Oracle Redo Log Performance Issues and Solutions

Sun Hongda

Abstract

The redo log plays a prime role in Oracle database's core functionality. However it imposes disk i/o for the redo log's inherent functionality, increased i/o can highly affect the Oracle performance. In this paper I am analyzing the performance issues related to the Oracle redo log, and solutions to address those issues, also I am covering how to minimize the overhead of redo logs and improve the overall database performance.

Concept review

The redo log buffer is a RAM area defined by the initialization parameter log_buffer that works to save changes to database, in case something fails and Oracle has to put it back into its original state (a "rollback"). When Oracle SQL updates a table, redo images are created and stored in the redo log buffer. Since RAM is faster than disk, this makes the storage of redo very fast.

Oracle will eventually flush the redo log buffer to disk, called online redo log. This can happen in a number of special cases, but what's really important is that Oracle guarantees that the redo log buffer will be flushed to disk after a commit operation occurs. When you make changes in the database you must commit them to make the changes permanent and visible to other users.

In case of Oracle's Archive log mode , online redo can be achieved and called as offline or achieved redo log ., In case of No archive log mode Oracle writes in rotation through its redo log groups, it will eventually overwrite a group which it has already written to. Data that is being overwritten would of course be useless for a recovery scenario. In order to prevent that, oracle has introduced the archive log mode,. In case of archive log mode, Oracle makes sure that online redo log files are not overwritten unless they have been safely archived at some other place in disk. Some key points are -

- A database can be recovered from media failure only if it runs under archive log mode.
- Oracle's background process LGWR process is responsible to write the redo log buffers to disk.
- Oracle's background process ARCn is in-charge for archiving redo logs , if automatic archiving is enabled by keeping achieve log mode On. All changes that are covered by redo are first written into the log buffer (RAM). The idea of storing redo log in the RAM is to reduce disk I/O. Of-course, when a transaction commits, the redo log buffer must be flushed to disk, otherwise the recovery for that commit could not be guaranteed. It is LGWR (Log Writer) process which is responsible for redo log buffer flushing into disk .

Issues

Excessive redo log generation is one of the potential issue related to redo log, Impact of excessive redo generation is very high and impacted many areas;

- Higher CPU usage - Generating redo records and copying them to log buffer consumes CPU resources
- LGWR process need to work hard if the redo log generation rate is very high.
- High redo log generation resulted into very frequent log switches and might increase the checkpoint frequency which in turn slow down the overall disk performance
- In case of archive log mode,
  - Arch process leads to generate more archive log files. This again introduces additional CPU and disk usage.
  - Even backup of these archive log files uses more CPU, disk and possibly tape resources.
- Redo entries are copied in to redo buffer under the protection of various latches, and thus excessive redo log leads to increase in contention for redo latches

Another potential issue is the disk performance, which is related to the above Issus, if database generates excessive redo logs, it will slowdown the disk performance drastically. Database performance is worse in case of heavily loaded environment, more load on database server leads to more disk I / o due to inherent nature of redo logs to be written to disk.

We can take a conclusion that, the main issues of redo logs are its rate of generation and disk performance, so the key aspect of redo log performance tuning is to control excessive redo log generation and improve the disk performance.

Redo log size

Shamsudeen.A.(2008) has done a excellent research on how the various factors are affect the redo log size, In his research, Table 1 below showing factors which affect redo log size -

Number	Factor
1	the number of index
2	the number of threads
3	the type of SQL statement
4	number of columns
5	commit frequency
6	cache size

Table 1, the factors affecting redo log size(1)

Emrick.S.E.(2006) also done an excellent research about oracle business reinforcement rules affect the oracle redo log performance. He divided the factors related to business reinforcement rules affect the redo log into following categories, please refer the Table 2 below -

Number	Factor
1	the Uniqueness Enforcement through primary key or unique key constraint
2	the referential integrity constraint

Table 2 the factors which affect redo log size(2)

However their research did not conclude the most affecting factors to reduce excessive redo log generation , In order to enhance the solution I came-up with a case study based on above factors to find out the rationale of redo log size changes under different conditions , listed in below table Table3.

In order to find out the factors which can reduce the size of redo log generation among those factor listed in above research, I have summarized their results in one table and created a chart by using data in Shamsudeen.A.(2008) and ratio R in Emrick.S.E.(2006), The results are listed in table 3 with test number.

