092964 PM           253
05.12.03.120867 PM           254
05.12.04.149798 PM           254
05.12.05.177229 PM           254
05.12.06.205295 PM           255
05.12.07.235103 PM           255
05.12.08.263965 PM           255
05.12.09.292339 PM             0
05.12.10.321628 PM             0
05.12.11.350899 PM             0
05.12.12.379346 PM             1
05.12.13.431346 PM             1
05.12.14.460496 PM             1
05.12.15.498567 PM             2
```

Note: This buffer was constantly being touched.

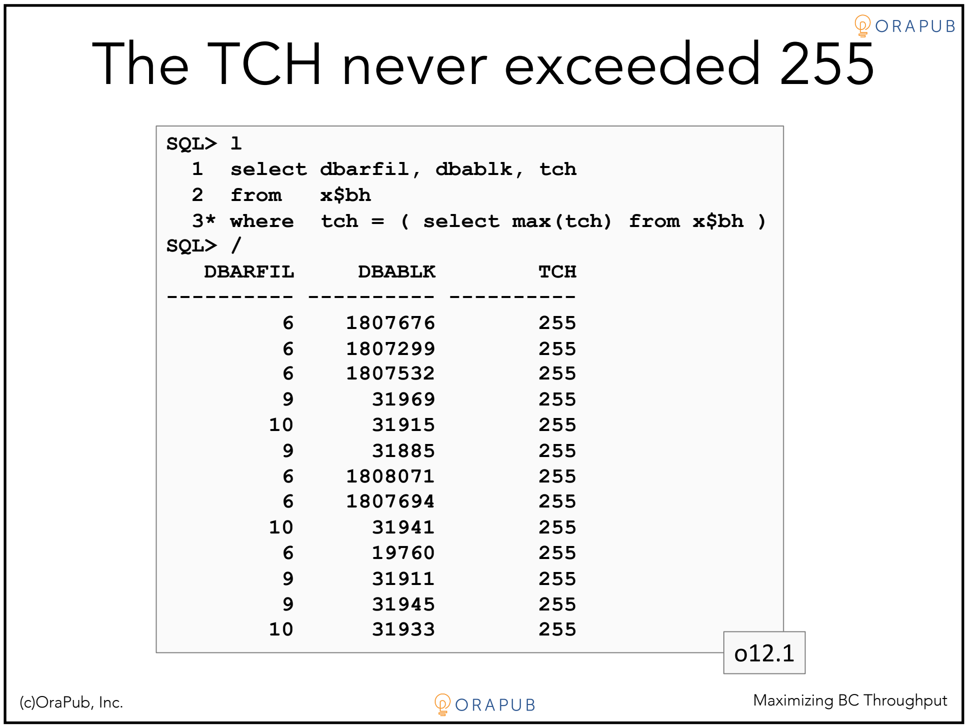
The workload was only LIO, so there was never any need to find a non-popular free buffer. Therefore, no buffer replacement occurred. And, no promotion therefore, the TCs where not reset to zero. Except... when it "rolled over."

Notice:

1. The TC never exceeds 255.
2. The TC never increases more than once every three seconds.

o12.1

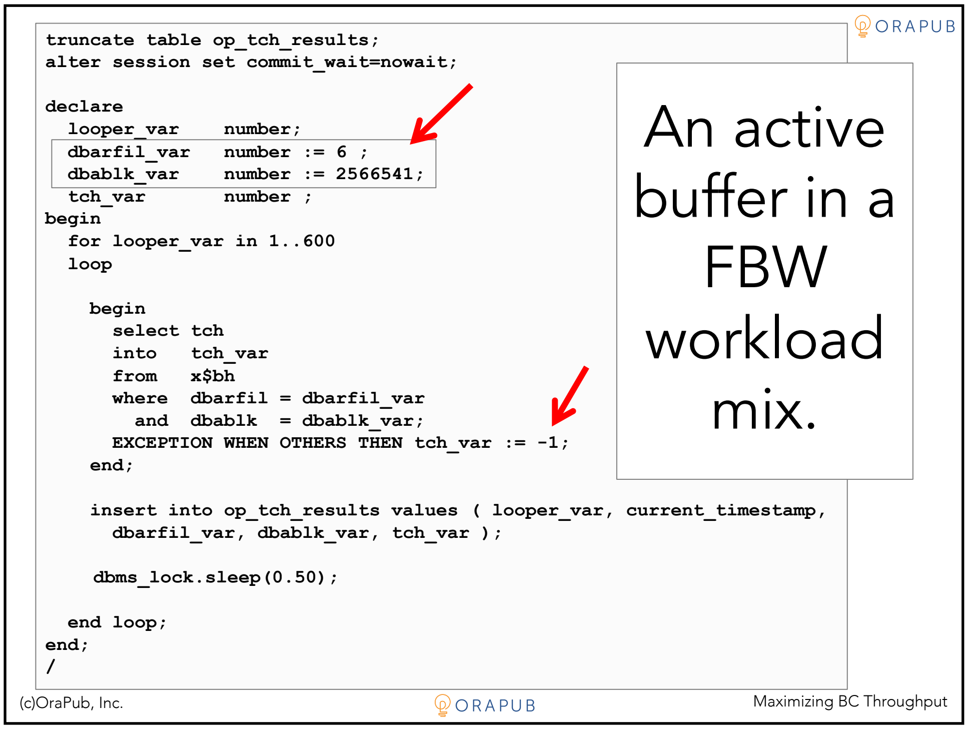
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



The TCH never exceeded 255

