**什么是高并发**

高并发（High Concurrency）是互联网分布式系统架构设计中必须考虑的因素之一，它通常是指，通过设计保证系统能够同时并行处理很多请求。

高并发相关常用的一些指标有响应时间（Response Time），吞吐量（Throughput），每秒查询率QPS（Query Per Second），并发用户数等。

响应时间：系统对请求做出响应的时间。例如系统处理一个HTTP请求需要200ms，这个200ms就是系统的响应时间。

吞吐量：单位时间内处理的请求数量。

QPS：每秒响应请求数。在互联网领域，这个指标和吞吐量区分的没有这么明显。

并发用户数：同时承载正常使用系统功能的用户数量。例如一个即时通讯系统，同时在线量一定程度上代表了系统的并发用户数。

思考，高并发的本质是什么？

**如何提升系统的并发能力**

互联网分布式架构设计，提高系统并发能力的方式，方法论上主要有两种：垂直扩展（Scale Up）与水平扩展（Scale Out）。

垂直扩展：提升单机处理能力。垂直扩展的方式又有两种：

（1）增强单机硬件性能，例如：增加CPU核数如32核，升级更好的网卡如万兆，升级更好的硬盘如SSD，扩充硬盘容量如2T，扩充系统内存如128G；

（2）提升单机架构性能，例如：使用Cache来减少IO次数，使用异步来增加单服务吞吐量，使用无锁数据结构来减少响应时间；

水平扩展：分布式，多进程，多线程，携程，雾计算….

**多进程/多线程/协程**

　　1、进程

进程是具有一定独立功能的程序关于某个数据集合上的一次运行活动,进程是系统进行资源分配和调度的一个独立单位。每个进程都有自己的独立内存空间，不同进程通过进程间通信来通信。由于进程比较重量，占据独立的内存，所以上下文进程间的切换开销（栈、寄存器、虚拟内存、文件句柄等）比较大，但相对比较稳定安全。

　　2、线程

线程是进程的一个实体,是CPU调度和分派的基本单位,它是比进程更小的能独立运行的基本单位.线程自己基本上不拥有系统资源,只拥有一点在运行中必不可少的资源(如程序计数器,一组寄存器和栈),但是它可与同属一个进程的其他的线程共享进程所拥有的全部资源。线程间通信主要通过共享内存，上下文切换很快，资源开销较少，但相比进程不够稳定容易丢失数据。

　　3、协程

**协程是一种用户态的轻量级线程，**协程的调度完全由用户控制。协程拥有自己的寄存器上下文和栈。协程调度切换时，将寄存器上下文和栈保存到其他地方，在切回来的时候，恢复先前保存的寄存器上下文和栈，直接操作栈则基本没有内核切换的开销，可以不加锁的访问全局变量，所以上下文的切换非常快。

二、区别：

　　1、进程多与线程比较

线程是指进程内的一个执行单元,也是进程内的可调度实体。线程与进程的区别:  
1) 地址空间:线程是进程内的一个执行单元，进程内至少有一个线程，它们共享进程的地址空间，而进程有自己独立的地址空间  
2) 资源拥有:进程是资源分配和拥有的单位,同一个进程内的线程共享进程的资源  
3) 线程是处理器调度的基本单位,但进程不是  
4) 二者均可并发执行