Number	Test case	Redo size
1	No indices(10000rows)	30107436
2	One index(10000rows)	38043080
3	Additional Six indices(10000rows)	202172680
4	one thread delete+insert	21735972
5	one thread update using merge	4608084
6	Delete+insert: two thread thread1	18432752
7	Delete+insert: two thread thread2	26020552
8	update using merge;thread 1	4612968
9	update using merge;thread 2	4612968
10	Update all columns except id column	419249316
11	update 3 varchar2(100)columns and 3 number columns	261210412
12	update 1 varchar2(100) columns and 3 number columns	145939212
13	commit frequency single row	7620572
14	commit frequency 10 rows	4296472
15	commit frequency 100 rows	4002344
16	commit frequency1000 rows	4217152

17	commit frequency 10000 rows	4238944
18	cache size 2mb	15419184
19	cache size 10mb	4975192
20	cache size 100mb	2628168
21	cache size 1000mb	2393420
22	Delete+insert Unique index failed transaction one thread(parent table)	52601052.24
23	Delete+insert No Unique index failed transaction one thread(parent table) fail	91725801.84
24	update 1 varchar2(100) columns and 3 number columns with unique index(parent table) fail	161992525.3
25	update 1 varchar2(100) columns and 3 number columns with Non unique index(parent table) fail	357551069.4
26	Deleted parent table and insert child table failed transaction(Unique index)	64990556.28
27	Deleted parent table and insert child table failed transaction(non Unique index)	592365952.4
28	update 1 varchar2(100) columns and 3 number columns with unique index(child table) fail	210152465.3
29	update 1 varchar2(100) columns and 3 number columns with non-unique index(child table) fail	103616840.5

Table 3, the test case of redo log size

As illustrated in Figure 1, we can see the different cases of redo log generation are changing under different conditions,

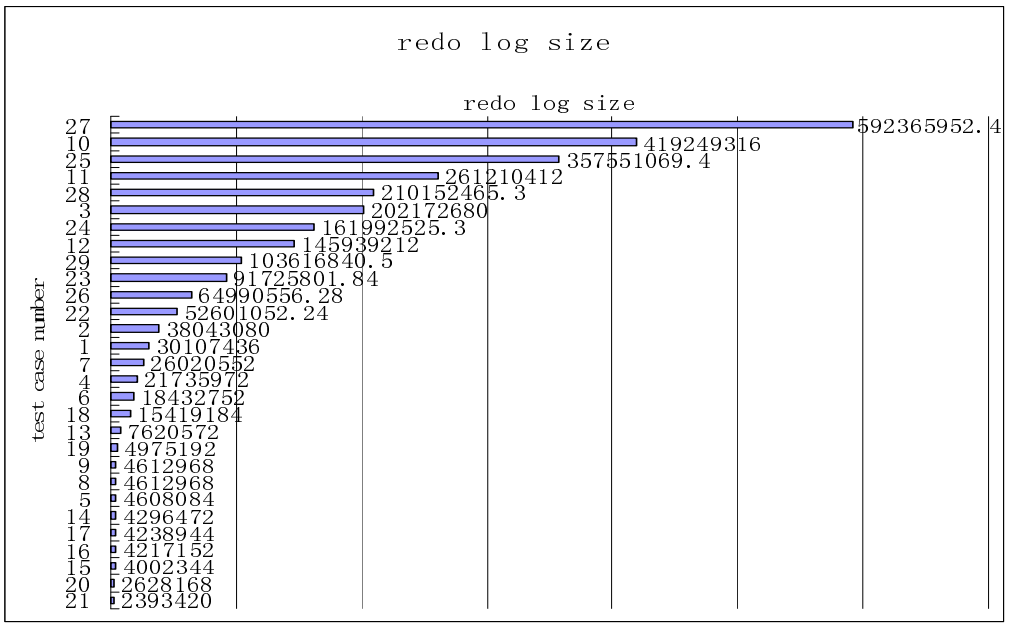

Figure 1, the test case of redo log size

To conclude, I have summarizes top five test cases which generate less redo log in table 5.

Number	Test case	Redo size
21	cache size 1000mb	2393420
20	cache size 100mb	2628168
15	commit frequency 100 rows	4002344
16	commit frequency1000 rows	4217152
17	commit frequency 10000 rows	4238944

Table 5 top five test case reduce redo log size

The Table 5 shows that large cache size cab reduce the corresponding redo log generation, as test case number 21 has lowest redo log generation due to large cache size. Another factor commit frequency is also a key factors to reduce redo log generation, compared to the test case number 15 test case number 16 has higher commit frequency and lower redo log generation, which indicated high commit frequency can reduces the redo log generation.

Reliability and Performance

From operating system point of view, the only issue related to redo log is tradeoff between system reliability and system performance. Reliability is the key aspect of an operating system, thus RAID technology has been used to provide the high reliability solutions. However RAID will increase the disk I/O which is already been at higher side due to oracle redo log maintenance. It is vital to analyze which level of RAID can yield better write performance.

