**[一份11gR2 rac awr报告的简单分析](https://www.cnblogs.com/andy6/p/5766785.html)**

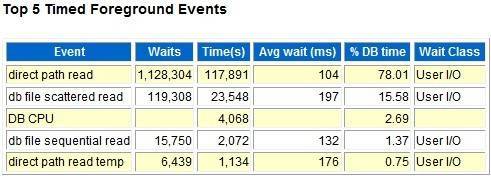
昨晚网友发来一份awr报告，希望帮忙分析一下。由于其他信息都没有，仅仅只有一份awr，鉴于目前有大多的朋友还不太熟悉或者说不知道如何去进行awr的分析。我这里就拿这个awr来进行分析，当抛砖引玉了。首先申请，网上分析awr的文章不少，大家也都可以参考一下。

首先来看awr前面部分信息，了解下系统的版本，以及大概判断下系统负载如何。  
[](http://www.killdb.com/wp-content/uploads/2012/09/111.jpg)

从上面信息我们可以得出如下结论：  
—-数据库是一套11.2.0.2的RAC  
—-Solaris 64bit环境，48c  
—-每个cpu 的可用处理时间是3591.6 s, 2518.79/(59.86\*48)=0.876，说明系统负载还是有些高的，但是连接数却不多。

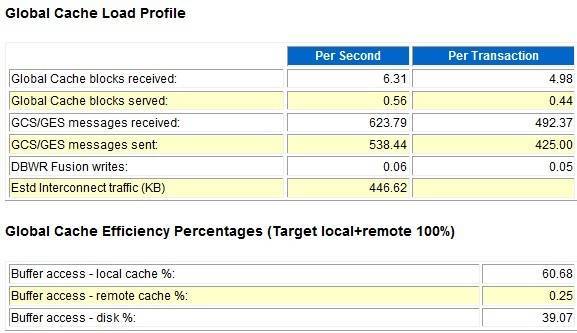
从下面的load profile来看，逻辑读、物理读都不算太高、软解析也很低。  
[](http://www.killdb.com/wp-content/uploads/2012/08/2.jpg)

下面重点来看下top 5 event：

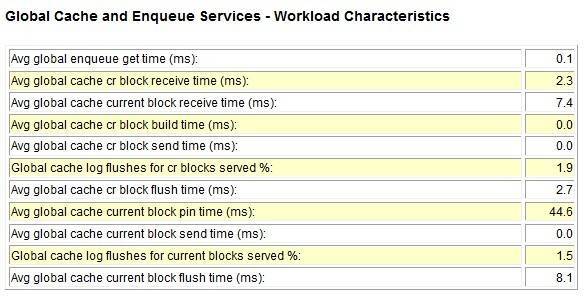
[](http://www.killdb.com/wp-content/uploads/2012/08/3.jpg)  
从top 5 等待来看，绝大多数event表现为IO类的等待，而且平均等待时间都比较高，均超过100ms，最高的为197ms。从这里，我们可以粗略的判断，系统IO问题较为严重，结合top 5来看，主要集中在direct path read等待，关于direct path read，主要表现为如下集中可能的情况：  
—–parallel query  
—-大量的disk排序  
—-table 预读取操作

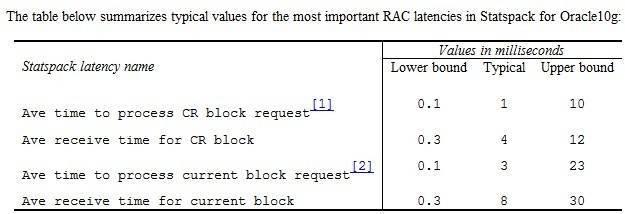
这个等待跟direct path write是不同的，direct path write主要表现为直接路径加载，parallel dml等操作。其次top 5 里的direct path read temp 主要表现为大量的disk temp 排序操作。

这里补充一点的是，在11gR2里面，部分全表扫描操作会被direct path read(简称DPR)所替代，当table大小超过5倍\_small\_table\_threshold 大小时，oracle将自动决定是否使用direct path read去替代传统的全表扫描。另外其实还有一个参数\_very\_large\_object\_threshold，当表的大小超过这个参数的0.8倍时，oracle也会自动决定使用使用DPR去代替传统的全表扫描。下面是我们11gR2版本的关于这2个参数设置和描述：  
\_small\_table\_threshold —单位是block,11.2中默认是287.  
\_very\_large\_object\_threshold —单位是MB，默认是500m

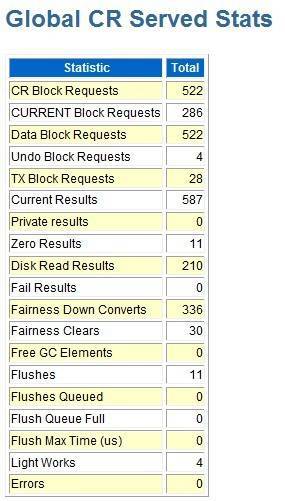
下面我们继续，由于这是一套rac，所以我们也关注下rac相关的信息，如下：  
[](http://www.killdb.com/wp-content/uploads/2012/08/4.jpg)