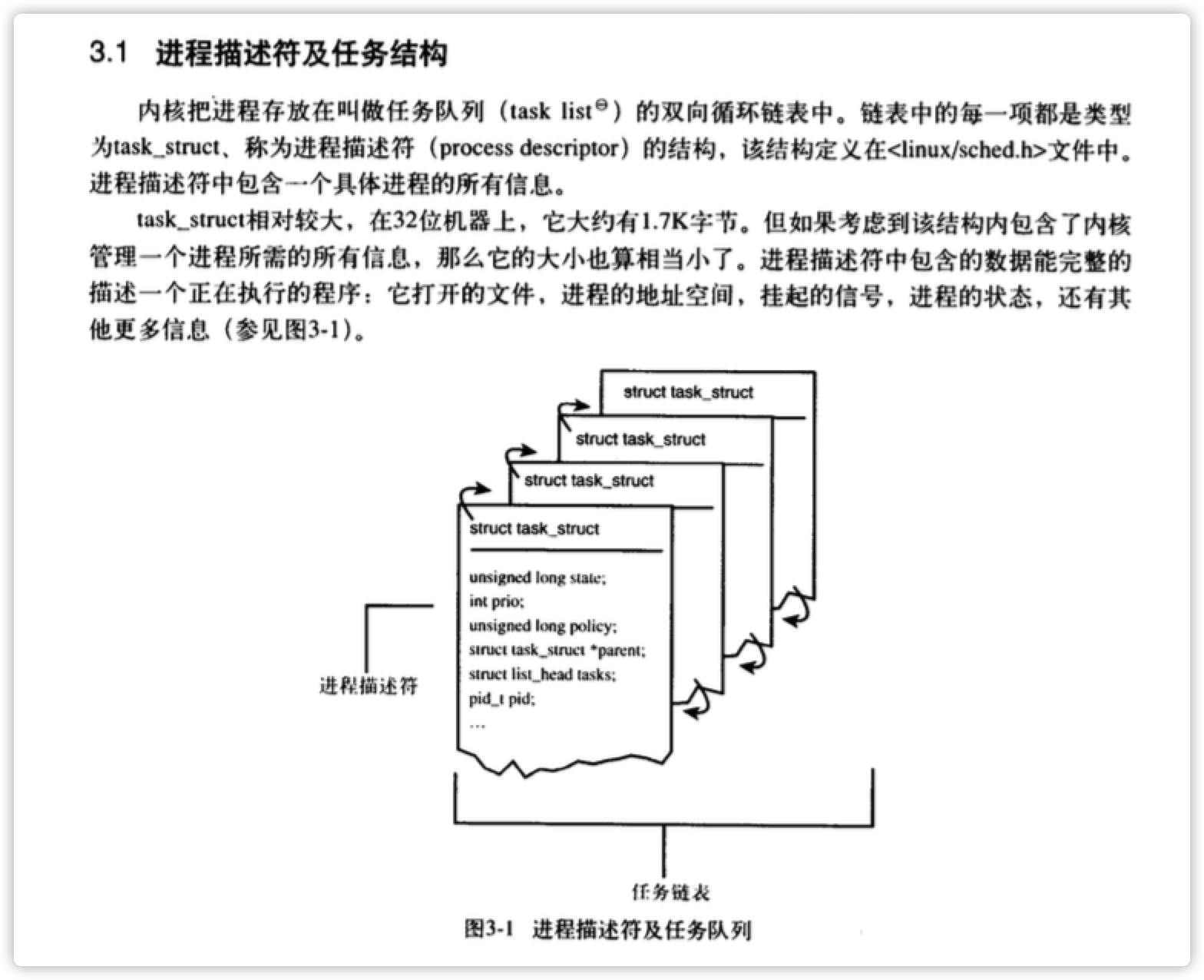
5) 每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口，但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制