```
SQL> l
  1  select dbarfil, dbablk, tch
  2  from   x$bh
  3* where  tch = ( select max(tch) from x$bh )
SQL> /
   DBARFIL     DBABLK        TCH
---------- ---------- ----------
         6    1807676        255
         6    1807299        255
         6    1807532        255
         9      31969        255
        10      31915        255
         9      31885        255
         6    1808071        255
         6    1807694        255
        10      31941        255
         6      19760        255
         9      31911        255
         9      31945        255
        10      31933        255
```

o12.1



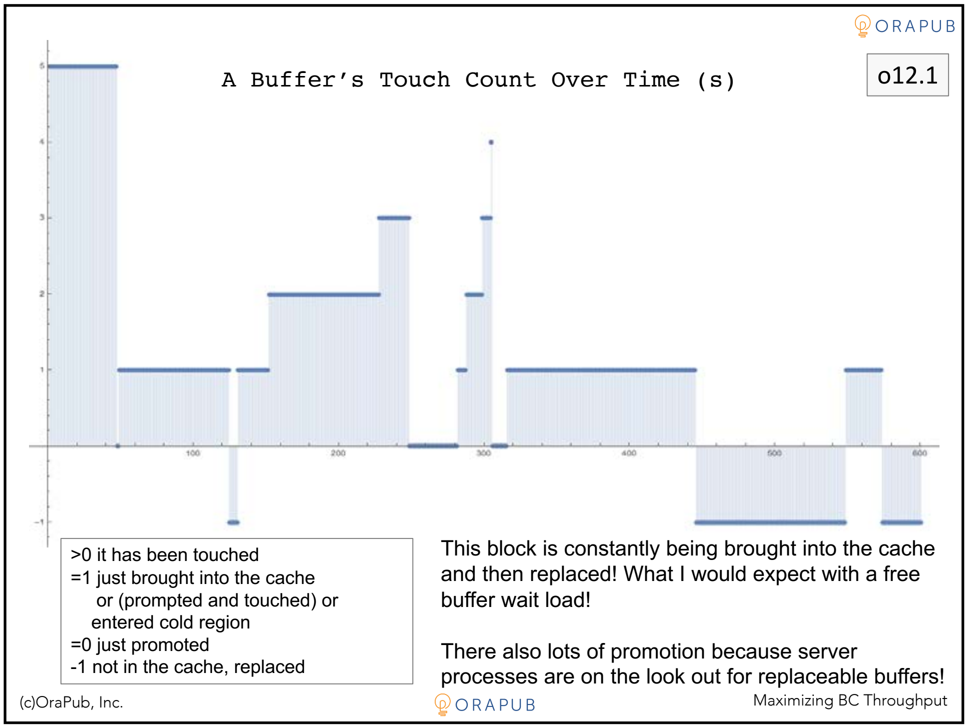
ORAPUB

```
truncate table op_tch_results;
alter session set commit_wait=nowait;

declare
  looper_var    number;
  dbarfil_var   number := 6 ;
  dbablk_var    number := 2566541;
  tch_var       number ;
begin
  for looper_var in 1..600
  loop

    begin
      select tch
      into   tch_var
      from   x$bh
      where  dbarfil = dbarfil_var
        and  dbablk  = dbablk_var;
      EXCEPTION WHEN OTHERS THEN tch_var := -1;
    end;

    insert into op_tch_results values ( looper_var, current_timestamp,
      dbarfil_var, dbablk_var, tch_var );

    dbms_lock.sleep(0.50);

