

本文为近期工作心得总结。

1. debug心得

1.1 基础数据类型的跨平台性

存储和操作业务数据的数据类型，比如一帧图片的数据，最好用定长数据类型，如uint32_t，而不用变长数据类型，如unsigned long。在32位操作系统上，unsigned long是4字节，而64位操作系统上，unsigned long是8字节。这往往会引起难以觉察的bug。

1.2 内存分配和释放

在多线程程序中分配内存，最好遵循谁分配内存谁负责回收的原则。

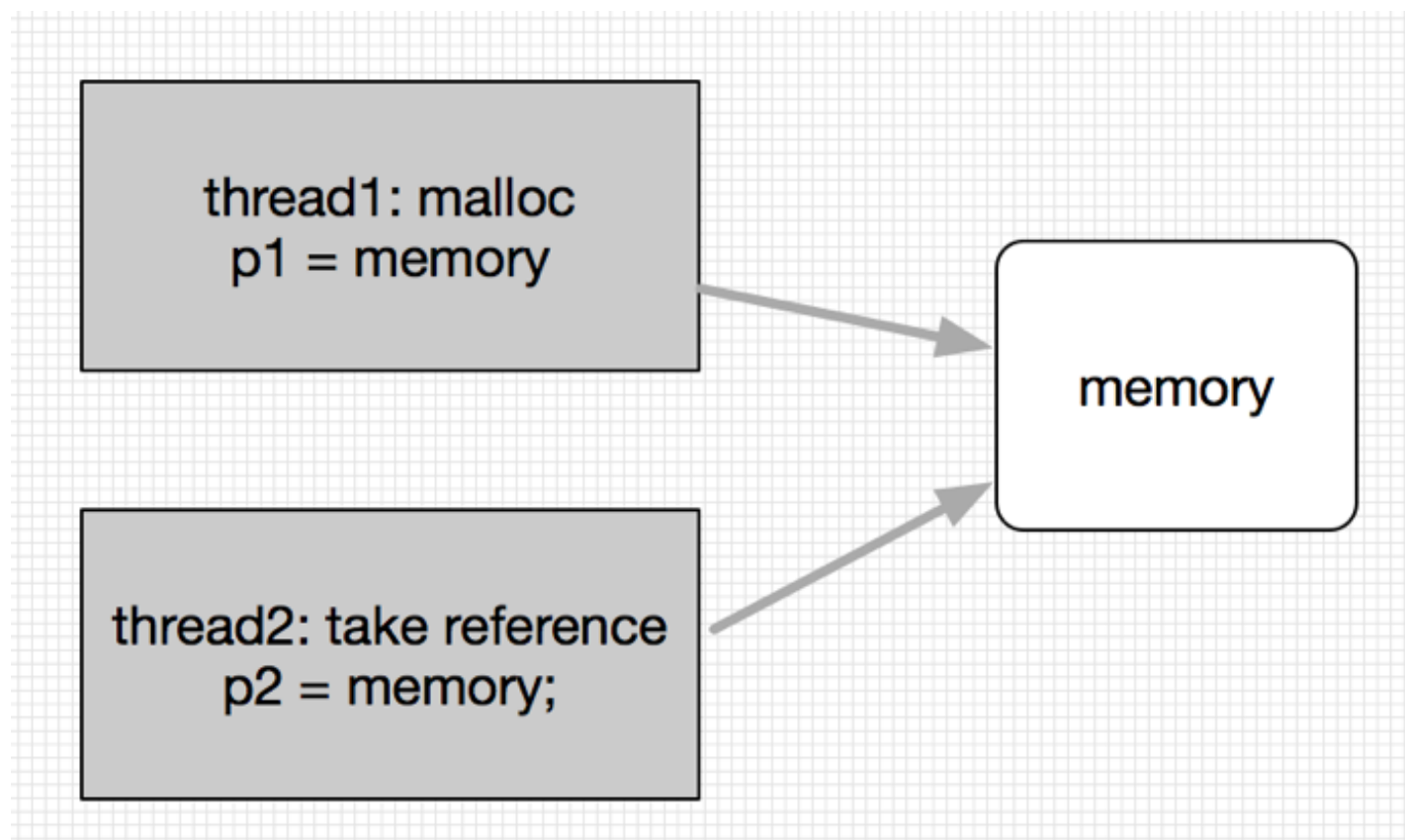


图1 多进程间引用同一块动态分配内存

如图1所示，线程1分配了一块内存memory并用p1引用，线程2用p2引用。此后，若线程1释放了memory，线程2再引用p2就会发生错误；反之亦然。这种情况下，最好由线程1既负责分配又负责释放memory；同时在保证时序的情况下在线程2中引用。

1.3 线程互斥

使用pthread_mutex_t进行线程互斥时，要先初始化锁，然后加锁/释放要配对进行。要确保在函数中的每一条非正常退出路径上都释放了锁。函数是程序的最小功能单元，要多思考，力争写出简洁明了思路清晰的函数模块。

1.4 linux进程CPU占用率分析

首先使用top命令获取目标进程的pid。
然后用top -H p pid查看进程内部线程的CPU占用率情况。
然后用gstack pid > gstack.log查看目标线程的函数调用栈。

1.5 多路复用和轮询

前者比后者好太多，后者经常把CPU吃完。

2. 工作心得

2.1 要严肃对待Coding

一个bug在编写代码阶段解决，花费的时间大约在10分钟；在单元测试阶段解决，花费的时间可能需要30分钟；在code review阶段如果多花10分钟，说不定也能够发现问题。但是如果在集成测试或者联合调试的时候解决bug，可能需要一两天。如果在产品线上发现bug，那么解决起来需要花费更长时间更多精力，同时用户体验也会受到极大影响。

所以，要用严肃认真的态度去Coding，多思考，多分析，多考虑不同情况。做好单元测试，测试覆盖率要尽可能广，最好能够100%。做好Code Review，如果条件允许邀请同事一起代码评测。慎重对待每一次commit，不要随意让代码进入Code Base。这些事情在思科的时候能够执行的很好，现在做的非常不好，切记切记。

2.2 程序员的核心竞争力

我认为程序员的核心竞争力包括两点：格局和极致。做人要有大格局，眼光长远；对技术要高屋建瓴，整体把握。做具体事务的时候要严肃认真，把事情做好做到极致。

3. 总结

脑子是个好东西，要好好用。