Super-Sizing PGA Workareas

Martin Decker, 17.09.2017, martin.decker@ora-solutions.net

Caution

The content from this paper only describes my findings. It may well be that I am not seeing the whole picture or that the proposed configuration has negative side effects not mentioned here. If you have anything to add or correct, please feel free to contact me directly.

Introduction

Todays hardware infrastructure is becoming more and more powerful and affordable. Quite often, the servers are equipped with several hundreds of Gigabytes of memory, if not even Terabytes. While the Oracle Database can use the memory for caching data blocks quite effectively, using the memory for PGA workareas, e.g. sort operations or hash joins, is more difficult. The following discussion is limited to PGA workareas. There are other consumers of PGA memory, e.g. PL/SQL collections, etc. Those are not in scope of this discussion.

There are two alternatives for using PGA memory: manual and automatic workarea management. Whereas the first one is practically useless because it is too difficult to configure optimally, the second is quite easy to set up. There is practically a single parameter, pga_aggregate_target, which controls the PGA memory for all the processes in total. This is common knowledge as this has not changed since the introduction in version 9i. But only few DBAs are aware of the limits for non-parallel queries and how much memory such serial queries can get from the huge total PGA memory.

Workarea Size Policies

Manual Workarea Size Policy

The behaviour of the manual workarea policy can be controlled by these main parameters, which can be set on instance or session level.

workarea_size_policy=manual
sort_area_size=<integer between 1 and 2GB-1>
hash_area_size=<integer between 1 and 2GB-1>

There is a little outdated, but still excellent whitepaper called "If your memory serves you right" by Richmond Shee, which explains why sometimes a bigger manual value for "sort_area_size" has negative impact on response time. The issue he describes makes it very hard to manually set the optimal size for the parameter, therefore making the manual workarea size policy disadvantageous, although I am not sure if the algorithms described in the paper are still in use.

Automatic Workarea Size Policy

Parameter	Description
pga_aggregate_target	Target size for the aggregtate PGA memory consumed by the instance
pga_aggregate_limit	
_pga_max_size	Maximum size of the PGA memory for one process
_smm_max_size	maximum work area size in auto mode (serial) in KB
_smm_px_max_size	maximum work area size in auto mode (global) in KB

The following table shows how different values for pga_aggregate_target result in different values for the underscore parameters listed. I have tested with releases 12.1.0.2 and 12.2.0.1.

pga_aggregate_target [MB]	_pga_max_size [MB]	_smm_max_size [MB]	_smm_px_max_size [MB]
1024	205	102	512
2048	410	205	1024
3072	614	307	1536
4096	819	410	2048
8192	1638	819	4096
16384	2048	1024	8192
32768	2048	1024	16384
65536	2048	1024	32768
98304	2048	1024	49152
131072	2048	1024	65536
163840	2048	1024	81920
196608	2048	1024	98304
524288	2048	1024	262144
4194304	2048	1024	2097152

Things to note:

- In Version 12.2.0.1, there is a protection which makes it impossible to size pga_aggregate_target to higher values than physical memory. *(ORA-00855: PGA_AGGREGATE_TARGET cannot be set because of insufficient physical memory.)* Version 12.1.0.2 does not have the limit to protect from overprovisioning memory. The maximum possible value for pga_aggregate_target is therefore 4096G-1. Even then, the parameter _smm_px_max_size is set to 50% of pga_aggregate_target.

- Starting with pga_aggregate_target = 16G, the values _pga_max_size and _smm_max_size reach their maximum values with 2GB and 1 GB respectively.

- The parameter _smm_px_max_size does not have an artificial maximum value but is always 50% of pga_aggregate_target.

This shows that a single work area is limited to 1GB and all the workareas of a single process in total are limited to 2 GB even in release 12.2.0.1.

While for parallel queries, this may not be a big issue, because when a very high degree of parallelism is used, up to 50% of pga_aggregate_target can be exploited, the problem is with serial queries or Standard Edition 2, where parallel query is not available.

What are the options to increase these limits?

_pga_max_size

When this parameter is increased, it is important to configure operating system parameters that are required for increase as well. There are 2 alternatives for this:

a. _pga_max_size - OS preparation

For Linux, it is necessary to change a parameter in sysctl.conf:
/etc/sysctl.conf:
vm.max_map_count (default 65535)

This parameter results in 65535 * 65535 = 4GB of process memory. If _pga_max_size needs to be increased, this parameter should be increased as well.

b. _pga_max_size - Oracle parameter preparation

If there is no chance to modify operating system parameter, then the oracle parameter

_realfree_heap_pagesize (12c) (default 65535)

can be increased.

