目录
- Python教程
  - Python简介
  - 安装Python
  - 第一个Python程序
  - Python基础
  - 函数
  - 高级特性
  - 函数式编程
  - 模块
  - 面向对象编程
  - 面向对象高级编程
  - 错误、调试和测试
  - IO编程
  - 进程和线程
    - 多进程
    - 多线程
    - ThreadLocal
    - 进程 vs. 线程
    - 分布式进程
  - 正则表达式
  - 常用内建模块
  - 常用第三方模块
  - virtualenv
  - 图形界面
  - 网络编程
  - 电子邮件
  - 访问数据库
  - Web开发
  - 异步IO
  - 实战
  - FAQ
  - 期末总结

关于作者

廖雪峰 V 关注 编辑过
编辑推荐

廖雪峰
自己的Python课程
Python商业爬虫全解码
让天下没有爬不到的数据！
Python爬虫引擎 + 数据分析
Python高速存储 + 深度学习
.....
找廖雪峰老师

廖雪峰老师
自己的Java课程
Java高级架构师
更专业 更权威
原创分析专题 | 微服务架构专题
高并发分布式专题 | 深度优化专题
.....
找廖雪峰老师

unicef 联合国儿童基金会
一个纸杯 不应堂堂地成为你的垃圾
¥2 向德国儿童基金会捐款

unicef 联合国儿童基金会
一个纸杯 不应堂堂地成为你的垃圾
¥2 向德国儿童基金会捐款

进程 vs. 线程

阅读: 150200

我们已经了多进程和多线程，这里这期单讲线程的相关内容。线程，我们来讲这一下这两种方式的优缺点。

首先，要实现多任务，通常我们会设计Master和Worker模式，Master负责调度任务，Worker负责执行任务。因此，多进程环境下，通常是一个Master，多个Worker。

如果用多进程实现Master-Worker，主进程就是Master，其他进程就是Worker。

如果用多线程实现Master-Worker，主线程就是Master，其他线程就是Worker。

多进程模式最大的优点是稳定性高，因为一个子进程崩了，不会影响主进程和其他子进程。（当然主进程挂了所有进程就全挂了，但是Master进程只负责分配任务，挂掉的概率低）著名的Apache最早就是采用多进程模式。

多进程模式最大的劣势是创建进程的代价大，在Unix/Linux系统下，用fork调用还行，在Windows下创建进程开销巨大。另外，操作系统能同时运行的进程数也是有限的，在内存和CPU的限制下，如果有几千个进程同时运行，操作系统调度能带来很麻烦。

多线程模式通常比多进程一点，但是仍然不到最快。因为，多线程模式也存在缺点：任何一个线程挂掉都可能直接造成整个进程崩溃，因为所有线程共享进程的内存。在Windows上，如果一个线程执行的代码出了问题，你知道可以重启进程的提示：请程序执行了非法操作，即将关闭，就这么结束了线程还不行呢，但是操作系统会强制结束整个进程。

在Windows下，多线程的执行效率比多进程要高，所以微软的三服务器版本和高性能版本，四个多线程版本的SQL Server性能要比四个多进程版本的MySQL要高。为了最解决这个问题，比如Apache提出了改单多进程+多线程的混合模式，具体怎么搞就有很多种方案了。

线程切换

无论是多进程还是多线程，只要数量一多，效率肯定上不去，为什么呢？

我们可以比喻，假设你不是正在做编程，每天晚上要做的事：洗澡，看电影，吃零食，看书。

如果你白天这样做了：晚上，你就可以一个接一个地做了，这样做，你的效率就很高。

但是，如果你是这样做的：你开始洗澡，洗到一半的时候，你的电话响了，你不得不去接电话，聊了一会，你回到了卫生间，发现你的水烧开了，你不得不去关了水，找了杯子，泡了杯茶，你端着茶杯回到了卫生间，你的MP3播放完了，你不得不去换了一张CD，你的CD太多了，你翻了半天才找到你要换的那张CD，你换上CD，你才可以继续洗澡。

这样做，你的效率就很低了。

线程切换的开销也是如此，这里面的开销大小，视具体的操作系统而定。

但是，切换的总是有代价的，比如以前正在执行的一个线程，现在要切换成另一个线程来执行，这个切换，需要把现在的进程的一切状态保存好，以便以后再次切换到这个线程时，恢复现场，继续执行。

对于多线程来说，因为是在一个进程内部切换，线程共享内存和全局变量，因此，在切换时，还要保存栈和寄存器的内容，但是，内存地址没有切换，因此，切换的开销要小。

对于多进程来说，每个进程有自己的内存空间，这个内存空间在切换后，操作系统还要刷新缓存，才能正常运行。由于刷新了缓存，比如Linux的TLB和CPU的L1/L2/L3缓存，很多指令的执行速度也会因为缓存失效而大大地降低。

总之，多任务环境下，线程切换的开销要小，因此，多线程的效率高。

计算密集型 vs. IO密集型

是否采用多任务的第二个考虑是看任务的类型，我们可以把任务分为计算密集型和IO密集型。

计算密集型任务的特点是要进行大量的计算，消耗CPU资源，比如计算圆周率、对视频进行高清解码等等，全靠CPU的运算能力。这种计算密集型任务虽然也可以用多任务完成，但任任务切换过的时间越多，CPU执行任务的效率就越低，所以，计算密集型任务同样还是开启多进程的效率较高。

计算密集型任务的另一个重要特点是CPU消耗很高，代码运行的效率至关重要。Python这样的脚本语言运行效率很低，完全不适合计算密集型任务。对于计算密集型任务，最好用C语言编写。