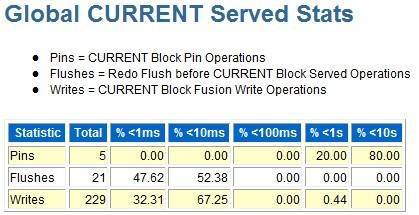
从上面来看，可以看到rac interconnect 交互比较低的，从buffer access信息来看，从disk中的读取比例有些高了，为39.07%，这也是前面direct path read/read temp的一个表现。

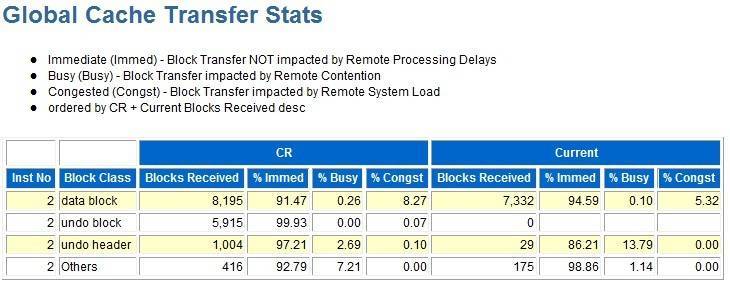
下面是ges/gcs的具体指标。  
[](http://www.killdb.com/wp-content/uploads/2012/08/5.jpg)

这里，我们重点关注receive和send的部分，通常来讲小于5ms认为是比较好的，小于20ms都算是可以接受的，这些信息其实没有一个明确的指标，都是根据经验来的，白鳝的dba日记中曾经提到过一些具体的指标，不过其中是参考的9i的rac文档。我从一份oracle internal的rac文档中找到了如下一些指标：  
[](http://www.killdb.com/wp-content/uploads/2012/08/6.jpg)

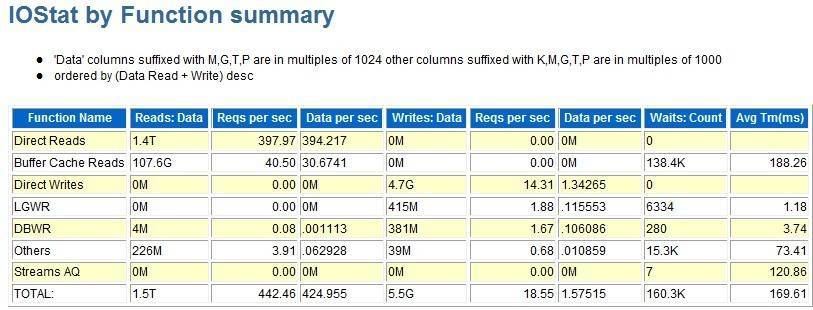
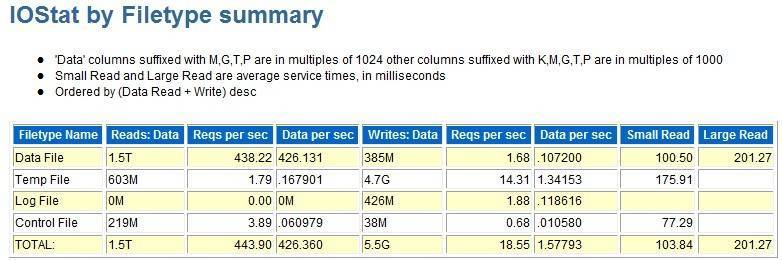
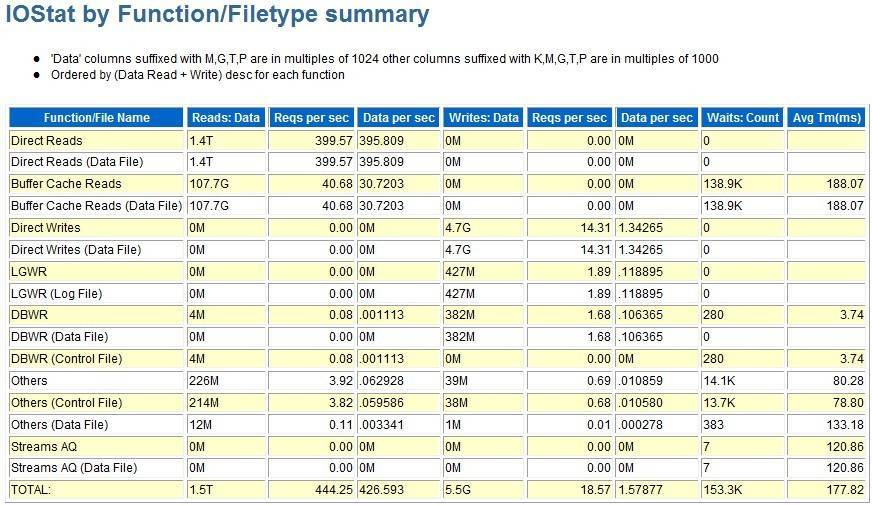
从这里来看，对于CR block小于5ms 认为是不错的，上限是10ms左右。而current block  
一般在3ms和8ms被认为不错，当超过20ms则被视为异常或不可能接受了。反过来讲，如果这里指比较高，那么top 5 event里面比如会有gc相关的等待。