  end loop;
end;
/
```

An active buffer in a FBW workload mix.

(c)OraPub, Inc. ORAPUB Maximizing BC Throughput

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



ORAPUB

A Buffer’s Touch Count Over Time (s)

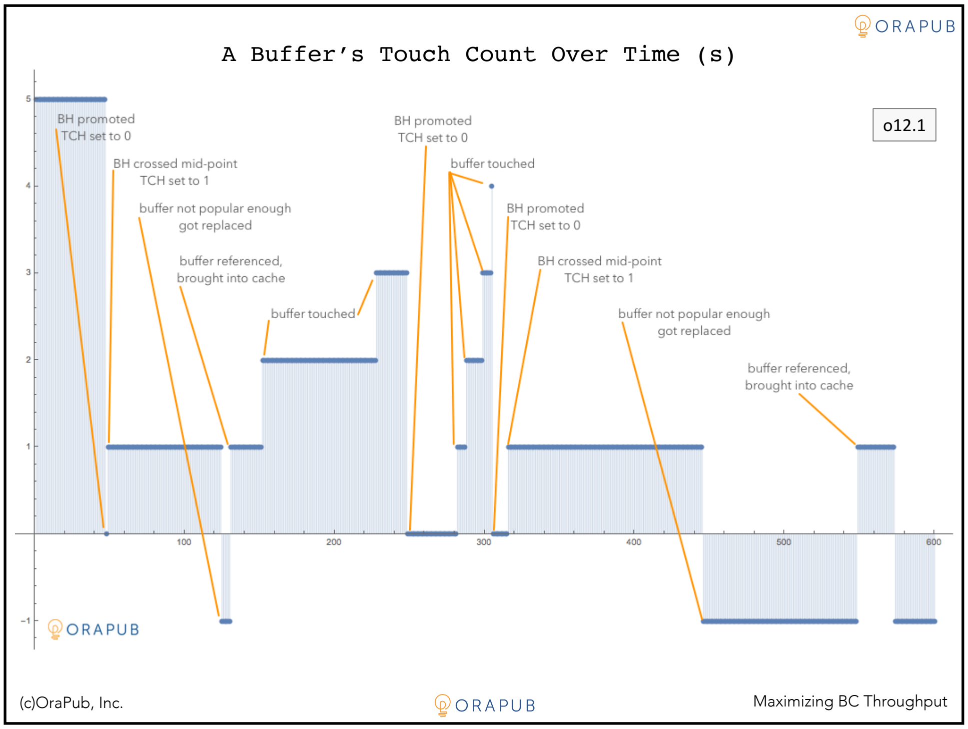
o12.1

>0 it has been touched
=1 just brought into the cache
or (prompted and touched) or
entered cold region
=0 just promoted
-1 not in the cache, replaced

This block is constantly being brought into the cache and then replaced! What I would expect with a free buffer wait load!

There also lots of promotion because server processes are on the look out for replaceable buffers!

(c)OraPub, Inc. ORAPUB Maximizing BC Throughput



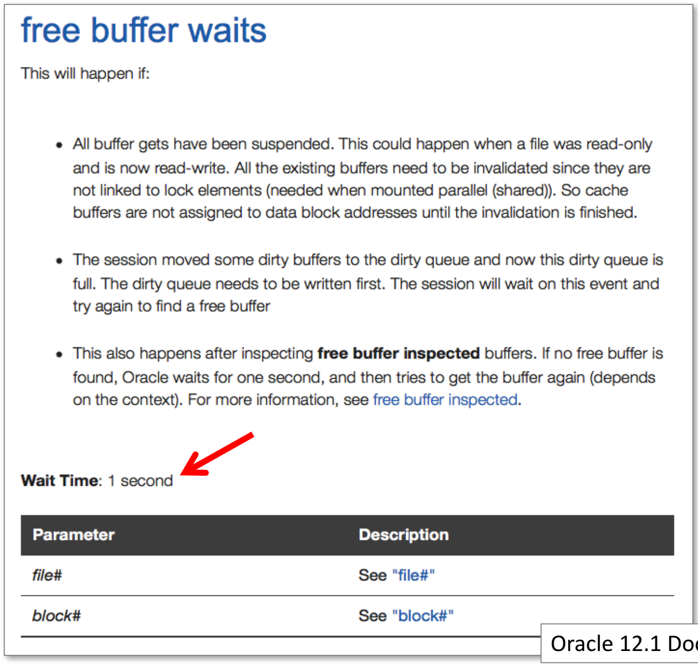
This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

12c Touch Count Algorithm

- Based on what I previously showed:
  - Three second rule is still the default
  - Promotion still occurs when a server process or the DBWR is looking for a buffer AND the buffer's TC is >= 2.
  - When promotion occurs, the TC is still reset to 0.
- Summary: operation remains the same.
- I did not test if the max 255 tch "rule" was in affect in 11g.

So, how long are the wait times... really

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



free buffer waits

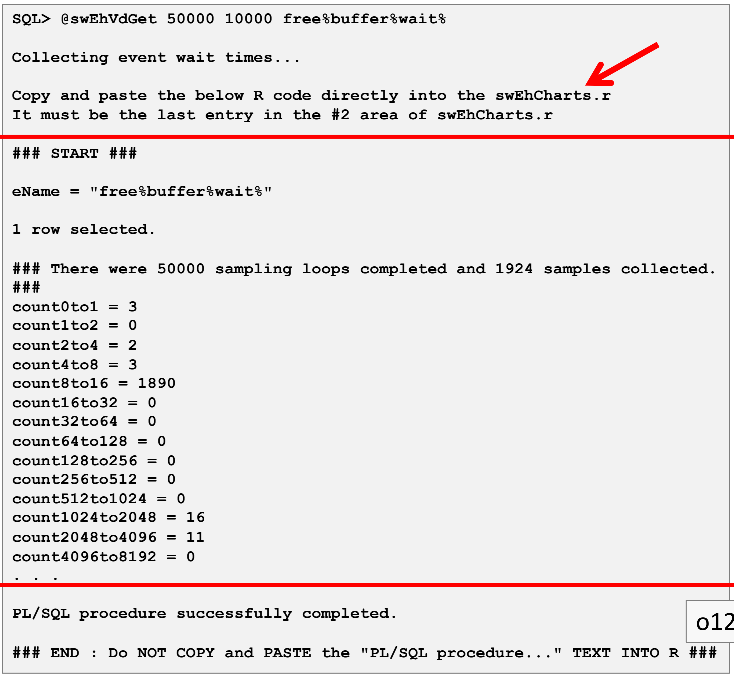
This will happen if:

- All buffer gets have been suspended. This could happen when a file was read-only and is now read-write. All the existing buffers need to be invalidated since they are not linked to lock elements (needed when mounted parallel (shared)). So cache buffers are not assigned to data block addresses until the invalidation is finished.
- The session moved some dirty buffers to the dirty queue and now this dirty queue is full. The dirty queue needs to be written first. The session will wait on this event and try again to find a free buffer
- This also happens after inspecting **free buffer inspected** buffers. If no free buffer is found, Oracle waits for one second, and then tries to get the buffer again (depends on the context). For more information, see free buffer inspected.

Wait Time: 1 second

Parameter	Description
file#	See "file#"
block#	See "block#"

Oracle 12.1 Documentation

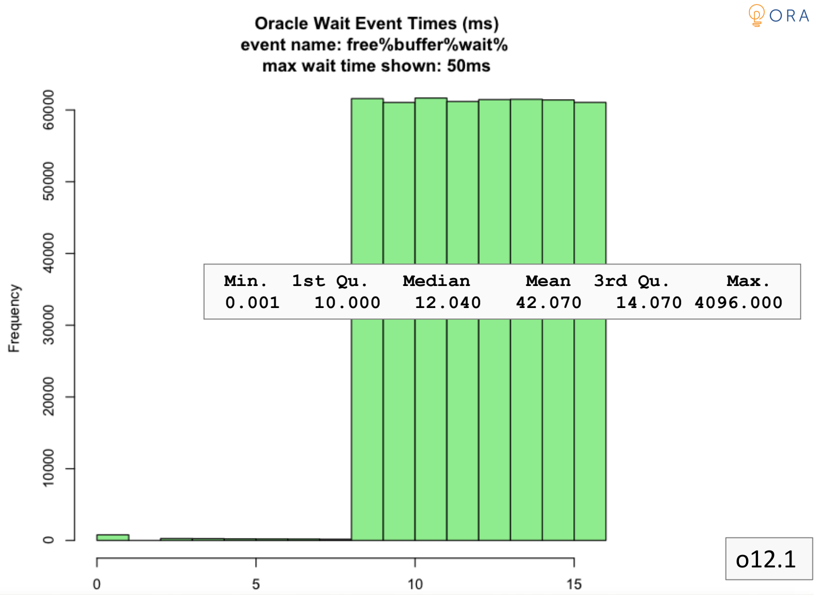


```
SQL> @swEhVdGet 50000 10000 free%buffer%wait%

Collecting event wait times...

Copy and paste the below R code directly into the swEhCharts.r
It must be the last entry in the #2 area of swEhCharts.r

### START ###

eName = "free%buffer%wait%"

1 row selected.

### There were 50000 sampling loops completed and 1924 samples collected.
###
count0to1 = 3
count1to2 = 0
count2to4 = 2
count4to8 = 3
count8to16 = 1890
count16to32 = 0
count32to64 = 0
count64to128 = 0
count128to256 = 0
count256to512 = 0
count512to1024 = 0
count1024to2048 = 16
count2048to4096 = 11
count4096to8192 = 0
. . .