So, to be able to increase _pga_max_size to 16GB, we could increase _realfree_heap_pagesize to 262144 or to set operating system parameter vm.max_map_count to 262144.

_smm_max_size

The parameter _smm_max_size defaults to 50% of _pga_max_size, but the tests show that there is a hard limit at 4GB. Even when the displayed value of _smm_max_size is shown as bigger than 4 GB, the workareas are spilled to disk when the 4GB limit is reached. Therefore, it hardly makes any sense to set _pga_max_size bigger than 16 GB, when a single work area can only hold 4 GB.

The following testcases show, how setting the hidden parameter _pga_max_size is affecting memory/disk sort operations.

Testcases:

In the following testcases, different configurations for pga_aggregate_target and _pga_max_size are used.

Testcase 1	p_a_t = 64GB, defaults for _pga_max_size / _smm_max_size
Testcase 2	p_a_t = 64 GB, _pga_max_size = 4GB, default for _smm_max_size
Testcase 3	p_a_t 64 GB, _pga_max_size = 8 GB, default for _smm_max_size
Testcase 4	p_a_t = 64 GB, _pga_max_size = 16 GB, default for _smm_max_size
Testcase 5	p_a_t = 19G, _pga_max_size = 8G, default for _smm_max_size

Testcase Preparations

I am using a test table with 26 GB data in one column with random data and the test query is using a sort operation. Then I am using the "SAMPLE(x)" Option to return only a specific subset of the data for the sort.

Query:
select /*+ FULL(r) NOPARALLEL(r) */ * FROM random_strings SAMPLE(&1) r ORDER BY 1;

Testcase 1: p_a_t = 64 GB, defaults for _pga_max_size / _smm_max_size

```
NAME                          Value  DESCRIPTION
-------------------- -------------- --------------------------------------------------------------------------
pga_aggregate_target    68719476736 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size            2147483648 Maximum size of the PGA memory for one process
_smm_max_size               1048576 maximum work area size in auto mode (serial)
_smm_px_max_size           33554432 maximum work area size in auto mode (global)
```

It can be observed, that the derived values for _pga_max_size are 2 GB and _smm_max_size 1GB.

=> When using a sample size of 3% of the 26 GB table, the query is succeeding in doing a memory sort.

```
Sun Sep 17 14:21:09 CEST 2017 -- RUNNING SERIAL QUERY SAMPLE 3 PCT

OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ---------------------- ---------------------- ---------------- ----------
           1 AUTO              935191552              9993216       831281152 OPTIMAL
```

=> When using sample of 4 percent of 26 GB, the workarea limit of 1GB is reached and the sort spills to disk. (1 pass)

```
Sun Sep 17 14:21:58 CEST 2017 -- RUNNING SERIAL QUERY SAMPLE 4 PCT
OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ---------------------- ---------------------- ---------------- ----------
           1 AUTO             1499351040             12592128      1073652736 1 PASS
```

This shows that single workareas do not benefit from increasing pga_aggregate_target to values > 16GB with the default configuration of _pga_max_size / _smm_max_size.

Testcase 2: p_a_t = 64GB, _pga_max_size = 4GB, default for _smm_max_size

```
NAME                      Value  DESCRIPTION
------------------- ------------ --------------------------------------------------------------------------
pga_aggregate_target    68719476736 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size            4294967296 Maximum size of the PGA memory for one process
_smm_max_size              2097152 maximum work area size in auto mode (serial)
_smm_px_max_size          33554432 maximum work area size in auto mode (global)
```

When setting _pga_max_size to 4GB, the parameter _smm_max_size defaults to 50% of _pga_max_size automatically. (2GB).

