os如何感知进程控制其资源

操作系统最像财务

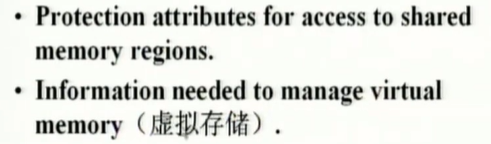
进程和资源的当前状态是要有table的

表之间还要关联

例如

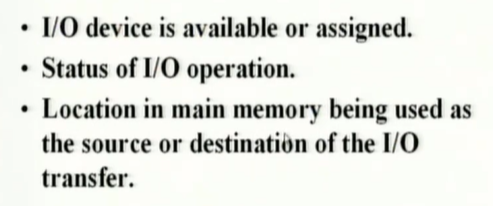
Memory tables：allocation of main memory to processes

进程对应外存的分配情况 secondary memory

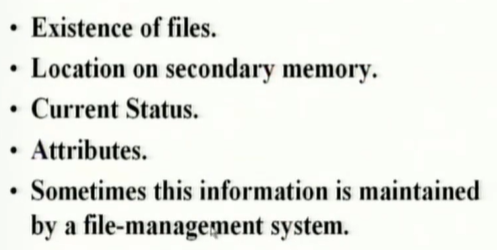


IO table

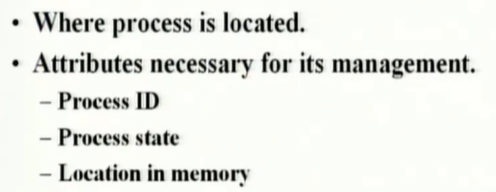
设备表

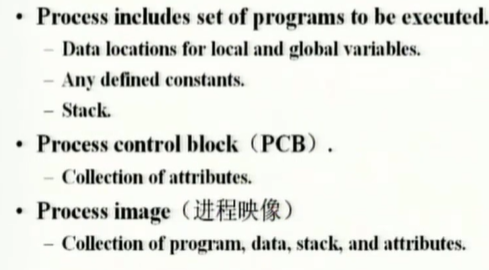


文件表



Process tables







线程

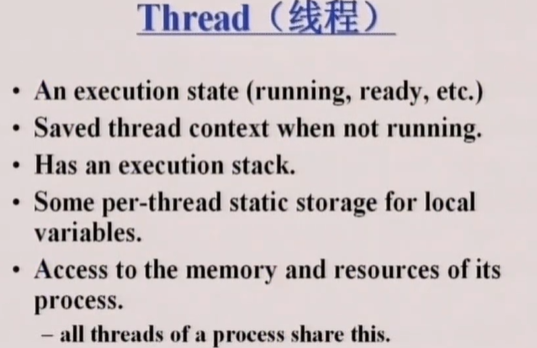
还有印象吗，真是白学了

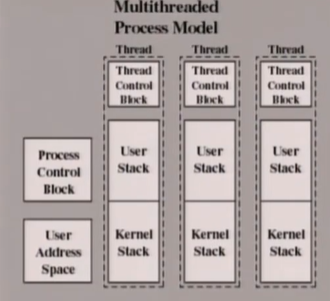
进程是资源分配单位

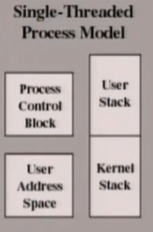
线程是调度资源的单位，不再分配资源，共享进程的资源

减轻系统开销在多道

线程也是程序的执行过程







创建线程的时间比创建进程的时间要少

终止一个进程比终止一个线程花更多时间

同一进程内的线程切换节省时间

