**[jvm系列 (三) ---锁的优化](http://www.cnblogs.com/-new/p/7427036.html)**

<http://www.cnblogs.com/-new/p/7427036.html>

# [jvm系列 (三) ---锁的优化](http://www.cnblogs.com/-new/p/7427036.html)

# 锁的优化

## 目录

* [jvm系列（一）：jvm内存区域与溢出](http://www.cnblogs.com/-new/p/7244460.html)
* [jvm系列（二）：垃圾收集器与内存分配策略](http://www.cnblogs.com/-new/p/7284368.html)
* [jvm系列（三）：锁的优化](http://www.cnblogs.com/-new/p/7427036.html)
* [我的博客目录](http://www.cnblogs.com/-new/p/7396648.html)

# 锁的四种状态

* 从低到高，只能升级不能降级
* 无锁状态
* 偏向锁状态
* 轻量级锁状态
* 重量级锁状态

# 自旋锁和自适应自旋

## 理解

* 同步互斥的时候会造成线程阻塞，而挂起线程和恢复线程需要转入内核态中完成
* 有时候往往共享数据的锁定状态只会持续很短一段时间，那么如果此时将互斥的线程挂起，等待下一次获得锁再恢复线程，这样的话效率不高
* 那么可以不讲线程立即挂起，而是让他自旋（循环）等待获得锁（这就是自旋锁），那么这样可以避免在一段很短的等待时间挂起和恢复线程的开销
* 也就说，不立刻将线程挂起，稍微自旋等一下，这样的话或许可以减少线程挂起和恢复的开销

## 分析

* 虽然可以避免切换线程的开销，但是因为在自旋，这同样是消耗cpu
* 倘若锁被占用的时间很短，那么这短时间的自旋比起切换线程来说是值得的
* 但是倘若锁被占用的时间很长，那么自旋会消耗cpu资源，却没有做有用的事

# 锁消除

## 理解

* 虚拟机即使编译器在运行的时候，同步代码被检测不存在共享数据竞争，那么锁会被消除。也就是说如果数据不会被其他线程访问到，那么我们没必要加锁

# 锁粗化

## 理解

* 我们加锁的时候会尽量减小锁的粒度，减少竞争
* 而如果我们对同一个对象反复加锁和解锁，会造成性能损耗。
* 所以不如把锁的粒度加大

## 使用

StringBuffer sb=new StringBuffer();

sb.append(s1);

sb.append(s2);

...

sb.sppend(sn);

* 我们知道StringBuffer的append方法是加锁的，而这里的锁就是sb对象，如果我们执行每一个append方法都加锁解锁，显然这样性能是不高的，所以当虚拟机检测到一系列操作都是对同一个对象加锁，他会将同步的范围扩大，比如这里扩大到s1-sn，那么这样的话就只需加一次锁

# MarkWord与线程之间的操作过程

* java对象头中的Mark Word默认存储对象的Hashcode、分代年龄和锁标志位
* 进入同步代码块的时候，当前线程的栈帧会建立一个名为锁记录的空间，用于存储锁对象MarkWord的拷贝

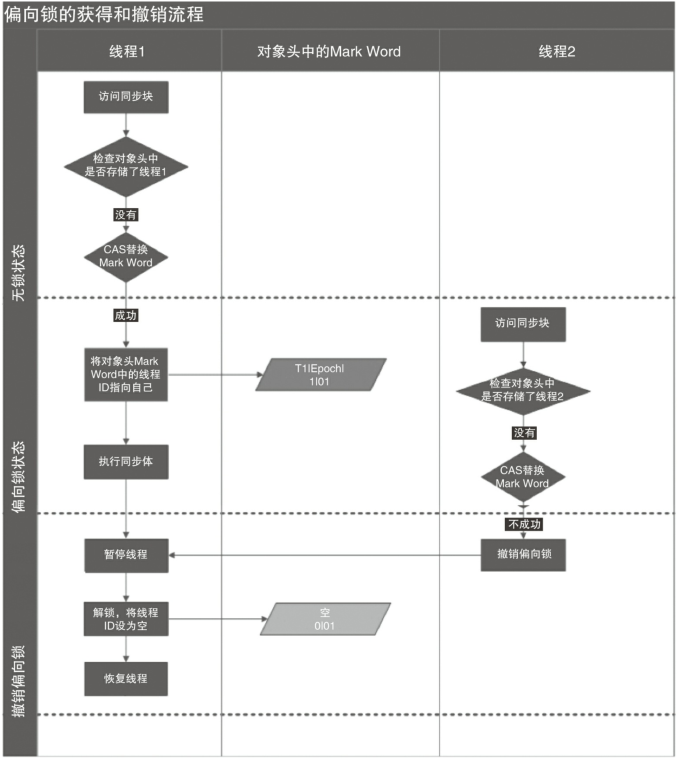
# 偏向锁

## 理解

* 锁会偏向于第一个获得他的线程，如果后面锁没有被其他线程获取，那么持有偏向锁的线程永远不需要再进行同步

## 使用

* 锁对象第一次被线程获取的时候，对象头的标志位会设为偏向模式，同时通过CAS操作把这个线程的ID记录在MarkWord中
* 如果CAS操作成功，持有偏向锁的线程以后每一进入这个锁的同步块时，不需要进行同步操作
* 当另一个线程尝试获取这个锁，偏向锁结束



