

图文 22 生产经验：如何通过chunk来支持数据库运行期间的Buffer Pool动态调整？

手机观看

539 人次阅读 2020-02-14 07:56:09

详情 评论

生产经验：如何通过chunk来支持数据库运行期间的Buffer Pool动态调整？

如何提问：每篇文章都有评论区，大家可以尽情留言提问，我会逐一答疑

如何加群：购买狸猫技术窝专栏的小伙伴都可以加入狸猫技术交流群，一个非常纯粹的技术交流的地方

具体加群方式，请参见目录菜单下的文档：《MySQL专栏付费用户如何加群》（购买后可见）

1、buffer pool这种大块头，能在运行期间动态调整大小吗？

上一篇文章给大家分析了一下buffer pool在多线程并发访问的时候的一些问题，以及通过多个buffer pool是如何优化多线程并发访问性能的。

那么这一篇文章我们接着分析下一个问题，那就是buffer pool这种大块头数据结构，在数据库运行期间，可以动态的调整他的大小吗？

其实如果就我们讲的这套原理的话，buffer pool在运行期间是不能动态的调整自己的大小的

为什么呢？因为动态调整buffer pool大小，比如buffer pool本来是8G，运行期间你给调整为16G了，此时是怎么实现的呢？

就是需要这个时候向操作系统申请一块新的16GB的连续内存，然后把现在的buffer pool中的所有缓存页、描述数据块、各种链表，都拷贝到新的16GB的内存中去。这个过程是极为耗时的，性能很低下，是不可以接受的！

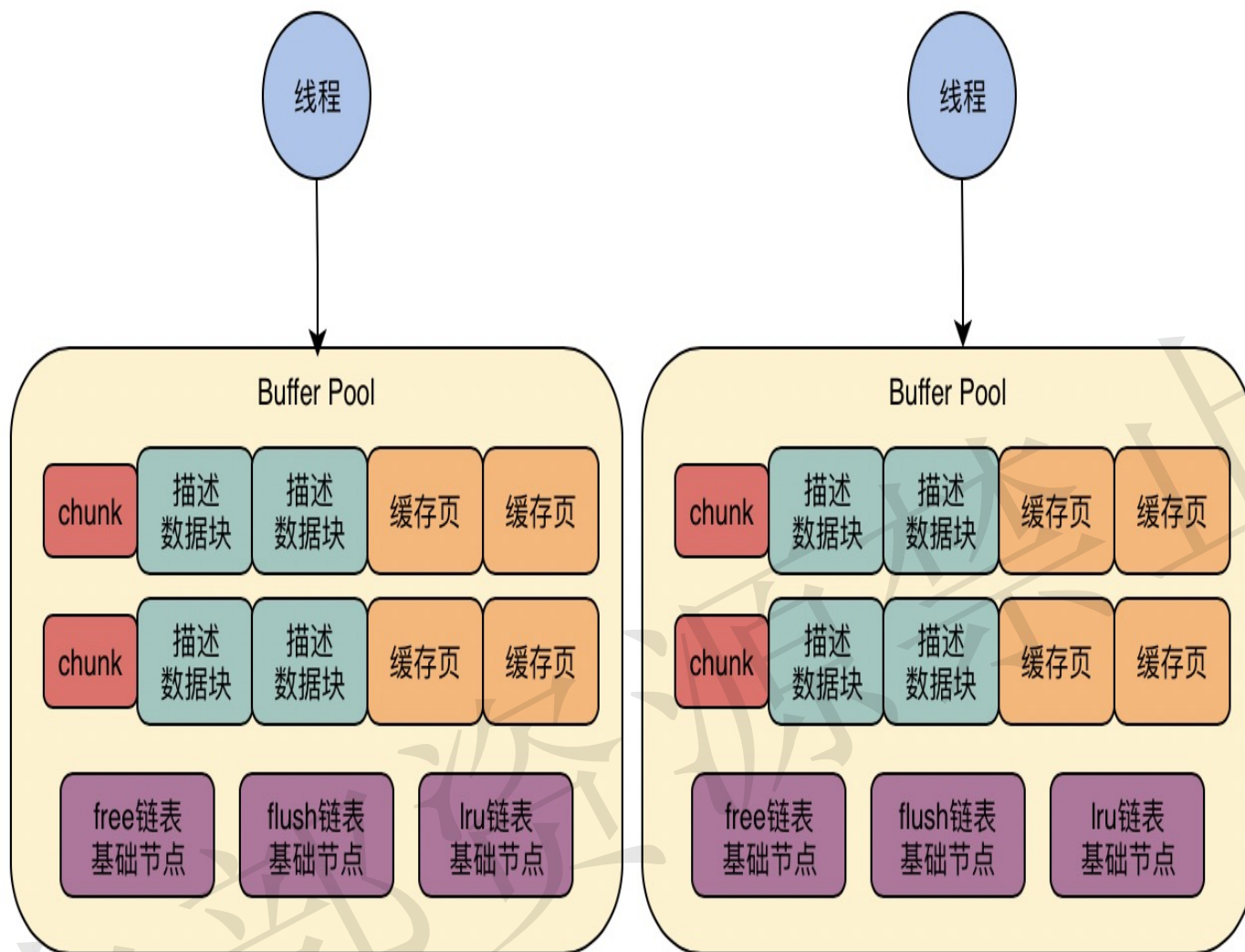
所以就目前讲解的这套原理，buffer pool是绝对不能支持运行期间动态调整大小的。