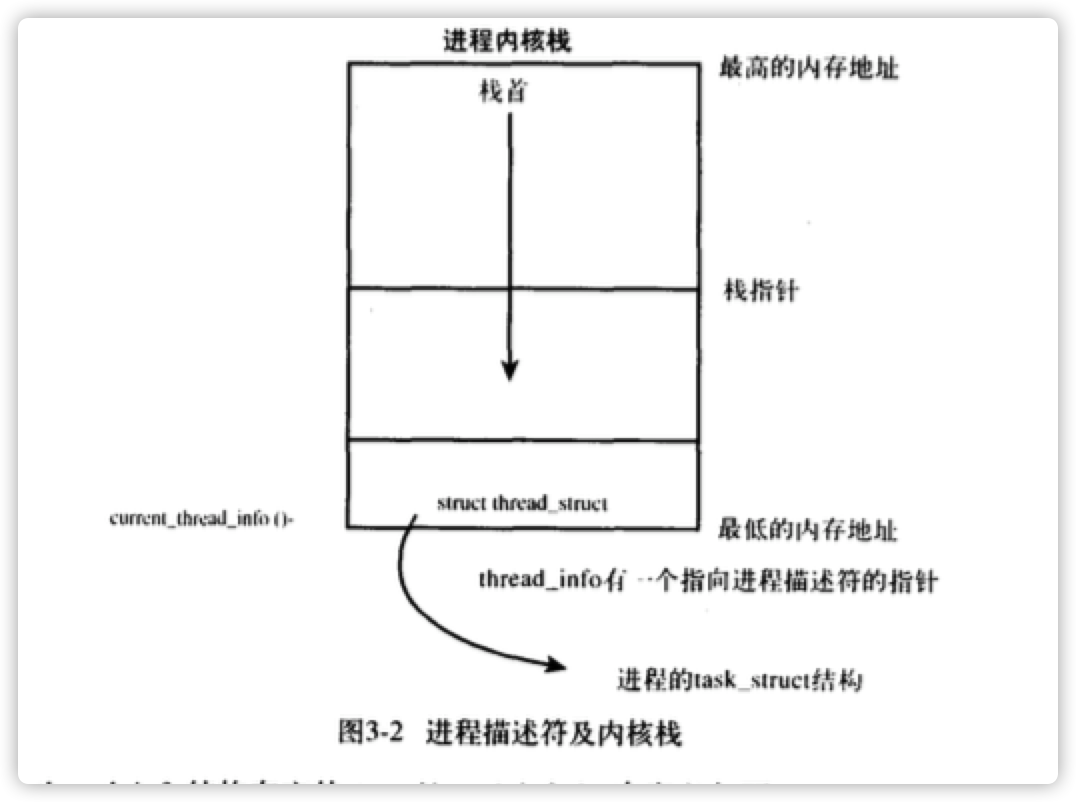
　　2、协程多与线程进行比较