因为在一个进程内的线程交流不用激活内核

挂起一个进程会挂起其所有的线程

windows和linux其实是单进程多线程的操作系统

只有init这个初始进程

而unix是多进程多线程

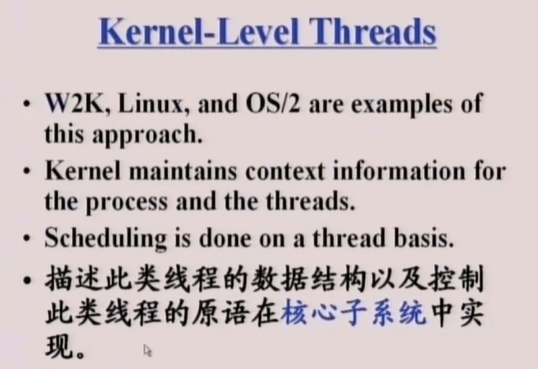
线程的状态



用户级线程



没有TCB



sun工作站在大型机里面是混合型的

先将调度schedule//刚刚赶上

计组刚刚讲过这都可以？

调度需要满足什么

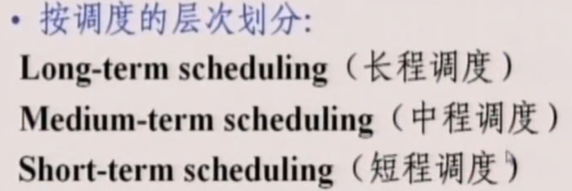
满足用户对响应时间的要求

基于个人的感觉

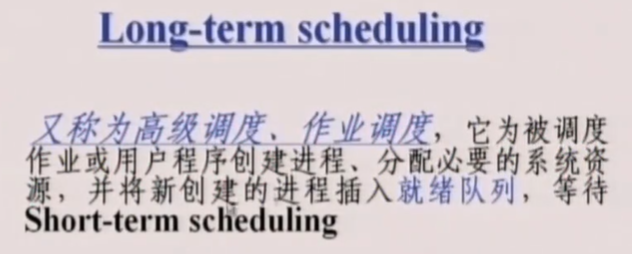
系统吞吐量

公平性：对各类进程都要公平，在某些系统

调度：在队列中进行选择才是调度呗



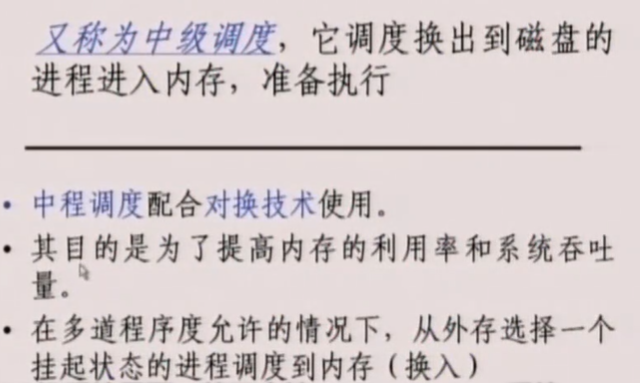
长程调度：基于作业的，从外存把作业放进内存里再创建进程的调度



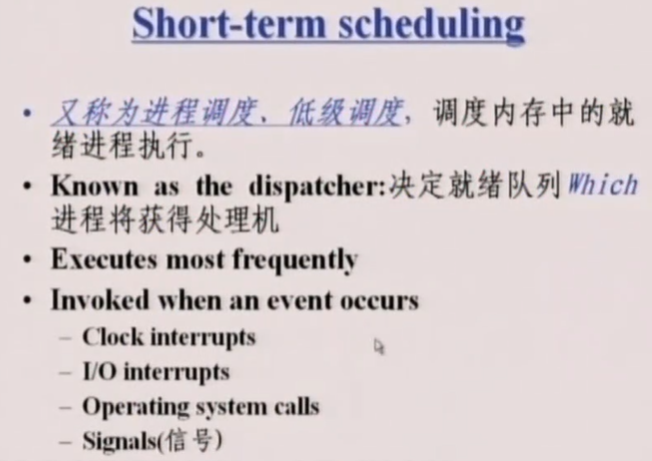
有作业终止了要新进嘛

处理器空闲时间达到阈值

中程调度：在内外存之间但进程已经创建

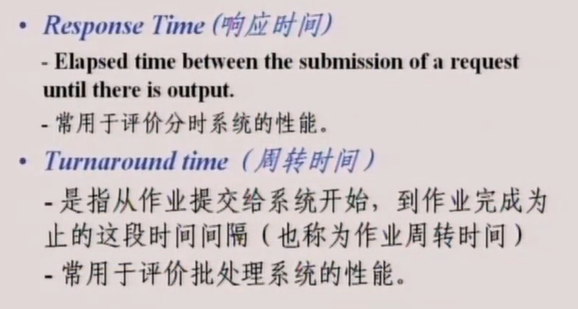


短程调度：发生在内存里面（我们比较关心的）



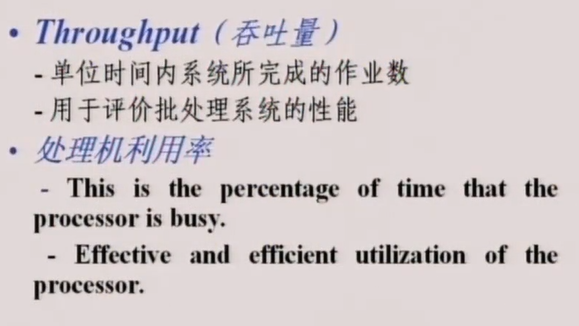
考虑进程调度算法需要考虑的是

