操作系统 3

jask

09/14/2024

锁，常用锁与比较

死锁

当两个线程为了保护两个不同的共享资源而使用两个互斥锁，那么这两个互斥锁应用不当的时候就会导致两个线程都在等待对方释放锁，在没有外力的作用下，这就无法运行了，这种情况叫做死锁。

条件

互斥条件：多个线程不能使用同一个资源

持有并等待条件：当线程 A 已经持有了资源 1，又想申请资源 2，但是不释放资源 1，线程 B 持有资源 2，想申请资源 1，但是不释放资源 2，导致死锁

不可剥夺条件：当线程持有了资源在自己使用完之前不能被其他线程获取

环路等待条件：两个线程的获取资源顺序构成了环形图

解决死锁的方法

资源有序分配法：要求线程获取资源的顺序一样，都是先获取资源 A，再获取资源 B

互斥锁和自旋锁

自旋锁

自旋锁加锁失败之后会进入忙等待，直到它拿到锁

这个锁在单核 CPU 无法使用，因为其永远不会放弃 CPU

在多核 cpu 上，如果被所著的代码时间很短，使用自旋锁是更好的选择

cpu 提供的 PAUSE 指令可以实现忙等待

互斥锁

互斥锁加锁失败后，线程会释放 cpu，给其他线程

既然会释放 cpu，对应线程 B 的代码被堵塞，被堵塞是由操作系统完成的，这种情况下会有运行-》暂停-》运行的动作，因为其只能发生在内核态，所以这种情况下会发生线程的上下文切换，产生成本开销

这种锁除了自旋锁适应的条件外都可以考虑

读写锁

这种锁能明确区分读写操作的场景

原理

当写锁被持有，所有的读写锁都会被阻塞

当写锁没有被持有，多个读的线程可以并发的读取资源

场景

根据场景的不同可以分为读优先锁和写优先锁，公平读写锁

读/写优先锁顾名思义，当读/写操作来的时候优先

公平读写锁是将获取锁的线程排队，不论读写按照先进先出的方式依次加锁

悲观锁与乐观锁

悲观锁

概念：自旋锁，读写锁，互斥锁都是悲观锁

悲观锁认为同时修改资源的概率高，很容器出现冲突，所以要先上锁

乐观锁

悲观锁认为同时修改资源的概率高，很容器出现冲突，所以要先上锁

只有在冲突概率非常低且加锁成本非常高的情况下，才使用乐观锁（如文档的在线编辑）