PL/SQL procedure successfully completed.

### END : Do NOT COPY and PASTE the "PL/SQL procedure..." TEXT INTO R ###
```

o12.1

(c)OraPub, Inc. ORAPUB Maximizing BC Throughput

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

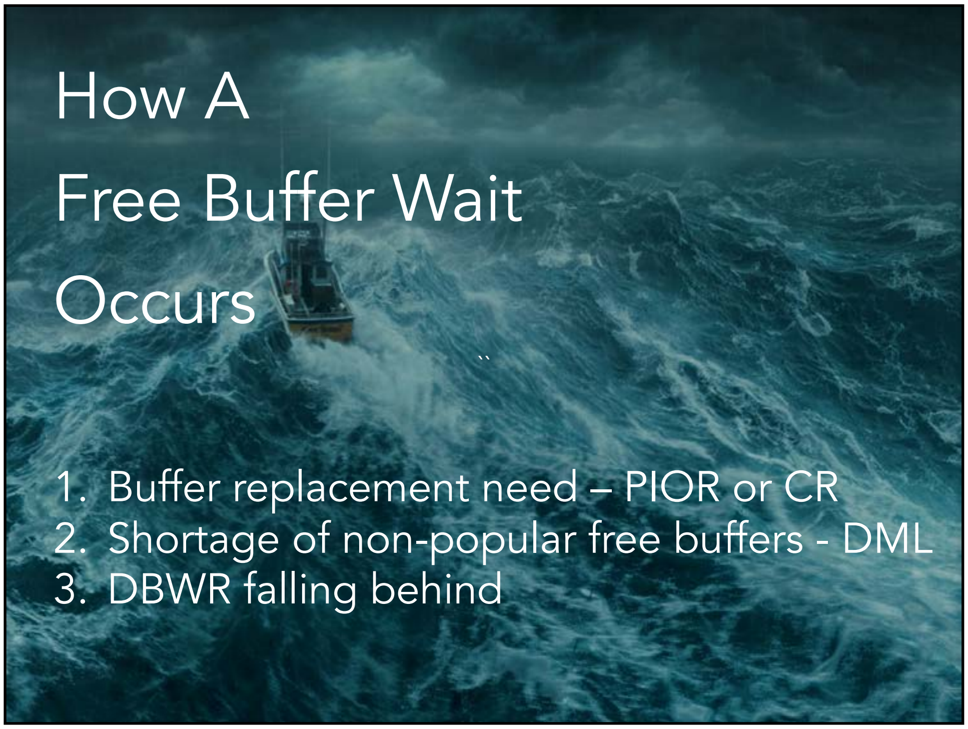


