MySQL 服务器内存使用

Every so often people ask me the question how should they estimate memory consumption by MySQL Server in given configuration. What is the formula they could use.

经常有人问我配置MySQL时该如何估算内存的消耗。那么该使用什么公式来计算呢？

The reasons to worry about memory usage are quite understandable. If you configure MySQL Server so it uses too small amount of memory it will likey perform suboptimally. If you however configure it so it consumes too much memory it may be crashing , failing to execute queries or make operation to swap seriously slowing down. On now legacy 32bit platforms you could also run out of address space so that had to be watched as well.  
Having said so, I do not think looking for the secret fomula to compute your possible memory usage is the right approach to this problem. The reasons are - this formula is very complex nowadays and what is even more important “theoretically possible” maximum it provides have nothing to do with real memory consumptions. In fact typical server with 8GB of memory will often run with maximum theoretical memory usage of 100GB or more. Furthermore there is no easy “overcommit factor” you can use - it really depends on application and configuration. Some applications will drive server to 10% of theoretical memory consumptions others only to 1%.

关心内存怎么使用的原因是可以理解的。如果配置MySQL服务器使用太少的内存会导致性能不是最优的；如果配置了太多的内存则会导致崩溃，无法执行查询或者导致交换操作严重变慢。在现在的32位平台下，仍有可能把所有的地址空间都用完了，因此需要监视着。  
话虽如此，但我并不觉得找到什么可以计算内存使用的秘诀公式就能很好地解决这个问题。原因有 -- 如今这个公式已经很复杂了，更重要的是，通过它计算得到的值只是“理论可能”并不是真正消耗的值。事实上，有8GB内存的常规服务器经常能运行到最大的理论值 -- 100GB甚至更高。此外，你轻易不会使用到“超额因素” -- 它实际上依赖于应用以及配置。一些应用可能需要理论内存的 10% 而有些仅需 1%。

So what could you do instead ? First take a look at global buffers which are allocated at start and always where - these are **key\_buffer\_size, innodb\_buffer\_pool\_size, innodb\_additional\_memory\_pool\_size, innodb\_log\_buffer\_size, query\_cache\_size**. If you’re using MyISAM seriously you can also add the size of Operation System cache you would like MySQL to use for your table. Take this number add to it number of memory Operation System and other applications need, add might be 32MB more for MySQL Server code and various small static buffers. This is memory which you can consider used when you just start MySQL Server. The rest of memory is available for connections. For exampe with 8GB server you might have everything listed adding up to 6GB, so you have 2GB left for your threads.

那么，我们可以做什么呢？首先，来看看那些在启动时就需要分配并且总是存在的全局缓冲-- **key\_buffer\_size, innodb\_buffer\_pool\_size, innodb\_additional\_memory\_pool\_size, innodb\_log\_buffer\_size, query\_cache\_size**。如果你大量地使用MyISAM表，那么你也可以增加操作系统的缓存空间使得MySQL也能用得着。把这些也都加到操作系统和应用程序所需的内存值之中，可能需要增加32MB甚至更多的内存给MySQL服务器代码以及各种不同的小静态缓冲。这些就是你需要考虑的在MySQL服务器启动时所需的内存。其他剩下的内存用于连接。例如有8GB内存的服务器，可能监听所有的服务就用了6GB的内存，剩下的2GB内存则留下来给线程使用。

Each thread connecting to MySQL server will needs its own buffers. About 256K is allocated at once even if thread is idle - they are used by default thread stack, net buffer etc. If transaction is started some more space can add up. Running small queries might only barely increase memory consumption for given thread, however if table will perform complex operations such as full table scans, sorts, or need temporary tables as much as **read\_buffer\_size, sort\_buffer\_size, read\_rnd\_buffer\_size, tmp\_table\_size**of memory might be allocated. But they are only allocated upon the need and freed once given stage of query is done. Some of them are allocated as single chunk at once others, for example **tmp\_table\_size** is rather maximum amount of memory MySQL will allocate for this operation. Note it is more complicated than once may think - multiple buffers of the same type might be allocated for exampe to handle subqueries. For some special queries memory usage might be even larger - bulk inserts may allocate **bulk\_insert\_buffer\_size**bytes of memory if done to MyISAM tables. **myisam\_sort\_buffer\_size** used for *ALTER TABLE, OPTIMIZE TABLE, REPAIR TABLE* commands.