RAID Level	Read Performance	Write Performance	Fault Tolerance	Cost
RAID 0	Good	Good	None	Low
RAID 1 and RAID 0+1	Good	OK 1 logical write = 2 physical I/Os	Excellent Can potentially tolerate the loss of multiple disks	Highest Requires that you purchase 2x disk drives
RAID 5	Good	Poor 1 logical write = 4 physical I/Os (2 then 2)	OK Can survive the loss of 1 disk at severely degraded performance level	Best for fault tolerance

Table6, Compare different RAID types source: Whalen(2005)

Whalen(2005) has outlined performance effect under the different levels of RAID. The Table 6 shows that RAID 5 has best of all fault tolerance. However, RAID 5 have very poor write performance. Furthermore, RAID 0 can be ignored as it cannot provide Fault Tolerance. Amongst all RAID systems .By the above Table 6 it has been concluded that the RAID 1 or RAID 0+1 can achieve optimal redo log performance in addition to good system reliability.

Faster disk or more memory

In last twenty years processor speed has been increased exponentially, but at the same time, conventional storage access time only improved marginally, resulted is a massive performance gap between the two. Concerning the same many solutions has been proposed aiming to close the gaps between the two. One of the well accepted solution is the use of Solid State Disks instead of conventional disk media.

Hutsell,W and Mike.A.(2009) defines that a Solid State Disk(or SSD) is a storage device that does not rely on mechanical parts for reading from and writing data to disk . However, they use memory (DDR or Flash) as the primary storage media. This generally result in storage speeds far greater and getting the same speed practically impossible to achieve by using conventional magnetic storage devices, they have done a very good research which is shown below in Figure 2 below -

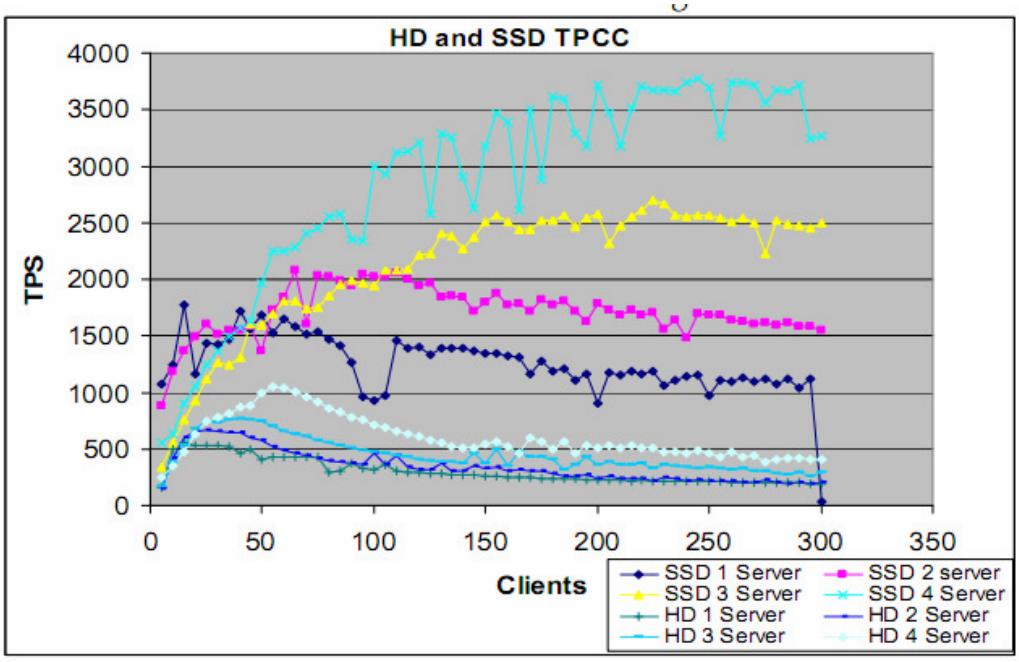

Figure 2, HD and SSD TPCC **Source: Hutsell,W and Mike.A.(2009)**

As we can spot in the Figure 2, the SSD Server is faster than HD server. It shows SSD 4 Server has highest TPS, while the number of client reaching 300 and HD2 Server has lowest TPS at the same time. There are certain other alternative solutions available in market to achieve the similar disk performance as SSD. One of solution is adding faster disk chains to the computer, the other solution is to increase the disk buffer in memory. Matthew .Y.(2008) has done the research to compare the performance of those solutions