```
Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.001   10.000   12.040   42.070   14.070 4096.000
```

o12.1

Data collection: OSM, **@swEhVdGet 50000 10000 free%buffer%wait%**
Statistics: **OSM, swEhCharts.r and R statistics package**

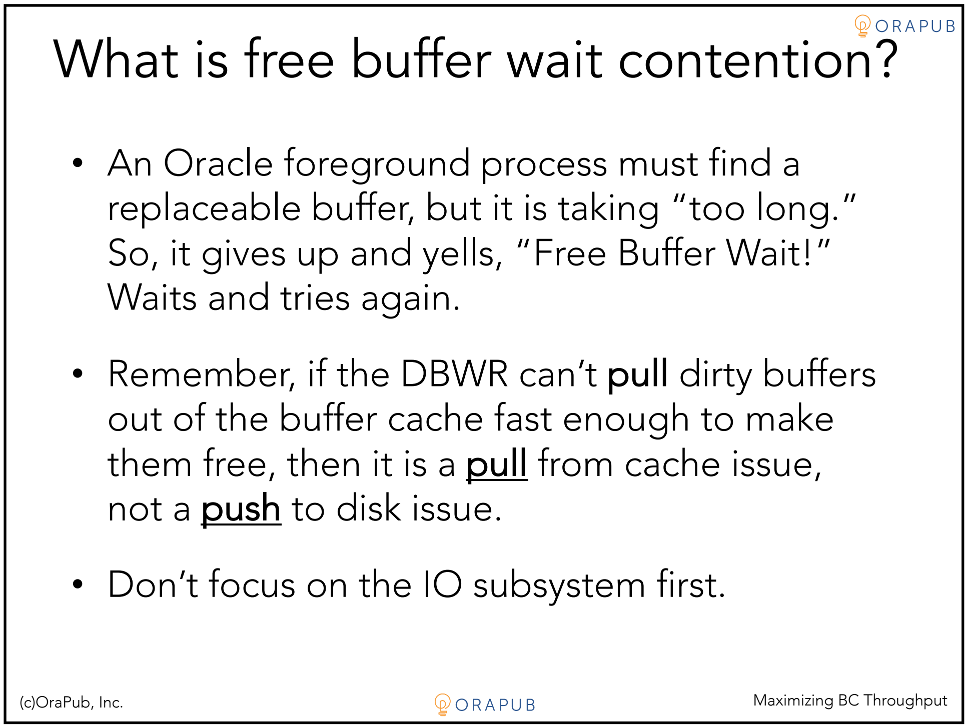
(c)OraPub, Inc. ORAPUB Maximizing BC Throughput



How A Free Buffer Wait Occurs

1. Buffer replacement need – PIOR or CR
2. Shortage of non-popular free buffers - DML
3. DBWR falling behind

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

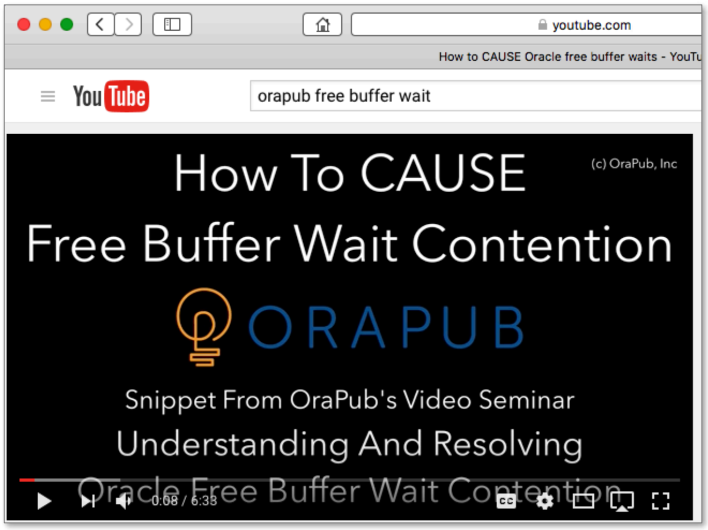


ORAPUB

What is free buffer wait contention?

- An Oracle foreground process must find a replaceable buffer, but it is taking "too long." So, it gives up and yells, "Free Buffer Wait!" Waits and tries again.
- Remember, if the DBWR can't **pull** dirty buffers out of the buffer cache fast enough to make them free, then it is a pull from cache issue, not a push to disk issue.
- Don't focus on the IO subsystem first.

Watch how easy it is to cause FBWs


youtube.com

How to CAUSE Oracle free buffer waits - YouTu

orapub free buffer wait

How To CAUSE

Free Buffer Wait Contention

(c) OraPub, Inc

ORAPUB

Snippet From OraPub's Video Seminar

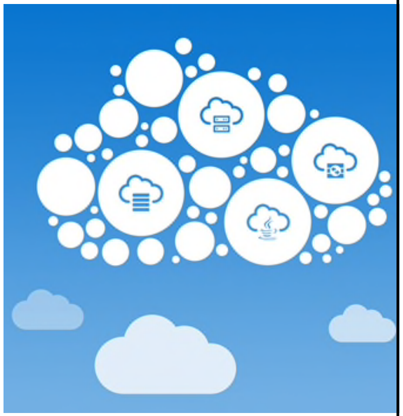
Understanding And Resolving

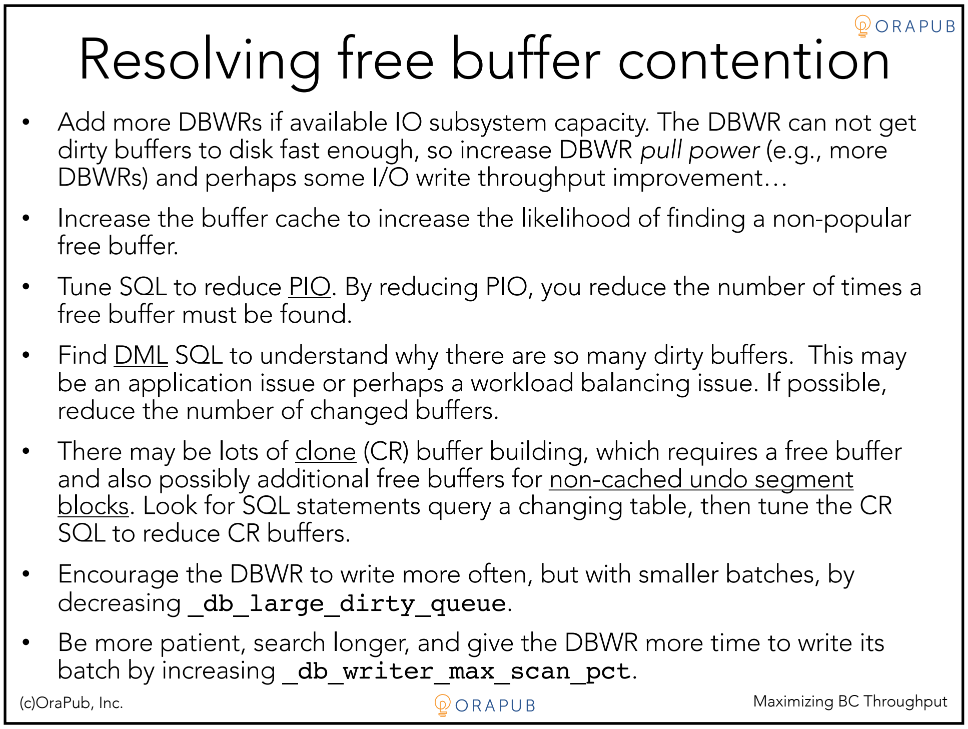
Oracle Free Buffer Wait Contention

https://youtu.be/pc2Wwi1cYAc

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Solutions To Resolve The Issue

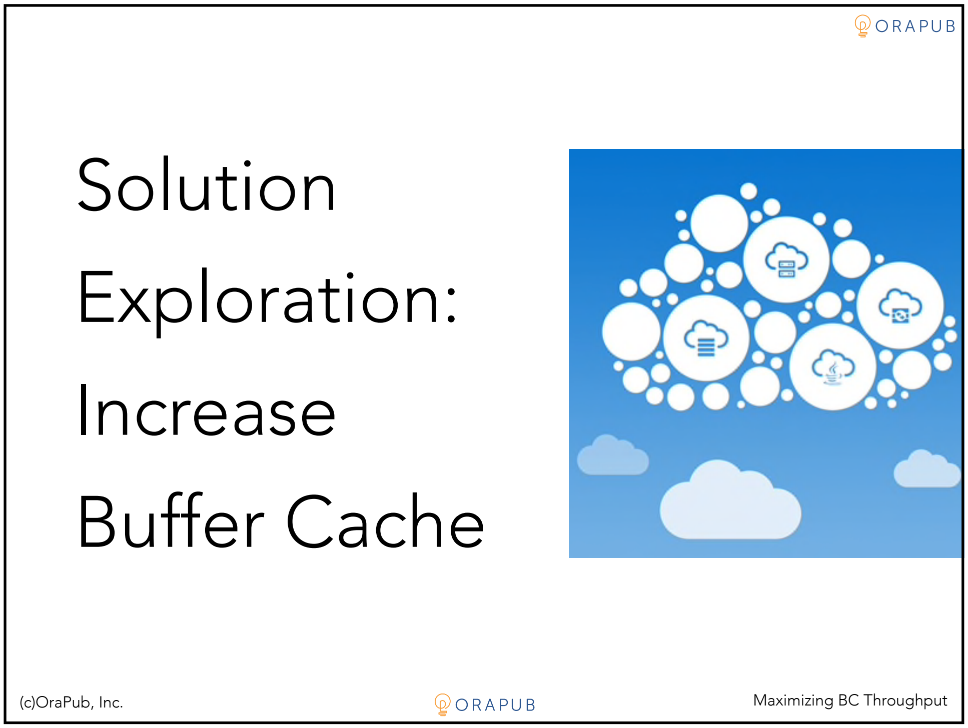




Resolving free buffer contention

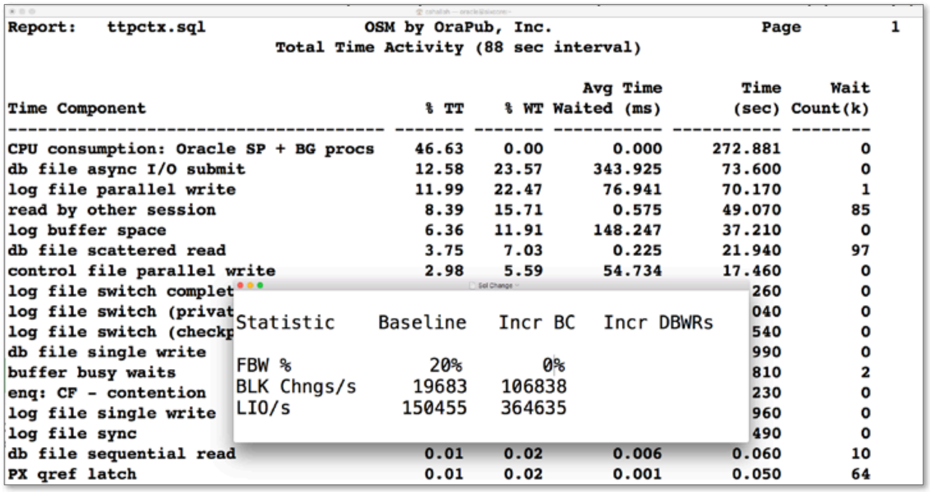
- Add more DBWRs if available IO subsystem capacity. The DBWR can not get dirty buffers to disk fast enough, so increase DBWR *pull power* (e.g., more DBWRs) and perhaps some I/O write throughput improvement…
- Increase the buffer cache to increase the likelihood of finding a non-popular free buffer.
- Tune SQL to reduce PIO. By reducing PIO, you reduce the number of times a free buffer must be found.
- Find DML SQL to understand why there are so many dirty buffers. This may be an application issue or perhaps a workload balancing issue. If possible, reduce the number of changed buffers.
- There may be lots of clone (CR) buffer building, which requires a free buffer and also possibly additional free buffers for non-cached undo segment blocks. Look for SQL statements query a changing table, then tune the CR SQL to reduce CR buffers.
- 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`.

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



Solution Exploration: Increase Buffer Cache

Increasing the buffer cache

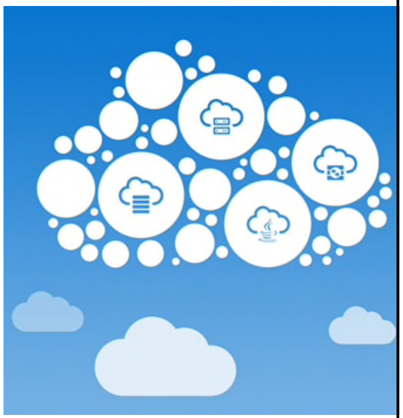


