不可扩展的锁是危险的

Non-scalable locks are dangerous

背景介绍

互斥锁是多线程程序中常用的一种线程同步机制。然而在大多数现有的操作系统和应用软件中，互斥锁都是用不可扩展的方式实现的。这些互斥锁给这些软件的综合性能所带来的影响将在该论文中被揭示。

通过测试一些实际的软件，发现当核数较少时，核数的增多能带来较好的性能增益；但当核数超过某一阈值后，软件的性能随着核数剧烈下降。注意到目前最广泛使用的互斥锁是门票锁，基于稳态马尔科夫过程对门票锁的性能进行建模，并对门票锁的性能进行测试，都可以发现核数超过某一阈值后性能急剧下降的特点。

通过测试更多可扩展的锁，可以发现，将软件中的门票锁替换为MCS锁后，可以大幅提升软件的整体可扩展性。

门票锁

算法简介

在Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors一文的阅读报告中介绍了门票锁的实现机制。在这里用C语言代码做详细描述

struct ticket\_lock\_t {

int current\_ticket;

int next\_ticket;

};

void ticket\_lock(ticket\_lock\_t\* lock){

int t = atomic\_fetch\_and\_add(&lock->next\_ticket, 1);

while(t != lock->current\_ticket)

; /\* spin \*/

}

void ticket\_unlock(ticket\_lock\_t\* lock){

lock->current\_ticket++;

}

每个线程发起获取锁的请求后，将会得到一张门票t，且每一个等待中的线程都拥有一张唯一的门票，且先发起请求的线程得到的门票号码更小。而每个线程在释放锁时将当前可进入的门票号加一，使得下一个等待者可以进入临界区。

缓存一致性协议

在讨论门票锁的性能模型前，我们需要对缓存一致性协议建立模型。在现有的多核处理器中，每个核都拥有私有的缓存。当两个核访问内存中同一个数据块时，各自的私有缓存都持有这块数据的一个副本；而此时若其中一个核对该数据块做了修改，另一个核的私有缓存中的副本则变为无效；当再次访问该数据块时，需要从主存或核的缓存中取该数据块，记这个过程的时间开销为。

再将核数增多至个。当这个核都持有同一个数据块的副本时，其中个核进行一次写，其余个核不断地重复读。当数据块被写后，这个核都需要重新取来这个数据块。这里假设是从核a的缓存中取，且核a的缓存一次只能传输一个数据块，那么这个核取得该数据块的平均延迟为。

性能模型

假设总线程数为，根据当前正在等待的线程数可以分为个状态。

在自旋过程中，需要不断地读，而每个线程释放锁时都会写，从而使所有等待中的线程都收到缓存无效化的消息，再次读取时所有这些等待中的线程都需要通过核间通信网络，平均延迟为。假设临界区的执行时间为，则从状态转移到状态的转移率为

假设在单线程、无竞争的情况下，连续两次进入临界区之间的时间间隔为，那么从状态转移到状态的转移率为

根据稳态假设

可得[[1]](#footnote-1)

再根据

可得

于是平均等待时间

将马尔科夫过程的稳态概率带入一些典型参数（，）进行计算得到概率分布如图1。在这组参数下，当核数少于32时，分布曲线呈左偏，有很大的概率等待者数量不多；而当核数达到32时，分布曲线变为右偏，有很大的概率等待者的数量接近。

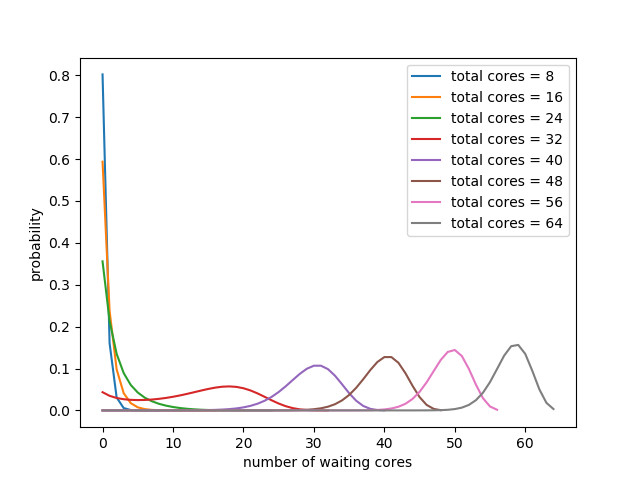


图 1

根据该模型的预测，门票锁性能随着的增大到一定值时会突然出现急剧下降，这也与之前测试到的某些实际软件的可扩展性相似。

更多底层测试程序（microbenchmark）的测试表明，当临界区执行时间固定时，门票锁的性能与该模型非常接近；当临界区自旋时间不固定时，该模型也具有较高的精确度。

该性能模型说明，门票锁随着处理器核数增多至一定值时突然急剧下降的现象并非某一特定硬件平台上所特有的现象，而是一个普遍规律。

在一些实际应用程序中也发现了类似的性能急剧下降现象，如FOPS、MEMPOP、PFIND和EXIM，分别在2核、9核、15核和39核以上出现性能急剧下降。

可扩展的锁

作者测试了MCS锁及其K42、CLH等变种的性能，发现这些锁都具有较好的可扩展性。由于原理类似，这些锁之间的性能差异不大。作者选择用MCS锁来替换Linux内核中的门票锁，修改了内核代码，并用前面提到的四个程序进行测试，都达到了3.5到16倍的性能提升。

结论

不可扩展的锁在多核处理器上性能较差，已经是一个公认的结论，但这些锁的性能对整个程序的性能造成的影响还没有引起足够的重视。该论文指出了不可扩展的锁给整体性能带来的巨大损失——一旦核数超过某个值，性能将会急剧地下降，并通过基于马尔科夫过程的性能模型解释了这一下降。通过将不可扩展的锁替换为可扩展的锁，使得某些原本性能急剧下降的程序避免了这一下降。可扩展的锁和无锁数据结构是将会达到高性能所必须的。

1. 原文中此处漏掉了除以2 [↑](#footnote-ref-1)