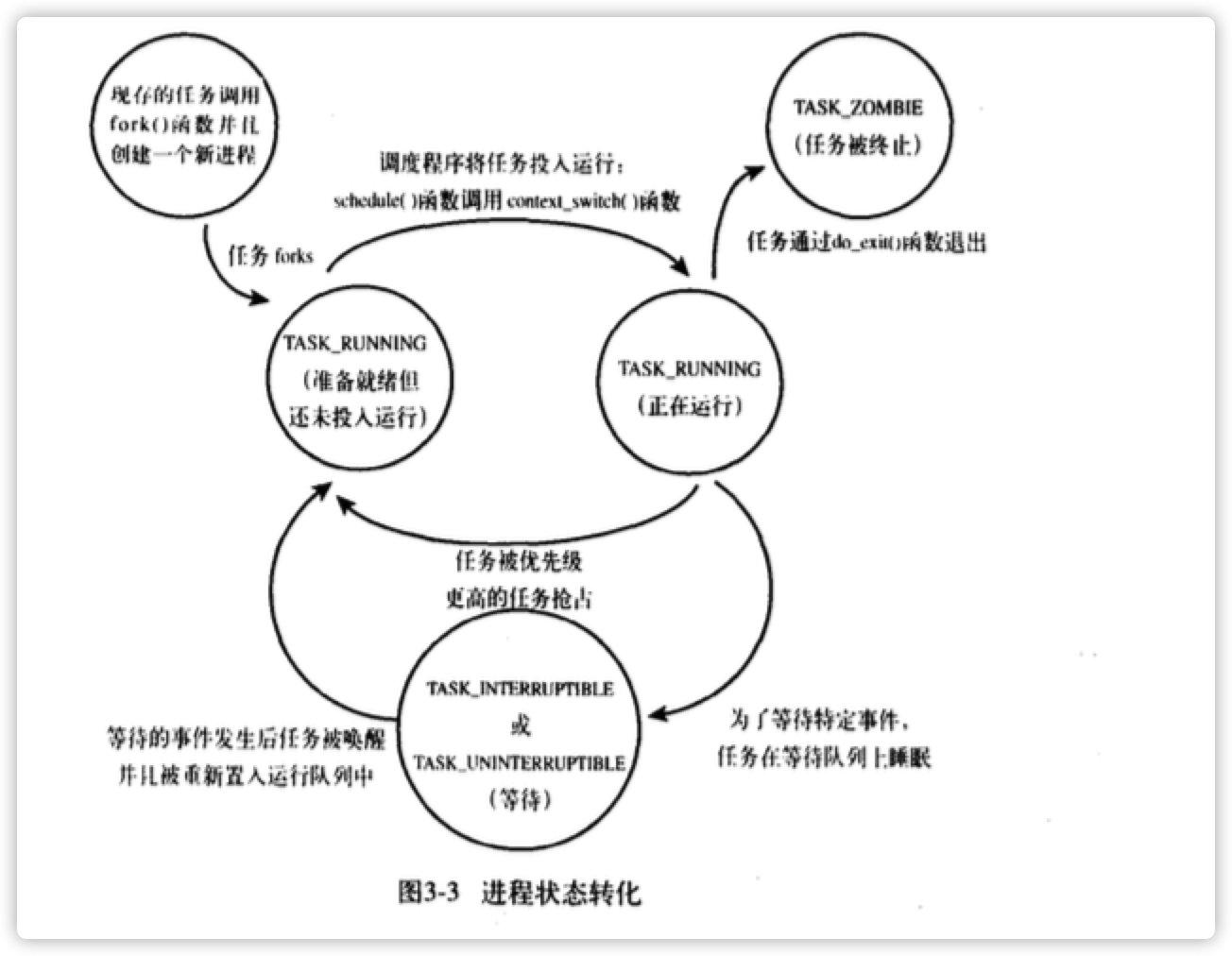
1) 一个线程可以多个协程，一个进程也可以单独拥有多个协程，这样python中则能使用多核CPU。

