D. 进程的调度

作用是解决 CPU 选择先运行哪个进程的算法。为了更好的理解调度，先介绍两个概念：计算密集型和 I/O 密集型。计算密集型指该进程占用较长时间的 CPU 和较短时间的 I/O 等待，而 I/O 密集型恰好相反。一个基本思想是让 I/O 密集型进程尽可能多的得到调用机会。

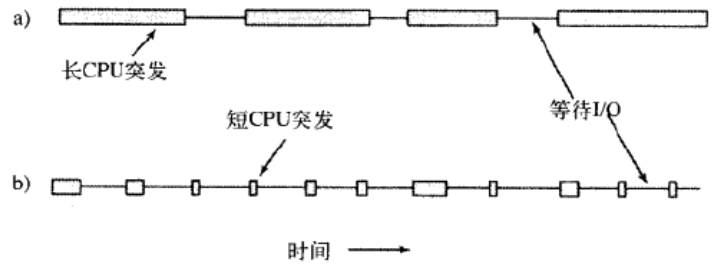

图2-38 CPU的突发使用和等待I/O的时期交替出现：a) CPU密集型进程 ； b) I/O密集型进程

什么时候调度？

1. 父进程和子进程可以被合法的选择。
2. 一个进程退出时必须做出调度选择，否则将运行空闲进程。
3. 当一个进程被阻塞时，必须选择另一个进程运行。阻塞有时成为选择的因素。
4. I/O 中断发生时，必须做出选择。

一. 调度算法的目标

所有系统

公平——给每个进程公平的CPU份额
策略强制执行——看到所宣布的策略执行
平衡——保持系统的所有部分都忙碌

批处理系统

吞吐量——每小时最大作业数
周转时间——从提交到终止间的最小时间
CPU利用率——保持CPU始终忙碌

交互式系统

响应时间——快速响应请求
均衡性——满足用户的期望

实时系统

满足截止时间——避免丢失数据
可预测性——在多媒体系统中避免品质降低

图2-39 在不同环境中调度算法的一些目标

非抢占式调度算法：已经的运行的进程就让它一直运行，直到被阻塞或者释放

CPU，调度程序才运行下一个进程。

抢占式调度算法：给每个进程固定一个运行时间，如果在该时间内进程运行完毕，则接着运行下面的进程，如果没有运行完毕，则强制将该进程阻塞运行其他进程。

1. 批处理系统中的调度

1）先来先服务：非抢占式算法，先到先得的思想。简单易于理解但对于 I/O 密集型进程不友好。

2）最短作业优先：非抢占式算法，找出这若干个**已经就绪**的进程，将他们的运行时间从小到大排列，并首先运行第一个进程。这里注意的一点是必须是已经就绪的进程，若该进程还处于阻塞状态，则不能计算。

3）最短剩余时间优先：选择剩余运行时间最短的那个进程运行，如果此时有一个进程进入且其剩余时间最短，则优先运行该程序。

2. 交互式系统中的调度

1）轮转调度：指定一个时间片（进程只能在该时间段内运行），若在某进程在时间片内未完成，则将该进程挂起，运行下一个进程。一般将时间片设为 20ms~50ms.因为时间片太短，导致进程切换频繁使 CPU 效率下降；时间片太长，使后面的进程等待时间太长。

2）优先级调度：将进程按照优先级的顺序排列，CPU 将首先运行那些优先级最高的，其次是较高的，最后才是低的。在同等层次中，每个进程按照时间片轮转法进行调度。

3)多级队列：设置不同的优先级，一个进程在最高优先级运行之后，将被移入下一个等级。时间片依次可以设置为 1，2，4，8.....等等。随着进程优先级逐渐降低，进程运行的频率也会降低，为短的交互进程让出 CPU。

2. 实时系统中的调度

硬实时：对时间要求极高，容忍度为零。

软实时：可以允许错失要求的时间，有一定容忍度。

可调度的实时系统满足的条件：

$$\sum_{i=1}^{m} \frac{Ci}{Pi} \leq 1$$

m:周期事件数量，事件 i 周期为Pi，占用Ci秒 CPU 时间，以上为处理负载的条件。满足这个条件说明该系统是可调度的。