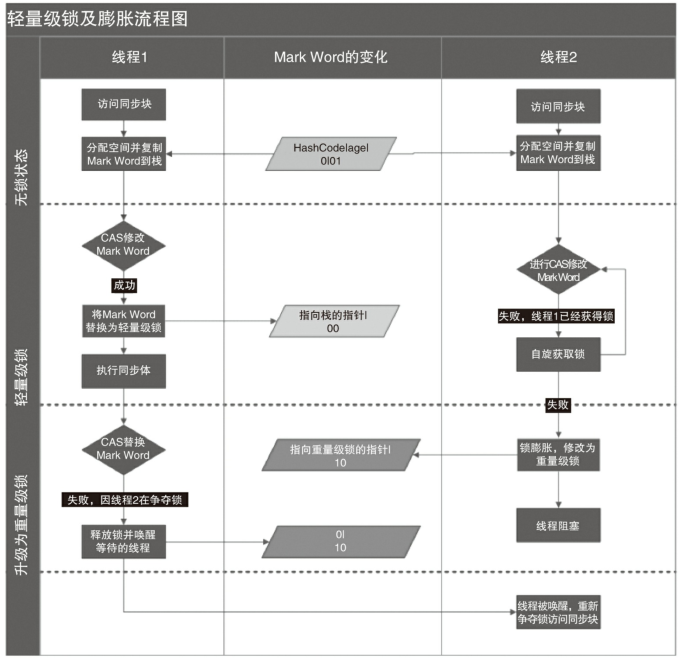
# 轻量级锁

## 理解

* 传统的重量级锁需要使用系统互斥量来实现，而轻量级锁的意图是在没有多线程竞争的情况下，减少重量级锁的系统互斥量产生的性能消耗

## 使用

* 虚拟机使用CAS操作尝试将MarkWord更新为指向LockRecord的指针，如果更新成功表示线程就拥有该对象的锁
* 如果失败，会检查MarkWord是否指向当前线程的栈帧，如果是表示当前线程已经拥有这个锁，如果不是说明这个锁被其他线程抢占，此时膨胀为重量级锁



# 轻量级锁和偏向锁

* 重量级锁需要使用操作系统的互斥量（常常使用一个整型量，0表示解锁，而其他所有的值则表示加锁。通过互斥量使同一资源同时只允许一个访问者对其进行访问）来实现
* 轻量级锁依据大部分的锁在同步周期内不存在竞争，使用CAS操作避免使用互斥量的开销
* 偏向锁的则是在无竞争的情况下，连CAS操作都不做

### 我觉得分享是一种精神，分享是我的乐趣所在，不是说我觉得我讲得一定是对的，我讲得可能很多是不对的，但是我希望我讲的东西是我人生的体验和思考，是给很多人反思，也许给你一秒钟、半秒钟，哪怕说一句话有点道理，引发自己内心的感触，这就是我最大的价值。（这是我喜欢的一句话，也是我写博客的初衷）

作者：jiajun 出处： <http://www.cnblogs.com/-new/>  
本文版权归作者和博客园共有，欢迎转载，但未经作者同意必须保留此段声明，且在文章页面明显位置给出原文连接，否则保留追究法律责任的权利。如果觉得还有帮助的话，可以点一下右下角的【推荐】，希望能够持续的为大家带来好的技术文章！想跟我一起进步么？那就【关注】我吧。

标签: [认识jvm](http://www.cnblogs.com/-new/tag/%E8%AE%A4%E8%AF%86jvm/)

[**好文要顶**](javascript:void(0);) [**关注我**](javascript:void(0);) [**收藏该文**](javascript:void(0);) **[http://common.cnblogs.com/images/icon_weibo_24.png](javascript:void(0);)** **[http://common.cnblogs.com/images/wechat.png](javascript:void(0);)**

[http://pic.cnblogs.com/face/900751/20161023114041.png](http://home.cnblogs.com/u/-new/)

[jiajun\_geek](http://home.cnblogs.com/u/-new/)  
[关注 - 0](http://home.cnblogs.com/u/-new/followees)  
[粉丝 - 224](http://home.cnblogs.com/u/-new/followers)

[+加关注](javascript:void(0);)

[关注我～](javascript:void(0);)

**3**

[«](http://www.cnblogs.com/-new/p/7417915.html)上一篇：[java基础解析系列(一)---String、StringBuffer、StringBuilder](http://www.cnblogs.com/-new/p/7417915.html)  
[»](http://www.cnblogs.com/-new/p/7469763.html)下一篇：[java基础解析系列(二)---Integer](http://www.cnblogs.com/-new/p/7469763.html)

posted @ 2017-08-25 11:35 [jiajun\_geek](http://www.cnblogs.com/-new/) 阅读(430) 评论(0) [编辑](https://i.cnblogs.com/EditPosts.aspx?postid=7427036) [收藏](http://www.cnblogs.com/-new/p/7427036.html)