```
Report:    ttpctx.sql                  OSM by OraPub, Inc.                       Page           1
                               Total Time Activity (88 sec interval)

                                                                   Avg Time          Time     Wait
Time Component                               % TT     % WT Waited (ms)         (sec) Count(k)
-------------------------------------- ------- -------- ----------- ------------ --------
CPU consumption: Oracle SP + BG procs    46.63     0.00       0.000      272.881        0
db file async I/O submit                 12.58    23.57     343.925       73.600        0
log file parallel write                  11.99    22.47      76.941       70.170        1
read by other session                     8.39    15.71       0.575       49.070       85
log buffer space                          6.36    11.91     148.247       37.210        0
db file scattered read                    3.75     7.03       0.225       21.940       97
control file parallel write               2.98     5.59      54.734       17.460        0
log file switch complet                                                      260        0
log file switch (privat                                                      040        0
log file switch (checkp                                                      540        0
db file single write                                                         990        0
buffer busy waits                                                            810        2
enq: CF - contention                                                         230        0
log file single write                                                        960        0
log file sync                                                                490        0
db file sequential read                   0.01     0.02       0.006        0.060       10
PX qref latch                             0.01     0.02       0.001        0.050       64
```

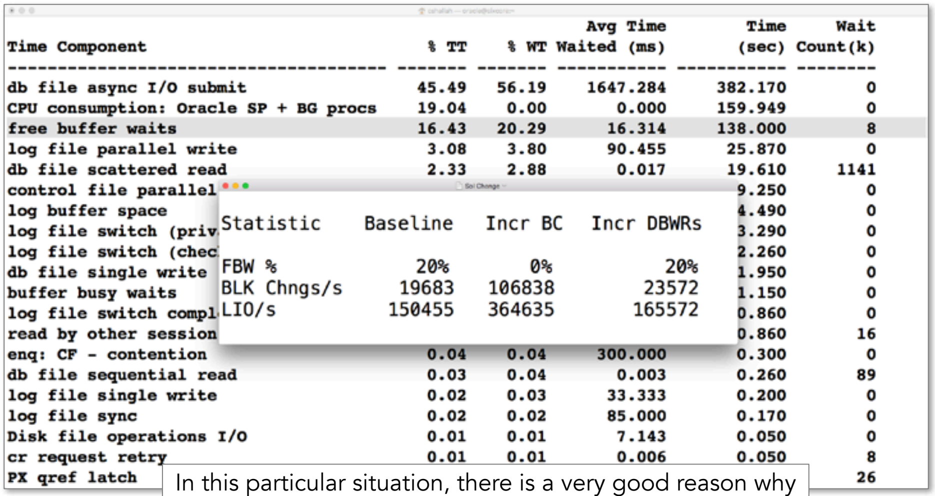
```
Statistic      Baseline    Incr BC    Incr DBWRs