每个连接到MySQL服务器的线程都需要有自己的缓冲。大概需要立刻分配256K，甚至在线程空闲时 -- 它们使用默认的线程堆栈，网络缓存等。事务开始之后，则需要增加更多的空间。运行较小的查询可能仅给指定的线程增加少量的内存消耗，然而如果对数据表做复杂的操作例如扫描、排序或者需要临时表，则需分配大约 **read\_buffer\_size, sort\_buffer\_size, read\_rnd\_buffer\_size, tmp\_table\_size**大小的内存空间。不过它们只是在需要的时候才分配，并且在那些操作做完之后就释放了。有的是立刻分配成单独的组块，例如 **tmp\_table\_size** 可能高达MySQL所能分配给这个操作的最大内存空间了。注意，这里需要考虑的不只有一点 -- 可能会分配多个同一种类型的缓存，例如用来处理子查询。一些特殊的查询的内存使用量可能更大 -- 如果在MyISAM表上做成批的插入时需要分配 **bulk\_insert\_buffer\_size** 大小的内存。执行 *ALTER TABLE, OPTIMIZE TABLE, REPAIR TABLE* 命令时需要分配 **myisam\_sort\_buffer\_size** 大小的内存。

For OLTP applications with simple queries memory consumption is often less than 1MB per thread with default buffers, and you really do not need to increase per thread buffers unless you have complex queries. Sorting 10 rows will be as fast with 1MB sort buffer as with 16MB (actually 16MB might be even slower but it is other story).

只有简单查询OLTP应用的内存消耗经常是使用默认缓冲的每个线程小于1MB，除非需要使用复杂的查询否则无需增加每个线程的缓冲大小。使用1MB的缓冲来对10行记录进行排序和用16MB的缓冲基本是一样快的（实际上16MB可能会更慢，不过这是其他方面的事了）。

Another approach you may take is to come up with amount of memory you want MySQL Server to consume at peak. This can be easily computed by memory needed for OS, File Cache and other applications. For 32bit envinronment you also should keep 32bit limits into account and probably limit “mysqld” size to about 2.5GB (exact number depens on a lot of factors). Now you can use “ps aux” to see VSZ - Virtual Memory allocated by MySQL process. You can also look at “Resident Memory” but I find it less helpful as it may down because of swapping - not what you would like to see. Monitor how the value changes so you know memory requirements with current settings and increase/decrease values appropriately.

另外，就是找出MySQL服务器内存消耗的峰值。这很容易就能计算出操作系统所需的内存、文件缓存以及其他应用。在32位环境下，还需要考虑到32位的限制，限制 “mysqld” 的值大约为2.5G（实际上还要考虑到很多其他因素）。现在运行 “ps aux” 命令来查看 VSZ 的值 -- MySQL 进程分配的虚拟内存。也可以查看 “Resident Memory” 的值，不过我想它可能没多大用处，因为它会由于交换而变小 -- 这并不是你想看到的。监视着内存变化的值，就能知道是需要增加/减少当前的内存值了。

Some may say, Hey we want to have 100% guarantee our server will never run out of memory, no matter which queries or users will decide to run. Unfortunately this is as much close to impossible to be impractical. Here is why:

可能有的人想说，我们想要让服务器能保证100%不会耗尽内存，不管决定用什么样的查询、什么样的用户。很不幸，这其实很不明智也不可能，因为：

**List of rarely considered MySQL Server Memory Requirements**

**以下是很少考虑的MySQL服务器内存需求**

* **Thread buffers can be allocated more than once for each thread**. Consider for example subqueries - each layer may need its own read\_buffer,sort\_buffer, tmp\_table\_size etc
* **每个线程可能会不止一次需要分配缓冲。** 考虑到例如子查询 -- 每层都需要有自己的 read\_buffer,sort\_buffer, tmp\_table\_size 等。
* **Many variabes can be set per connection**. So you can’t relay on global values if developers may use their local values to run some queries.
* **在每个连接中很多变量都可能需要重新设置。** 如果开发者想设定自己的变量值来运行某些查询就不能继续使用全局值。
* **There can be mutiple key caches.** Multiple key caches can be created to accomodate query executions
* **可能有多个索引缓存。** 为了配合执行查询可能会创建多个索引缓存。
* **Query Parsing and optimization needs memory.**This is usually small to be ignored but certain queries can have very large memory requrement for this step, especially specially crafted ones.
* **解析查询和优化都需要内存。** 这些内存通常比较小，可以忽略，不过如果是某些查询在这个步骤中则需要大量内存，尤其是那些设计的比较特别的查询。
* **Stored Procedures.**Compex stored procedures may require a lot of memory
* **存储过程。** 复杂的存储过程可能会需要大量内存。
* **Prepared statements and Cursors.** Single connection may have many prepared statements and cursors. Their number finally can be limited but each of them still can have very large memory consumption
* **准备查询语句以及游标。** 单次链接可能会有很多的准备好的语句以及游标。它们的数量最后可以限定，但是仍然会消耗大量的内存。
* **Innodb Table Cache.** Innodb has its own table cache in which meta data about each table accessed from the start is stored. It is never purged and may be large if you have a lot of tables. It also means user having *CREATE TABLE*privilege should be able to run MySQL server out of memory
* **Innodb表缓存。** Innnodb表有自己的缓存，它保存了从一开始访问每个表的元数据。它们从未被清除过，如果有很多Innodb表的话，那么这个量就很大了。这也就意味着拥有 *CREATE TABLE* 权限的用户就可能把MySQL服务器的内存耗尽。
* **MyISAM buffers.**MyISAM may allocate buffer which is large enough to contain largest record in the given table which is held until table is closed.
* **MyISAM缓冲。** MyISAM表可能会分配一个足以装下指定表最大记录的缓冲，而且这个缓冲直到表关闭了才释放。
* **Federated Storage Engine.**This may have unbound memory requirements retriving result sets from remove queries.
* **FEDERATED存储引擎。** This may have unbound memory requirements retriving result sets from remove queries.
* **Blobs may require 3x time of memory.**This is important if you’re deaing with large Blobs (your **max\_allowed\_packet**is large) Processing of 256MB of blob may require 768MB of memory.
* **Blobs可能需要3倍的内存。** 这在处理很大（**max\_allowed\_packet** 的值较大）的Blobs数据时很重要，如果处理256MB的数据可能需要768MB的内存。
* **Storage Engines.**In general storage engines may have their own per thread or global memory allocations which are not tuned as buffers. Watch for these especially now with many storage engines being released for MySQL by various parties.
* **存储引擎。** 通常情况下，存储引擎会设置自己的每个线程的全局分配内存，它通常不能像缓存一样可以调节。现在应该通过各种方式来特别关注MySQL释放出来的存储引擎。

I do not pretend this to be complete list. On the contrary I’m quite sure I’ve missed something (drop me a note if you have something to add). But the main point is - there are a lot of memory consumers out where and trying to find peak possible usage for each is impractical - so my advice would be measure what you get in practice and how memory consumption reacts to changing various variables. For example you may find out increasing **sort\_buffer\_size** from 1MB to 4MB and 1000 **max\_connections** increases peak memory consumption just 30MB not 3000MB as you might have counted.

我想这还不是完成的列表，相反地，我觉得还是漏掉了一些（如果你知道，请给我回复加上）。但主要的原因是 -- 找到每次内存消耗峰值是不切实际的，因此我的这些建议可以用来衡量一下你实际修改一些变量值产生的反应。例如，把 **sort\_buffer\_size** 从1MB增加到4MB并且在 **max\_connections** 为 1000 的情况下，内存消耗增长峰值并不是你所计算的3000MB而是30MB。