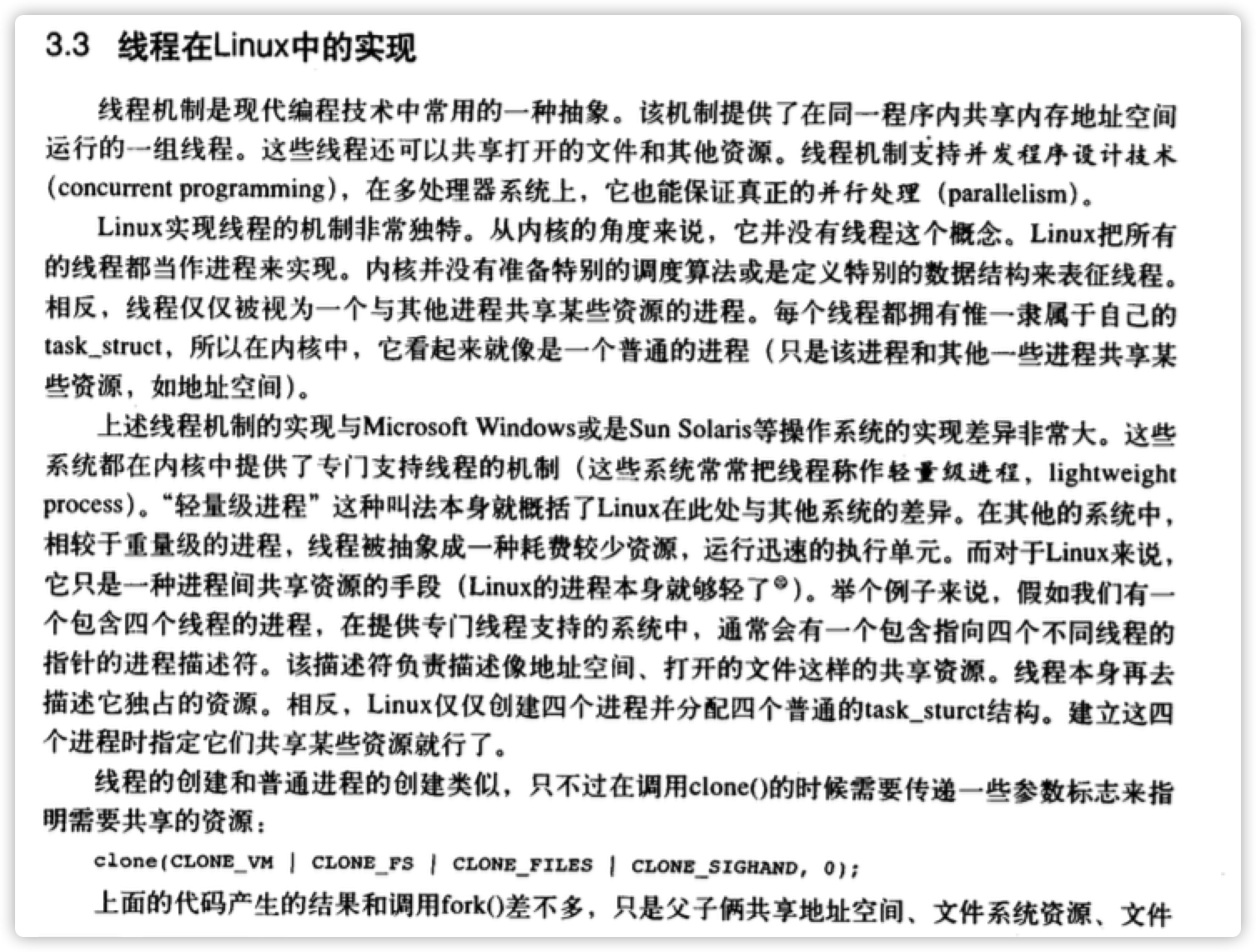
2) 线程进程都是同步机制，而协程则是异步

3) 协程能保留上一次调用时的状态，每次过程重入时，就相当于进入上一次调用的状态

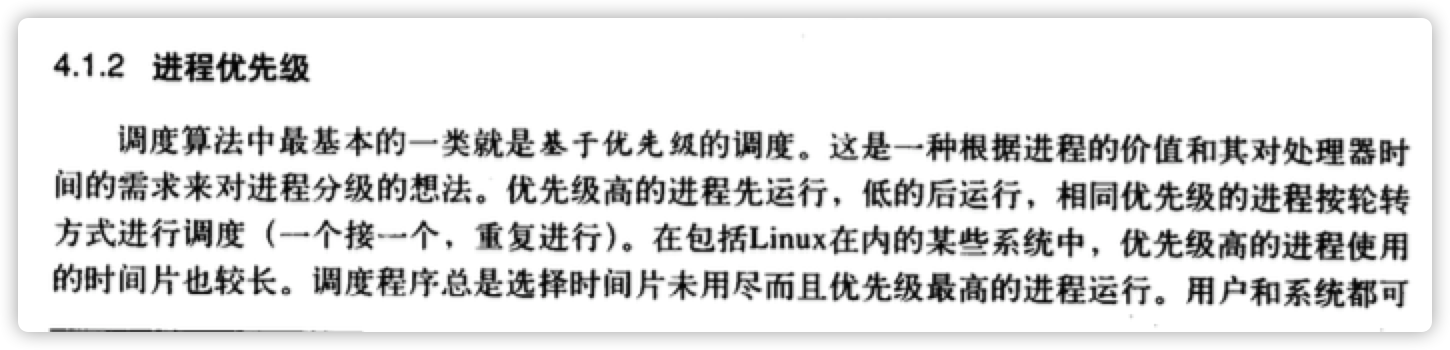


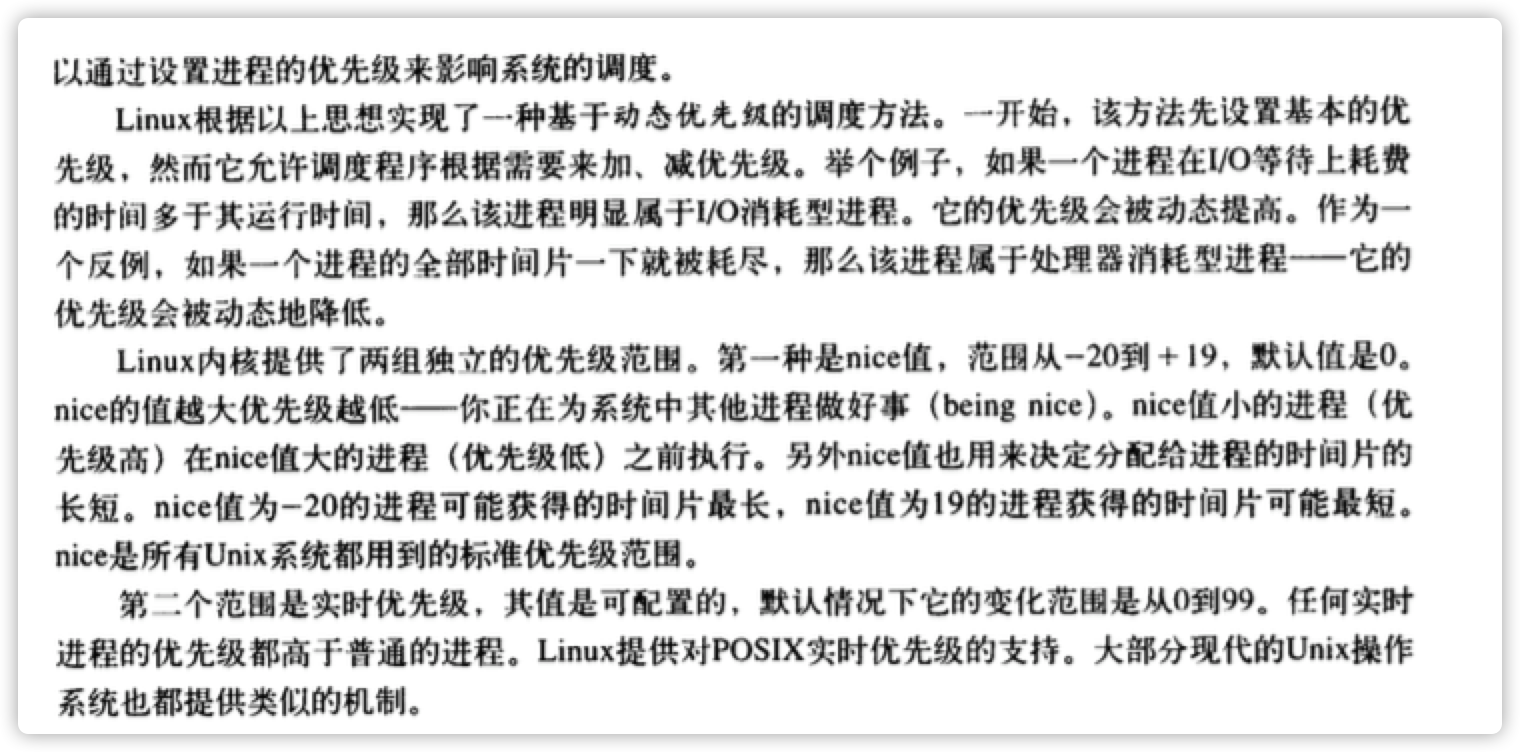






进程调度





cfs

<https://www.cnblogs.com/tianguiyu/articles/6091378.html>

Posix

<https://www.cnblogs.com/1130136248wlxk/articles/5308863.html>

**高并发编程面临的首要问题-进程/线程间同步**

1 互斥锁

2 信号量

3 消息队列

4 管道

5 共享变量（非线程安全）

# IO多路复用的三种机制Select，Poll，Epoll

<https://www.jianshu.com/p/397449cadc9a>