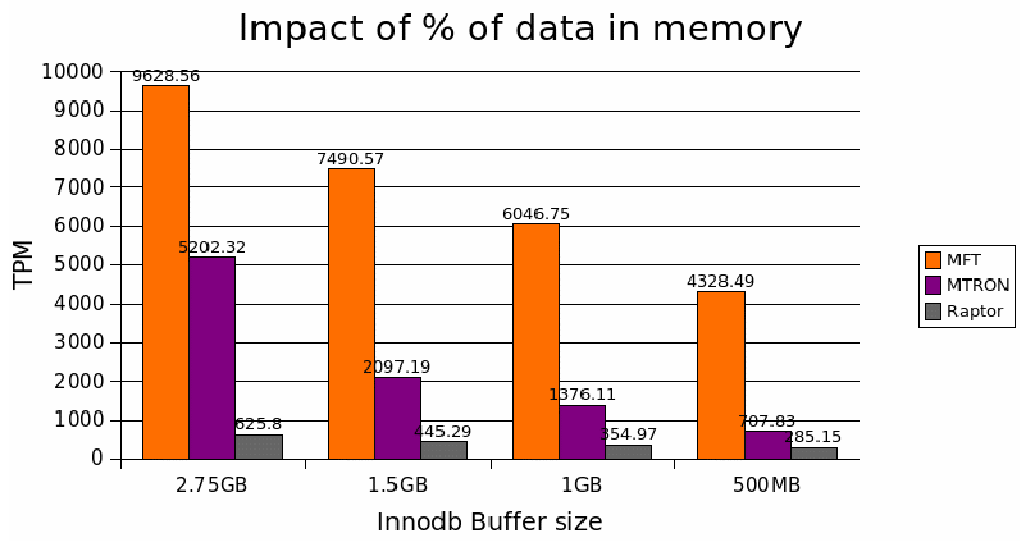

Figure 3, Innodb Buffer size **source: Matthew .Y.(2008)**

The Figure 3 shows data buffer and different type of disk's effect on disk TPM. The disk of MFT and MTRON has more number of disks. Also, these both type of disks are faster than Raptor. Under the same disk Raptor, while buffer size is increasing from 500MB to 1GB, the TPM of disk is increasing from 285.15 to 354.97, hence Disk TPM is increasing In case of more memory addition.

If the type of disk changes from Raptor to MFT , it will sharply increasing TPM from 285.15 to 4328.49 i.e. 15x faster than Raptor. However, by increasing memory up to 2.75GB increases Disk TPM from 285.15 to 625.8 i.e. only 2x faster.

The Figure 3 is also indicates if Disk Raptor is changed to MFT, and the innodb Buffer size become 1GB, the disk TPM increases from 285.15 to 6046.75, i.e. 21x faster. It conclude that the faster disk is more important in order to speed up disk performance. But at the same time it is even better to add more memory as well . For the system engineers to improve the disk i/o performance , recommended solution in priority order given below -

1. Faster disk with higher main memory
2. Faster disk

Summary

In order to reduce the redo log generation, DBAs can increase the main memory cache size, one should also increase the commit frequency to reduce the disk i/o. However, for the faster i/o, DBAs can add more disk chains, for an economic solution, if cost is not the factor then it is suggested to use SSD disks. It is also recommended that one should not use the RAID 5 for system reliability as it will marginally lower the disk performance.

Reference

Shamsudeen.A.(2008) redo internals and tuning by redo reduction[Electronic version]. retrieved May 10,2010 from http://orainternals.files.wordpress.com/2008/07/riyaj_redo_internals_and_tuning_by_redo_reduction_doc.pdf

Emrick.S.E.(2006) The Oracle Redo Generation [Electronic version]. Retrieved May 25,2010 from http://hosteddocs.ittoolbox.com/EE010306.pdf

Craig .S (2007). *Forecasting Oracle Performance*. New York: Apress

Matthew .Y.(2008) Common Performance Mistake; Disk [Electronic version]. Retrieved June 10,2010 from http://www.bigdbahead.com/?p=49

Confio software .(2009) Log File Switch Completion [Electronic version]. retrieved June 1,2010 from http://www.confio.com/English/Tips/Log_File_Switch_Completion.php

George(2007) Comprehensive RAID performance report [Electronic version]. retrieved June 10,2010 from http://www.zdnet.com/blog/ou/comprehensive-raid-performance-report/484?pg=3

Whalen(2005) Overview of I/O Performance and RAID in an RDBMS Enviroment retrieved June 10,2010 from http://www.perftuning.com/pdf/RAID1.pdf

Hutsell,W and Mike.A.(2009) Faster Oracle Performance with Solid State Disks retrieved May 1,2010 http://www.texmemsys.com/files/f000139.pdf

ACKNOWLEDGEMENTS

I greatly appreciate the guidance, valuable feedback and insight from Shailesh Paliwal , Denise Kalm for all her encouragement.

Author's Profile

Sun Hongda is Cisco Certified Network Associate, Certified Internet Web Security Analyst, Certified Internet Web (CIW) professional, MSc in Networking and Web Technology (UK).He is working in TAIJI Computer Corporation Company limited, E-government research center, which is The Leading supplier of government IT-service in CHINA. He works as system engineer and mainly focuses on Web system performance and reliability.