

图文 16 简单的LRU链表在Buffer Pool实际运行中，可能导致哪些问题？

561 人次阅读 2020-02-11 09:03:06

[详情](#) [评论](#)

简单的LRU链表在Buffer Pool实际运行中，可能导致哪些问题？

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）



狸猫技术窝

进店逛

1、简单回顾一下

之前我们讲解了Buffer Pool在使用过程中如果缓存页都使用了，没有空闲的缓存页时，可以去LRU链表中的尾部找一个最近最少使用的缓存页，把他的数据刷入磁盘，腾出来一个空闲缓存页，然后加载需要的新的磁盘数据页到空闲缓存页里去。

而LRU链表的机制也很简单，只要是刚从磁盘上加载数据到缓存页里去，这个缓存页就放入LRU链表的头部，后续如果对任何一个缓存页访问了，也把缓存页从LRU链表中移动到头部去。

这样在LRU链表的尾部，一定是最近最少被访问的那个缓存页。

2、预读带来的一个巨大问题

但是这样的一个LRU机制在实际运行过程中，是存在巨大的隐患的。

首先会带来隐患的就是MySQL的预读机制，这个所谓预读机制，说的就是当你从磁盘上加载一个数据页的时候，他可能会连带着把这个数据页相邻的其他数据页，也加载到缓存里去！

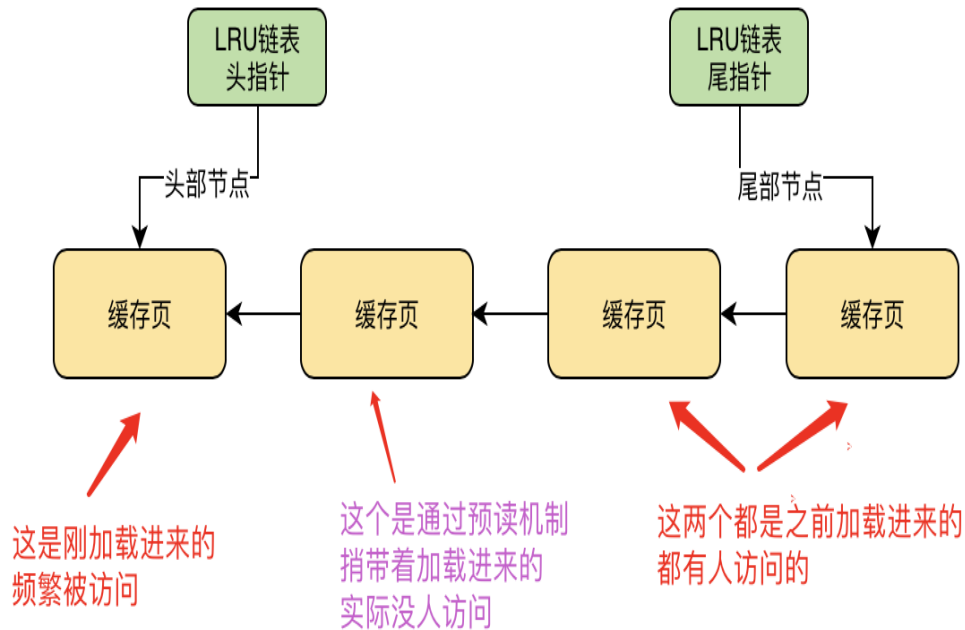
举个例子，假设现在有两个空闲缓存页，然后在加载一个数据页的时候，连带着把他的一个相邻的数据页也加载到缓存里去了，正好每个数据页放入一个空闲缓存页！

但是接下来呢，实际上只有一个缓存页是被访问了，另外一个通过预读机制加载的缓存页，其实并没有人访问，此时这两个缓存页可都在LRU链表的前面，如下图。

相关频道



从零开始
实战优化
已更新3



我们可以看到，这个图里很清晰的表明了，前两个缓存页都是刚加载进来的，但是此时第二个缓存页是通过预读机制捎带着加载进来的，他也放到了链表的前面，但是他实际没人访问他。

除了第二个缓存页之外，第一个缓存页，以及尾巴上两个缓存页，都是一直有人访问的那种缓存页，只不过上图代表的是刚刚把头部两个缓存页加载进来的时候的一个LRU链表当时的情况。

这个时候，假如没有空闲缓存页了，那么此时要加载新的数据页了，是不是就要从LRU链表的尾部把所谓的“最近最少使用的一个缓存页”给拿出来，刷入磁盘，然后腾出来一个空闲缓存页了？

这个时候，如果你把上图中LRU尾部的那个缓存页刷入磁盘然后清空，你觉得合理吗？他可是之前一直频繁被人访问的啊！只不过在这一个瞬间，被新加载进来的两个缓存页给占据了LRU链表前面的位置，尤其是第二个缓存页，居然还是通过预读机制加载进来的，根本就不会有人访问！