FBW %               20%         0%
BLK Chngs/s       19683     106838
LIO/s            150455     364635
```

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Solution Exploration: Add More DBWRs



Add more DBWRs



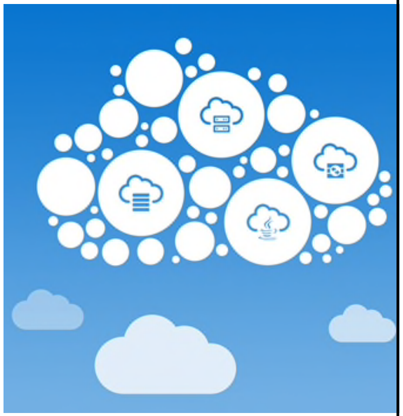
```
                                                           Avg Time          Time      Wait
Time Component                             % TT     % WT Waited (ms)        (sec)  Count(k)
---------------------------------------- -------- -------- ----------- ------------ --------
db file async I/O submit                    45.49    56.19    1647.284      382.170        0
CPU consumption: Oracle SP + BG procs       19.04     0.00       0.000      159.949        0
free buffer waits                           16.43    20.29      16.314      138.000        8
log file parallel write                      3.08     3.80      90.455       25.870        0
db file scattered read                       2.33     2.88       0.017       19.610     1141
control file parallel                                                          9.250        0
log buffer space                                                               4.490        0
log file switch (priv                                                          3.290        0
log file switch (chec                                                          2.260        0
db file single write                                                           1.950        0
buffer busy waits                                                              1.150        0
log file switch compl                                                          0.860        0
read by other session                                                          0.860       16
enq: CF - contention                         0.04     0.04     300.000        0.300        0
db file sequential read                      0.03     0.04       0.003        0.260       89
log file single write                        0.02     0.03      33.333        0.200        0
log file sync                                0.02     0.02      85.000        0.170        0
Disk file operations I/O                     0.01     0.01       7.143        0.050        0
cr request retry                             0.01     0.01       0.006        0.050        8
PX qref latch                                                                              26
```

