|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 锁升级 | 锁状态 | **56 bit** | | | **1 bit** | **4bit** | **1 bit**  **（是否偏向锁）** | **2 bit**  **（锁标志位）** |
| **25 bit** | **31 bit** | |
| 无锁 | Unused | 对象 hashCode | | Cms\_free | 对象分代年龄 | 0 | 01 |
| 偏向锁 | （锁偏向的线程） thread ID (54 bit) | | Epoch (2bit) | Cms\_free | 对象分代年龄 | 1 | 01 |
| 轻量级锁 | 指向栈中锁记录的指针 | | | | | | 00 |
| 重量级锁 | 指向重量级锁的指针 | | | | | | 10 |
| GC 标记 | 空 | | | | | | 11 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 锁升级 | 锁状态 | 优点 | 缺点 | 适用场景 | 优化 |
| 无锁 |  |  | 单线程场景 |  |
| 偏向锁 | 加锁解锁无需额外的消耗，和非同步方法时间相关纳秒级别 | 如果竞争的线程多，那么会带来额外的锁撤销的消耗（重操作，会STW） | 基本没有线程竞争锁的同步场景 | -XX:-UseBiasedLocking 禁用偏向锁 (偏向锁是撤销是重的操作)  延迟启动偏向锁-XX:BiasedLockingStartupDelay=5 |
| 轻量级锁 | 竞争的线程不会阻塞，使用 CAS 自旋，提高程序响应速度 | 自旋是消耗CPU资源的，如果锁的时间长，或者自旋线程多，CPU会被大量消耗 | 适用于少量线程竞争对象，且线程持有锁的时间不长，追求响应速度的场景 | 竞争加剧：有线程超过10次自旋， -XX:PreBlockSpin， 或者自旋线程数超过CPU核数的一半，1.6之后，加入自适应自旋 adapative Self Spinning，jvm 自己控制 |
| 重量级锁 | 线程竞争不适用 CPU 自旋，不会导致 CPU 空转消耗 CPU 资源 | 线程阻塞，响应时间长 | 很多线程竞争锁，且锁持有的时间长，追求吞吐量的场景 | 减少上锁时间、减少锁粒度、锁粗化、锁消除、读写分离 |