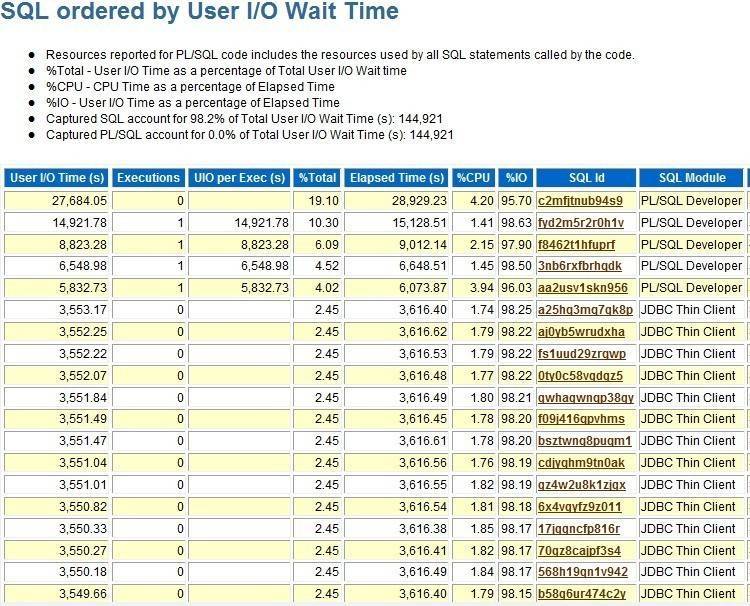
如果这里看到cr或current block 接受或send有异常，你可以看awr中rac相关的具体细节部分，例如如下部分：  
[](http://www.killdb.com/wp-content/uploads/2012/08/7.jpg)

[](http://www.killdb.com/wp-content/uploads/2012/08/8.jpg)

[](http://www.killdb.com/wp-content/uploads/2012/08/9.jpg)

由于这里没什么问题，所以我就不多说了，继续分析这个awr。

由于前面我们已经清楚的看到，问题集中在IO上，那么我们就来看看IO方面的细节：  
[](http://www.killdb.com/wp-content/uploads/2012/08/10.jpg)  
[](http://www.killdb.com/wp-content/uploads/2012/08/11.jpg)  
[](http://www.killdb.com/wp-content/uploads/2012/08/12.jpg)

可以看到1小时内，所读的block达到了1.5T，太大了。其中绝大部分是集中在对datafile的访问，tmepfile的操作有一些，相对来讲算是比较少的了。  
这样来看，问题集中在应用层面了，我们来看看SQL部分。  
[](http://www.killdb.com/wp-content/uploads/2012/09/112.jpg)  
可以看到，主要是就那几个parallel query的SQL导致的大量IO，从sql 部分信息，我没有看到Top SQL with event的信息，所以从这来看，前面单从top 5event中的direct path read怀疑可能是11gR2的特性 当表超过5倍\_small\_table\_threshold 大小而是用direct path read来代替全表扫描是不正确的。从这里来看，就是那几个parallel query导致的direct path read，而且direct path read temp应该也是这个导致，因为sql包含大量的排序了。

刚刚网友说，他们这个库目前是每天下午6~7点都会down机，目前已经调整了应用了。我想，问题应该已经解决了，拭目以待了。

最后补充下IO类相关几个event的一些指标，比如平均等待时间超过多少ms，我们就认为有问题了。  
可以参考如下表格：  
[](http://www.killdb.com/wp-content/uploads/2012/08/14.jpg)

说明：awr报告中还有很多其他的内容，就不说明了，例如：  
[](http://www.killdb.com/wp-content/uploads/2012/08/15.jpg)

例如instance activity statistics，我们可以通过里面的数据去判断数据中是否有表存在行链接或行迁移，再或者看系统回滚率是否正常、一些参数设置是否合理等等。buffer pool statistics和advisory statistics部分信息是从来判断oracle sga和pga中内存组件设置是否合理的，当然这个仅仅是参考值，而且最好是采集多份awr包括进行综合分析。后面的undo/latch/dictionary等信息就不多说了，回头有具体的案例涉及到在进行分析描述。这次就先到这里吧。