```
Statistic     Baseline    Incr BC   Incr DBWRs

FBW %              20%         0%          20%
BLK Chngs/s      19683     106838        23572
LIO/s           150455     364635       165572
```

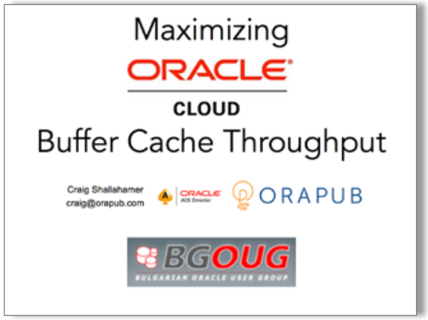
In this particular situation, there is a very good reason why adding DBWRs did not increase throughput.

This presentation was given by Craig Shallahamer (craig@orapub.com) at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Moving To The Next Level



Download the FULL presentation.



Maximizing ORACLE CLOUD Buffer Cache Throughput

Craig Shallahamer
craig@orapub.com

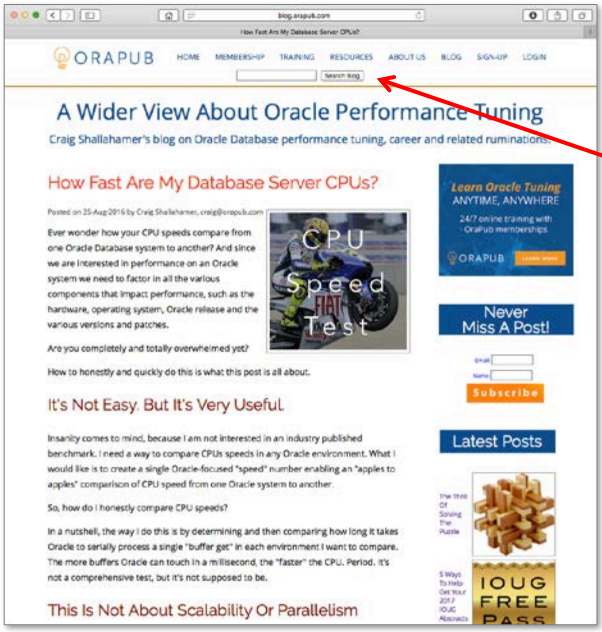
ORAPUB

BGOUG
BULGARIAN ORACLE USER GROUP

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.

Resource Listing

- OraPub Membership for premium content
  - "How To" Webinars – two each month
  - Video Seminars – any device, any time, high quality
  - Community SLACK team
  - Learning paths, mentoring, assessments and certificates, priority response
- Live Virtual Training – Multiple 2 hours sessions with daily break
  - Oracle Tuning Fastpath
  - Oracle Buffer Cache Performance Analysis & Tuning
  - Tuning Oracle Using Advanced ASH Strategies
  - Tuning Oracle Using An AWR Report
  - Oracle Predictive Analysis Using Linear Regression
- **Craig's Blog & Website** – Search: "lru", "free buffer"
- **Toolkits** – OSM and BloodHound
- **Presentations** – Search OraPub.com: "lru","free buffer", etc.
- Books
  - Oracle Performance Firefighting.
  - Forecasting Oracle Performance.



A Wider View About Oracle Performance Tuning

Craig Shallahamer's blog on Oracle Database performance tuning, career and related ruminations.

How Fast Are My Database Server CPUs?

Posted on 25-Aug-2016 by Craig Shallahamer, craig@orapub.com

Ever wonder how your CPU speeds compare from one Oracle Database system to another? And since we are interested in performance on an Oracle system we need to factor in all the various components that impact performance, such as the hardware, operating system, Oracle release and the various versions and patches.

Are you completely and totally overwhelmed yet?

How to honestly and quickly do this is what this post is all about.

It's Not Easy. But It's Very Useful.

Insanity comes to mind, because I am not interested in an industry published benchmark. I need a way to compare CPUs speeds in any Oracle environment. What I would like is to create a single Oracle-focused "speed" number enabling an "apples to apples" comparison of CPU speed from one Oracle system to another.

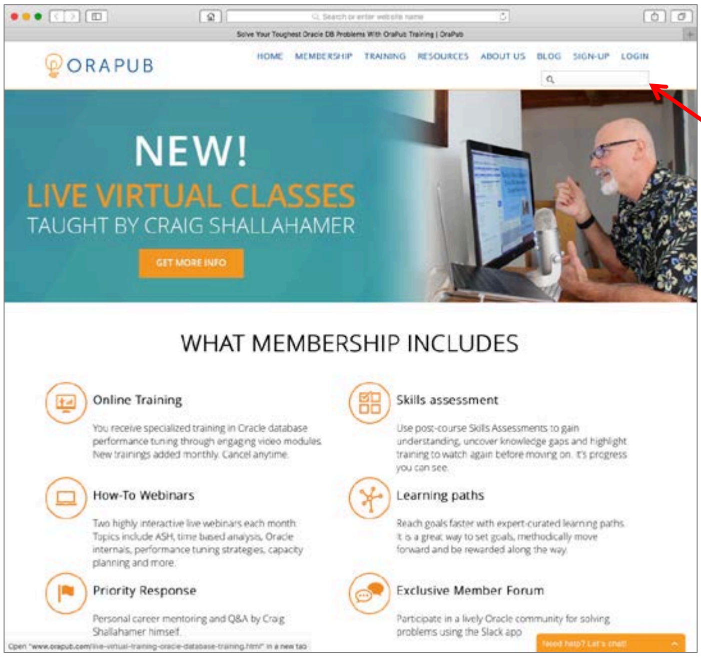
So, how do I honestly compare CPU speeds?

In a nutshell, the way I do this is by determining and then comparing how long it takes Oracle to serially process a single "buffer get" in each environment I want to compare. The more buffers Oracle can touch in a millisecond, the "faster" the CPU. Period. It's not a comprehensive test, but it's not supposed to be.

This Is Not About Scalability Or Parallelism

lru,
free
buffer,
visual

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.


Search or enter website name
Solve Your Toughest Oracle DB Problems With OraPub Training | OraPub
ORAPUB
HOME
MEMBERSHIP
TRAINING
RESOURCES
ABOUT US
BLOG
SIGN-UP
LOGIN
NEW!
LIVE VIRTUAL CLASSES
TAUGHT BY CRAIG SHALLAHAMER
GET MORE INFO
Need help? Let's chat!

WHAT MEMBERSHIP INCLUDES

Online Training

You receive specialized training in Oracle database performance tuning through engaging video modules. New trainings added monthly. Cancel anytime.

Skills assessment

Use post-course Skills Assessments to gain understanding, uncover knowledge gaps and highlight training to watch again before moving on. It's progress you can see.

How-To Webinars

Two highly interactive live webinars each month. Topics include ASH, time based analysis, Oracle internals, performance tuning strategies, capacity planning and more.

Learning paths

Reach goals faster with expert-curated learning paths. It is a great way to set goals, methodically move forward and be rewarded along the way.

Priority Response

Personal career mentoring and Q&A by Craig Shallahamer himself.

Exclusive Member Forum

Participate in a lively Oracle community for solving problems using the Slack app

Open "www.orapub.com/live-virtual-training-oracle-database-training.html" in a new tab

lru,
free
buffer,
visual

ORAPUB

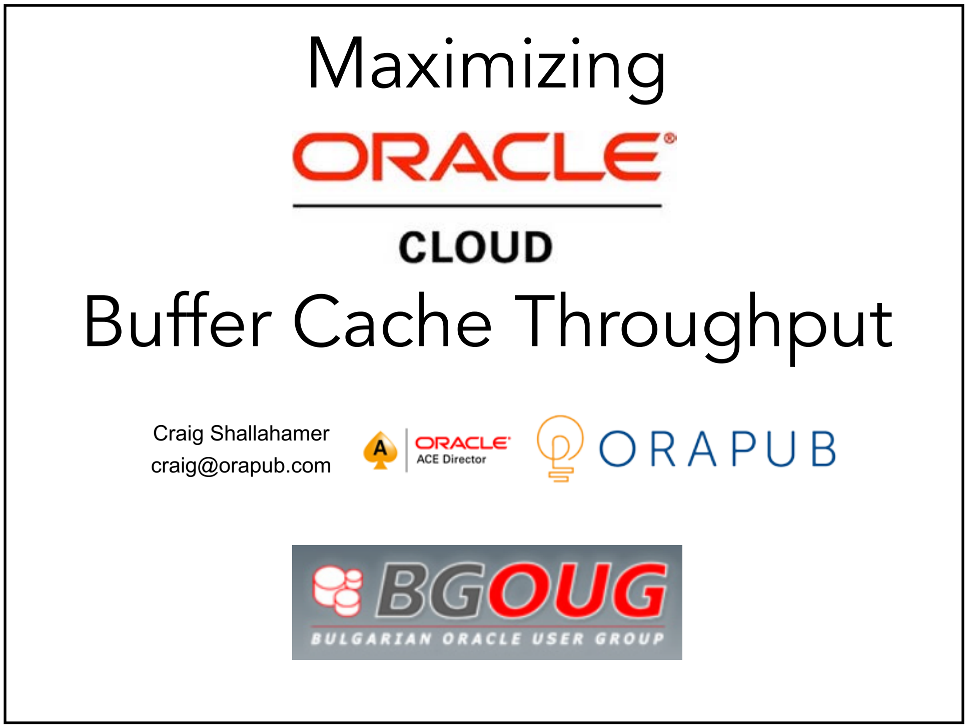
Thank You!

(c)OraPub, Inc.

ORAPUB

Maximizing BC Throughput

This presentation was given by Craig Shallahamer (craig@orapub.com)
at the November 2017 Bulgaria OUG (BGOUG) conference in Sophia, Bulgaria.



Maximizing ORACLE CLOUD Buffer Cache Throughput

Craig Shallahamer
craig@orapub.com

ORACLE ACE Director

ORAPUB

BGOUG
BULGARIAN ORACLE USER GROUP