第二种任务的类型是IO密集型，涉及到网络、磁盘IO的任务都是IO密集型任务，这种任务的特点是CPU消耗很少，任务的大部分时间都在等待IO操作完成（因为IO的速度远远低于CPU和内存的速度）。对于IO密集型任务，任务越多，CPU效率越高，但仅仅会有一个限度。常见的大型计划系统的OS都是基于这种模型。

IO密集型任务的工程师，编写异步程序的时间成本会比较高。因此，我们需要更多的工程师来编写Python这样高开发效率的脚本语言，毕竟天花板很低的计算密集型任务，最终会的瓶颈基本只剩下效率高低（代码质量高）的因素，脚本语言高效。（语言质量）

异步IO

考虑到CPU和IO之间的巨大速度差异，一个任务完全时间花费在等待IO操作的时候是很常见的情况。比如，我们才需要多进程模型或者多线程模型来支持多任务同时执行。

现代操作系统对IO操作已经做了巨大的改进，最大的特点就是支持异步IO。如果当时用操作系统提供的异步IO接口，就可以用单进程单线程模型来执行多任务，这种计算机模型称为事件驱动模型，Nginx就是支持异步IO的Web服务器，在单核CPU上采用单进程模型就可以高效地支持多任务。在多核CPU上，可以运行多个进程（数量与CPU核心相同），充分利用多核CPU。由于系统总是优先调度计算密集型任务，因此操作系统调度效率高的进程，单个进程的效率也非常高。

对应到Python语言，单线程的异步编程模型称为协程，有了协程支持，就可以基于事件驱动编写高效的多任务程序。我们在也后面还会对协程有介绍。

说后有收获可以请作者喝瓶啤酒，说后有疑问请加群讨论：

还可以分享给朋友：
QQ 分享给好友

上一节 | 下一节

廖雪峰
自己的Python课程
Python商业爬虫全解码
让天下没有爬不到的数据！
找廖雪峰老师

ACM金牌得主
世界顶尖算法一线教研计算机核心技术
人工智能与自然语言/计算机视觉课程培训
BEIJING/SHANGHAI/SHENZHEN/HANGZHOU
无offer退全款

廖雪峰推荐
JAVA进阶教程
原价1699元
0元领取
立即查阅

腾讯云
高性能云服务器首台5折
企业上云解决方案，最大20Gbps内网带宽，40亿PPS
立即购买

腾讯云
学生服务器体验官 10元价
1核2G 1M带宽 50GB流量
立即开通
跨网云对接VIP，最高赠100元/月
腾讯云最新推出的云对接VIP，多线接入，高质量线路，稳定可靠，性价比极高，BGP多线可选
查看详情

1 python免费公开课
编课模式：在线直播+课后辅导，从零基础到项目实战上线班
报名不了吗

2 python全技实战课程限时免费领取
100天最长就业保障体系 零基础到项目就业，全流程学习路线，直播授课，辅导答疑，python开发python开发技术
测试方案

评论

- 多线程模式稳定的缺点的原因是什么呢？一个线程挂掉都可能直接造成整个进程崩溃
  - 小明 commented on June 9, 2015 11:12 PM. Last updated on June 9, 2015 11:12 PM
    但过期，多进程模式一个子线程挂掉并不不影响主线程的执行，现在理解明白。
    View 全部评论 | 回复
- 有个疑问
  qwert1 commented on August 25, 2015 2:37 PM. Last updated on May 29, 2016 7:17 PM
  切换内容不可能每个都同，两道菜肴同一时间都等待时间不同，不应该说谁切换了不同菜肴不同时间（即切菜时间和炒菜时间）。
  - 廖雪峰 commented on August 27, 2015 3:37 PM
    cpu不不会切换任务，是操作系统负责切换
  - 廖雪峰 commented on August 27, 2015 3:37 PM
    cpu的工作是一直要执行指令，切换任务是操作系统保存要执行指令，然后把某上次的事保存，代码不是直接给为内存就能执行的
  - lan13 commented Apr 29, 2016 4:03 PM
    原理，，应该是两次发现理解了，确定，如果从原理的角度来说是理解的过程，如果不是多核的情况下，应该是中断，所以不切换大致了，不是超多核的情况的话也可以理解成，但是多核的话就得上第四点理解了
  - 廖雪峰 commented on September 16, 2016 5:31 AM
    这样不够完，因为一般情况下，CPU切换时间远远大于等待时间，所以，切换的时间可以忽略不计。
    View 全部评论 | 回复
- 王帅 commented on February 10, 2017 8:34 PM
  但不懂小姐姐为什么windows没有fork
  View 全部评论 | 回复
- 小一 commented on January 9, 2019 7:27 PM
  一个CPU就是一条道子
  - Nick commented on January 29, 2019 11:15 AM
    是单核 但是多核的话那个不对这样也不能一边听歌一边玩游戏了，为了保证公平能够同步修改资源的锁和信号，所以公平调度
  - 刘帅一天天 commented on January 29, 2019 2:11 PM
    增加了单一下了解决饥饿现象的发展问题的情况解决方案。当广一个线程之前，你先问问内存中那一块内存空间，有内存，这时内存中保存的一块像随机表那行保存指针，它代表内存偏移量，告知程序应该跳过表的这表的偏移量，之后才开始执行程序。这样这样程序的运行时间长短，也随机性，每段程序的偏移量的长短也长短，基于表的偏移量的长短，程序每段长入的表的长短长，程序每段的长短了。
    这大概是多线程的内存中的重要的总结吧，它不只是内存中的一个区只分配给一个，好了，就说说吧。
  - 刘帅一天天 commented 2 days ago
    1 0 1 0 1 0