那么这个时候，你要是把LRU链表尾部的缓存页给刷入磁盘，这是绝对不合理的，最合理的反而是把上图中LRU链表的第二个通过预读机制加载进来的缓存页给刷入磁盘和清空，毕竟他几乎是没什么人会访问的！

3、哪些情况下会触发MySQL的预读机制？

现在我们已经理解了预读机制一下子把相邻的数据页加载进缓存，放入LRU链表前面的隐患了，预读机制加载进来的缓存页可能根本不会有人访问，结果他却放在了LRU链表的前面，此时可能会把LRU尾部的那些被频繁访问的缓存页刷入磁盘！

所以我们来看看，到底哪些情况下会触发MySQL的预读机制呢？

(1) 有一个参数是`innodb_read_ahead_threshold`，他的默认值是56，意思就是如果顺序的访问了一个区里的多个数据页，访问的数据页的数量超过了这个阈值，此时就会触发预读机制，把下一个相邻区中的所有数据页都加载到缓存里去

(2) 如果Buffer Pool里缓存了一个区里的13个连续的数据页，而且这些数据页都是比较频繁会被访问的，此时就会直接触发预读机制，把这个区里的其他的数据页都加载到缓存里去

这个机制是通过参数`innodb_random_read_ahead`来控制的，他默认是OFF，也就是这个规则是关闭的

所以默认情况下，主要是第一个规则可能会触发预读机制，一下子把很多相邻区里的数据页加载到缓存里去，这些缓存页如果一下子都放在LRU链表的前面，而且他们其实并没什么人会访问的话，那就会如上图，导致本来就在缓存里的一些频繁被访问的缓存页在LRU链表的尾部。

这样的话，一旦要把一些缓存页淘汰掉，刷入磁盘，腾出来空闲缓存页，就会如上所述，把LRU链表尾部一些频繁被访问的缓存页给刷入磁盘和清空掉了！这是完全不合理的，并不应该这样！

4、另外一种可能导致频繁被访问的缓存页被淘汰的场景

接着我们讲另外一种可能导致频繁被访问的缓存页被淘汰的场景，那就是**全表扫描**

这个所谓的全表扫描，意思就是类似如下的SQL语句：SELECT * FROM USERS

此时他没加任何一个where条件，会导致他直接一下子把这个表里所有的数据页，都从磁盘加载到Buffer Pool里去。

这个时候他可能会一下子就把这个表的所有数据页都——装入各个缓存页里去！此时可能LRU链表中排在前面的一大串缓存页，都是全表扫描加载进来的缓存页！那么如果这次全表扫描过后，后续几乎没用到这个表里的数据呢？

此时LRU链表的尾部，可能全部都是之前一直被频繁访问的那些缓存页！

然后当你淘汰掉一些缓存页腾出空间的时候，就会把LRU链表尾部一直被频繁访问的缓存页给淘汰掉了，而留下了之前全表扫描加载进来的大量的不经常访问的缓存页！

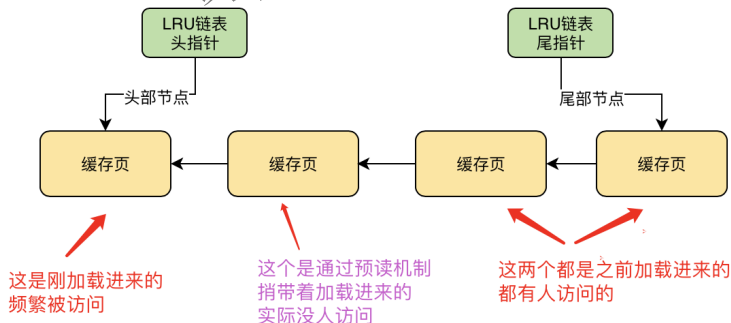
5、总结

所以我们对今天讲到的内容做一点小小的总结，如果你使用简单的LRU链表的机制，其实是漏洞百出的，因为很可能预读机制，或者全表扫描的机制，都会一下子把大量未来可能不怎么访问的数据页加载到缓存页里去，然后LRU链表的前面全部是这些未来可能不怎么会被访问的缓存页！

而真正之前一直频繁被访问的缓存页可能此时都在LRU链表的尾部了！

如果此时此刻，需要把一些缓存页刷入磁盘，腾出空间来加载新的数据页，那么此时就只能把LRU链表尾部那些一直频繁被访问的缓存页给刷入磁盘了！

最后我们再看一下下面的图示，想必大家是很好理解的。



6、今日思考题

今天希望大家思考一下：

为什么MySQL要设计预读这个机制？

他加载一个数据页到缓存里去的时候，为什么要把一些相邻的数据页也加载到缓存里去呢？这么做的意义在哪里？

是为了应对什么样的一个场景？

希望大家积极思考，在评论区给出自己的答案！

End

专栏版权归公众号狸猫技术窝所有

未经许可不得传播，如有侵权将追究法律责任

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）](#)