=> The 7% Sample can avoid a disk sort and use memory-only sort operation.

```
USERNAME PROGRAM                        SQL_ID          COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ ------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   gaa5yj6nxz9am           1            1881             1881                   0           1881

       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ---------------- --------------- ------------- ---------------
       613             1850            1850             0               0

USERNAME PROGRAM                        SQL_ID          COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ ------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   gaa5yj6nxz9am           1            1881             1881                   0           1881

OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ---------------------- ---------------------- ----------------- ----------
           1 AUTO              2182265856               15143936        1939790848 OPTIMAL
```

=> Increasing the sample size to 8%, the workarea size limit is reached and the sort spills to disk. (Temp Tablespace)

```
       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ---------------- --------------- ------------- ---------------
       613                1            2048             1            2035

USERNAME PROGRAM                        SQL_ID          COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ ------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   3xrsthjvfaa46           1               3                4                   1           2083

OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ---------------------- ---------------------- ----------------- ----------
           1 AUTO              2404113408               15884288        2147390464 1 PASS
```

Testcase 3: p_a_t = 64 GB, _pga_max_size = 8GB, default for _smm_max_size

```
NAME                      Value  DESCRIPTION
------------------- ------------ --------------------------------------------------------------------------
pga_aggregate_target    68719476736 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size            8589934592 Maximum size of the PGA memory for one process
_smm_max_size               4194304 maximum work area size in auto mode (serial)
_smm_px_max_size           33554432 maximum work area size in auto mode (global)
```

Here, we are setting _pga_max_size to 8GB. The derived value for _smm_max_size is 50% of _pga_max_size. This value is displayed when the hidden parameters are queried. This is noteworthy, because when trying to set "_smm_max_size", it only accepts values up to 2GB -1.

=> When trying to use a 15% Sample size, we still succeed in doing a memory sort and we can see that in fact, the workarea used was roughly 4 GB. This is somewhat surprising because I was of the opinion, that there still is a 2 GB limit in place.

```
       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ----------------- --------------- ------------- ---------------
       613              3963            3963             0               0

USERNAME PROGRAM                        SQL_ID          COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ ------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   buq9cbcty51ym           1            4027             4028                   0           4028


OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ----------------------- ----------------------- ----------------- ----------
           1 AUTO              4675251200                22060032        4155778048 OPTIMAL
```

=> Increasing the sample size to 16%, we are reaching the limit of the workarea and the sort spills to disk.

```
       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ----------------- --------------- ------------- ---------------
       613                 1            4096             1            4068

USERNAME PROGRAM                        SQL_ID          COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ ------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   9kwu6t5wrwpyk           1               3                4                   1           4164
OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ----------------------- ----------------------- ----------------- ----------
           1 AUTO              4799130624                22347776        4294865920 1 PASS
```

Testcase 4:

```
NAME                           Value  DESCRIPTION
-------------------- -------------- --------------------------------------------------------------------------
pga_aggregate_target    68719476736 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size           17179869184 Maximum size of the PGA memory for one process
_smm_max_size               8388608 maximum work area size in auto mode (serial)
_smm_px_max_size           33554432 maximum work area size in auto mode (global)
```

Although the parameter "_smm_max_size" is shown here to be around 8GB and 50% of _pga_max_size, in reality, the workarea can not grow beyond 4 GB. At that limit, the sort is spilled to disk. This can be verified by the following test.

We repeat the two tests with sample sizes 15% and 16%.

=> 15 PCT

```
OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ----------------------- ----------------------- ----------------- -----------
           1 AUTO              4676485120                22063104        4156874752 OPTIMAL

       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ----------------- --------------- ------------- ---------------
       613              3964            3964             0               0

USERNAME PROGRAM                        SQL_ID           COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ -------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   buq9cbcty51ym           1            4029             4029                   0           4029
```

=> 16 PCT

```
       SID WORK_AREA_SIZE_MB MAX_MEM_USED_MB NUMBER_PASSES TEMPSEG_SIZE_MB
---------- ----------------- --------------- ------------- ---------------
       613                 1            4096             1            4067

USERNAME PROGRAM                        SQL_ID           COUNT(*) PGA_USED_MEM_MB PGA_ALLOC_MEM_MB PGA_FREEABLE_MEM_MB PGA_MAX_MEM_MB
-------- ------------------------------ -------------- ---------- --------------- ---------------- ------------------- --------------
MDECKER  sqlplus@serv3897 (TNS V1-V3)   9kwu6t5wrwpyk           1               3                4                   1           4164

OPERATION_ID POLIC ESTIMATED_OPTIMAL_SIZE ESTIMATED_ONEPASS_SIZE LAST_MEMORY_USED LAST_EXECU
------------ ----- ----------------------- ----------------------- ----------------- -----------
           1 AUTO              4797287424                22343680        4294865920 1 PASS
```

Again, the 15% sample can be satisfied with a memory sort, but the 16% still spills to disk. This shows that even higher values for _smm_max_size are displayed when the value is derived from _pga_max_size values higher than 8 GB, the hard limit is still 4GB.