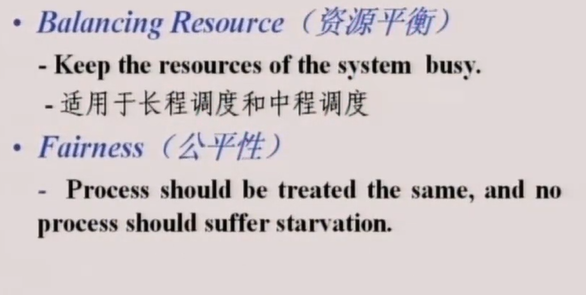
面对用户



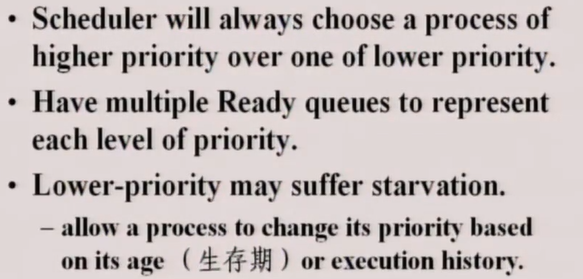
计算机最大的原则 best effort 则会也是没办法的事情嘛

面对系统来说



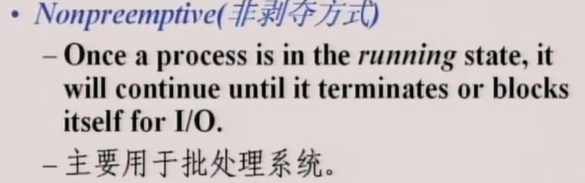


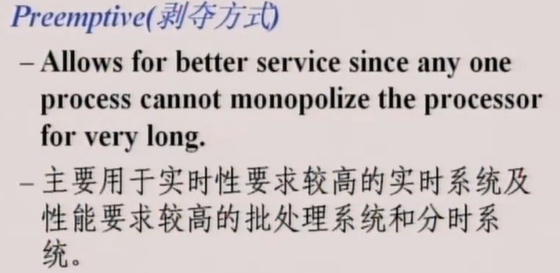
优先级：



出身贫寒但是节节攀升

具体的调度算法



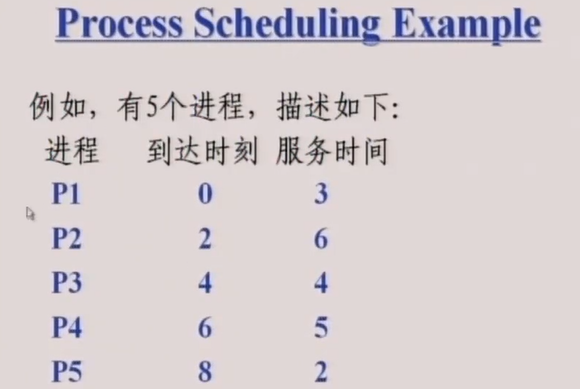


巨大的压力，又有什么意义nei

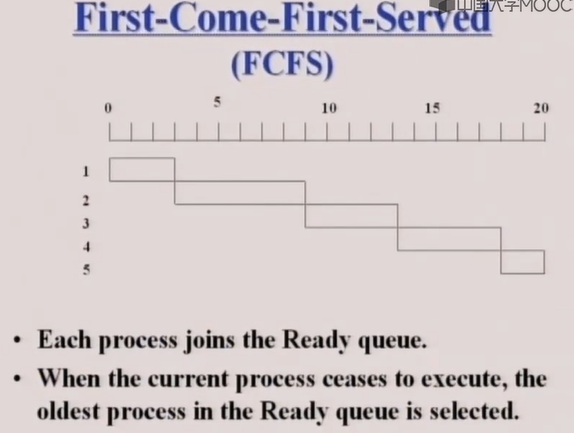
还不是得转变自己的观点

真是b了🐕了

来具体分析一下：平均周转算法



先来先服务谁都能想到么



当前进程结束就先找老年人咯

我们来编这种算法也是能很快实现的🐎，但对短进程不公平

分一下类，按照不同优先级来排队，素质有问题么。

这公开课可真是尴尬

一般来说FCFS结合其它调度算法

多个就绪队列，再使用FCFS

悲天悯人

轮转调度法：给你固定长时间片

老师喂饭就是这样的一种方法



一人喂一口饭

IO型的回到一个就绪队列，但是优先级高时间片短

短进程先服务的调度也是可以的嘛

但如果短进程太多了，那长进程可不是不公平了呗



工程与理论之间真的是差别很多

考虑怎么用，这个是很完美的算法但是工程上可能是不会去实现的