2、如何基于chunk机制把buffer pool给拆小呢？

但是MySQL自然会想办法去做一些优化的，他实际上设计了一个chunk机制，也就是说buffer pool是由很多chunk组成的，他的大小是innodb_buffer_pool_chunk_size参数控制的，默认值就是128MB。

所以实际上我们可以来做一个假设，比如现在我们给buffer pool设置一个总大小是8GB，然后有4个buffer pool，那么每个buffer pool就是2GB，此时每个buffer pool是由一系列的128MB的chunk组成的，也就是说每个buffer pool会有16个chunk。

然后每个buffer pool里的每个chunk里就是一系列的描述数据块和缓存页，每个buffer pool里的多个chunk共享一套free、flush、lru这些链表，此时的话，看起来可能大致如下图所示。



在上面的图里，可以清晰的看到，每个buffer pool里已经有了多个chunk，每个chunk就是一系列的描述数据块和缓存页，这样的话，就是把buffer pool按照chunk为单位，拆分为了一系列的小数据块，但是每个buffer pool是共用一套free、flush、lru的链表的。

3、基于chunk机制是如何支持运行期间，动态调整buffer pool大小的？

那么现在有了上面讲的这套chunk机制，就可以支持动态调整buffer pool大小了。

比如我们buffer pool现在总大小是8GB，现在要动态加到16GB，那么此时只要申请一系列的128MB大小的chunk就可以了，只要每个chunk是连续的128MB内存就行了。然后把这些申请到的chunk内存分配给buffer pool就行了。

有个这个chunk机制，此时并不需要额外申请16GB的连续内存空间，然后还要把已有的数据进行拷贝。

给大家讲解这个chunk机制，倒不是让大家在数据库运行的时候动态调整buffer pool大小，其实这不是重点，重点是大家要了解数据库的buffer pool的真实的数据结构，是可以由多个buffer pool组成的，每个buffer pool是多个chunk组成的，然后你只要知道他运行期间可以支持动态调整大小就可以了。

4、昨日思考题解答

现在我们来解答一下昨天的思考题，昨天让大家思考了一下，到底如何避免你执行crud的时候，频繁地发现缓存页都用完了，完了还得先把一个缓存页刷入磁盘腾出一个空闲缓存页，然后才能从磁盘读取一个自己需要的数据页到缓存页里来。

如果频繁这么搞，那么很多crud操作，每次都要执行两次磁盘IO，一次是缓存页刷入磁盘，一次是数据页从磁盘里读取出来，性能是很不高的。

其实结合我们了解到的buffer pool的运行原理就可以知道，如果要避免上述问题，说白了就是避免缓存页频繁的被使用完毕。那么我们知道实际上你在使用缓存页的过程中，有一个后台线程会定时把LRU链表冷数据区域的一些缓存页刷入磁盘中。

所以本质上缓存页一边会被你使用，一边会被后台线程定时的释放掉一批。

所以如果你的缓存页使用的很快，然后后台线程释放缓存页的速度很慢，那么必然导致你频繁发现缓存页被使用完了。但是缓存页被使用的速度你是没法控制的，因为那是由你的Java系统访问数据库的并发程度来决定的，你高并发访问数据库，缓存页必然使用的很快了！

然后你后台线程定时释放一批缓存页，这个过程也很难去优化，因为你要是释放的过于频繁了，那么后台线程执行磁盘IO过于频繁，也会影响数据库的性能。

所以这里的关键点就在于，你的buffer pool有多大！

如果你的数据库要抗高并发的访问，那么你的机器必然要配置很大的内存空间，起码是32GB以上的，甚至64GB或者128GB。此时你就可以给你的buffer pool设置很大的内存空间，比如20GB，48GB，甚至80GB。

这样的话，你会发现高并发场景下，数据库的buffer pool缓存页频繁的被使用，但是你后台线程也在定时释放一些缓存页，那么综合下来，空闲的缓存页还是会以一定的速率逐步逐步的减少。

因为你的buffer pool内存很大，所以空闲缓存页是很多很多的，即使你的空闲缓存页逐步的减少，也可能需要较长时间才会发现缓存页用完了，此时才会出现一次crud操作执行的时候，先刷缓存页到磁盘，再读取数据页到缓存页来，这种情况是不会出现的太频繁的！

而一旦你的数据库高峰过去，此时缓存页被使用的速率下降了很多很多，然后后台线程会定是基于flush链表和lru链表不停的释放缓存页，那么你的空闲缓存页的数量又会在数据库低峰的时候慢慢的增加了。

所以线上的MySQL在生产环境中，buffer pool的大小、buffer pool的数量，这都是要用心设置和优化的，因为多MySQL的性能和并发能力，都会有较大的影响。

5、实践思考题

请每位同学，去看看自己负责的系统的buffer pool大小、buffer pool数量、chunk大小，然后看看自己的数据库的机器配置，思考一下，当前设置是否合理？为什么要这样设置？

大家可以把自己的思考发在评论区一起交流。

End

狸猫技术窝精品专栏及课程推荐：

[《从零开始带你成为消息中间件实战高手》](#)

[《21天互联网Java进阶面试训练营》（分布式篇）](#)

[《互联网Java工程师面试突击》（第1季）](#)

[《互联网Java工程师面试突击》（第3季）](#)

[《从零开始带你成为JVM实战高手》](#)