Testcase 5: p_a_t = 19G, _pga_max_size = 8G, default for _smm_max_size

We know that the concept of automatic workarea policy is that a big pool of memory is divided between all the workareas of an instance. Therefore, the concept has to limit the amount of memory that a single workarea can take in order to not blindly overallocate memory or to starve other workarea consumers. The test shows that this value is "pga_aggregate_target" divided by 5.

In this test, we reduce the amount of pga_aggregate_target below the factor of 5 * workarea size (4GB) -> 20GB.

With "pga_aggregate_target = 20G", these are the derived values for _smm_max_size:

```
NAME                          Value  DESCRIPTION
-------------------- -------------- --------------------------------------------------------------------------
pga_aggregate_target    21474836480 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size            8589934592 Maximum size of the PGA memory for one process
_smm_max_size               4194304 maximum work area size in auto mode (serial)
_smm_px_max_size           10485760 maximum work area size in auto mode (global)
```

With pga_aggregate_target = 20G, the test with 15% sample size results in a memory sort.

With "pga_aggregate_target = 19G", this is the derived valuefor _smm_max_size.

```
NAME                          Value  DESCRIPTION
-------------------- -------------- --------------------------------------------------------------------------
pga_aggregate_target    20401094656 Target size for the aggregate PGA memory consumed by the instance
_pga_max_size            8589934592 Maximum size of the PGA memory for one process
_smm_max_size               3984588 maximum work area size in auto mode (serial)
_smm_px_max_size            9961472 maximum work area size in auto mode (global)
```

In this case, we set it to 19G and can immediately, see that the derived value for _smm_max_size has decreased as well and now the query with 15% sample results in a disk sort.

This confirms MOS Note 453540.1, which notes that you have to make pga_aggregate_target bigger or equal 5 times the required workarea!

Parallel Query

In parallel query, the task of reading the table data and sorting is distributed amog multiple parallel query slaves. Each slave has its own process and each slave can in turn use a 4 GB workarea. The parameter "_smm_max_px_size", which defaults to 50% of pga_aggregate_target specifies how much memory for the parallel operation is used in total. Lets consider this example:

Data Set to Sort: 100 GB
Workarea Size per Slave: 4GB (fixed)
pga_aggregate_target = 200G
=> Required Degree of Parallelism = 25

You can see that even with parallel query, to fully exploit the huge amount of memory of todays servers, one needs a very very high degree of parallelism. It is also important to consider the amount of CPU cores, that are available, when deciding on a degree of parallelism.

Conclusion

For users of Oracle Database - Standard Edition 2 or non-parallel sort operations, one can delay the point of spilling to disk and therefore causing a slower disk sort, by increasing parameter "_pga_max_size" after having considered operating system or realfree parameter increase. If "_pga_max_size" is increased to 8 GB, the single work area (_smm_max_size) can be 4 GB. This is double the value which can be set to _smm_max_size or to sort_area_size explicitly, which is a little surprising. Also, pga_aggregate_target has to be set 5 times the desired workarea size (4GB), which means 20 GB in order to fully exploit the possible workarea. Any further increase of pga_aggregate_target or _pga_max_size do not offer any additional benefit for a single work area.

I have filed an enhancement request in 2014 to have the amount of memory for a single workarea increased and I am sure that I was not the only one. However, even in release 12.2.0.1, the workarea size is still strongly limited. It remains to be seen with which future version, this limit can be avoided.

References:

- If your memory serves you right
  http://www.dbguide.net/upload/20060317/1142597040621.pdf

- Secrets of the Oracle Database - Norbert Debes, Apress

- Alex Fatkulin - Hotsos 2014 - Leveraging in-memory storage to overcome Oracle Database PGA memory limits - https://de.slideshare.net/Enkitec/fatkulin-hotsos-2014

- MOS 453540.1 - How To Super-Size Work Area Memory Size Used By Sessions? (Doc ID 453540.1)

- Bug 24969248 : INCREASE SORT AREA PER PROCESS LIMIT IN AUTOMATIC PGA MANAGEMENT

- Bug 19206047 : INTERNAL LIMITATION THAT A SINGLE WORKAREA SIZE IS LIMITED TO 2GB

Todos:

Research _smm_iosort_cap