**处理机调度与死锁**

1. 高级、中级和低级调度

a.高级调度(High Scheduling)

只将外存的作业调入到内存中，也称作业调度

b.低级调度(Low Level Scheduling)

只将内存的进程获取相应的执行资源，也称进程调度

1. 非抢占方式(Non-preemptive Mode)

这种调度方式的优点是实现简单、系统开销小，适用于大多数的批处理系统环境。但它难以满足紧急任务的要求——立即执行，因而可能造成难以预料的后果。显然，在要求比较严格的实时系统中，不宜采用这种调度方式。

(2)抢占方式(Preemptive Mode)

抢占的原则有:(1)优先权原则。(2) 短作业(进程)优先原则。 (3) 时间片原则。

c.中级调度(Intermediate-Level Scheduling)

中级调度又称中程调度(Medium-Term Scheduling)。 引入中级调度的主要目的，是为了提高内存利用率和系统吞吐量。 为此，应使那些暂时不能运行的进程不再占用宝贵的内存资源，而将它们调至外存上去等待，把此时的进程状态称为就绪驻外存状态或挂起状态。当这些进程重又具备运行条件、且内存又稍有空闲时，由中级调度来决定把外存上的哪些又具备运行条件的就绪进程，重新调入内存，并修改其状态为就绪状态，挂在就绪队列上等待进程调度。

1. 调度算法

1.周转时间：指从作业被提交给系统开始，到作业完成为止的这段时间间隔（完成时间-到达时间）

2.带权周转时间：作业的周转时间T与系统为它提供服务的时间Ts之比，即W=T/Ts，

3.先来先服务调度算法（FCFS）：既适用于作业调度，也适用于进程调用,比较有利于长作业（进程），不利于短（作业）进程

4.短作业(进程)优先调度算法（SJ(P)F）:

1. 短作业优先(SJF)的调度算法:是从后备队列中选择一个或若干个估计运行时间最短的作业，将它们调入内存运行。
2. 短进程优先(SPF)调度算法:从就绪队列中选出一估计运行时间最短的进程，将处理机分配给它，使它立即执行并一直执行到完成，或发生某事件而被阻塞放弃处理机时，再重新调度。

缺点: (1)该算法对长作业不利,

(2)该算法完全未考虑作业的紧迫程度，因而不能保证紧迫性作业(进程)会被及时处理

(3)由于作业(进程)的长短只是根据用户所提供的估计执行时间而定的，而用户又可能会有意或无意地缩短其作业的估计运行时间，致使该算法不一定能真正做到短作业优先调度

5.高优先权优先调度算法:

1. 非抢占式优先权:在这种方式下，系统一旦把处理机分配给就绪队列中优先权最高的进程后，该进程便一直执行下去，直至完成； 或因发生某事件使该进程放弃处理机时，系统方可再将处理机重新分配给另一优先权最高的进程。
2. 抢占式优先权:在这种方式下，系统同样是把处理机分配给优先权最高的进程，使之执行。但在其执行期间，只要又出现了另一个其优先权更高的进程，进程调度程序就立即停止当前进程(原优先权最高的进程)的执行，重新将处理机分配给新到的优先权最高的进程。

c.优先权的类型:

1. 静态优先权:静态优先权是在创建进程时确定的，且在进程的整个运行期间保持不变
2. 动态优先权:动态优先权是指，在创建进程时所赋予的优先权，是可以随进程的推进或随其等待时间的增加而改变的，以便获得更好的调度性能。

6.高响应比优先调度算法:

引用动态优先权，使作业的优先级随等待时间的增加而增加

优先权=（等待时间+要求服务时间）/要求服务时间

响应时间=等待时间+要求服务时间

7.基于时间片的轮转调度算法

1. 时间片轮转法:系统将所有的就绪进程按先来先服务的原则，排成一个队列，每次调度时，把CPU分配给队首进程，并令其执行一个时间片。当执行的时间片用完时，由一个计时器发出时钟中断请求，调度程序便据此信号来停止该进程的执行，并将它送往就绪队列的末尾；然后，再把处理机分配给就绪队列中新的队首进程，同时也让它执行一个时间片。
2. 多级反馈队列调度算法: 应设置多个就绪队列，并为各个队列赋予不同的优先级。 第一个队列的优先级最高，第二个队列次之，其余各队列的优先权逐个降低。该算法赋予各个队列中进程执行时间片的大小也各不相同，在优先权愈高的队列中，为每个进程所规定的执行时间片就愈小。
3. 实时调度

1.实现实时调度的基本条件

a.提供必要的信息:(1)就绪时间。(2) 开始截止时间和完成截止时间。 (3) 处理时间。 (4) 资源要求。 (5) 优先级。

b.系统处理能力强

c.采用抢占式调度机制

d.具有快速切换机制

2.实时调度算法的分类

1. 非抢占式调度算法:易于实现，在一些小型实时系统或要求不太严格的实时控制系统中采用
2. 非抢占式轮转调度算法。 b.非抢占式优先调度算法。
3. 抢占式调度算法:用于要求较严格的实时系统
4. 基于时钟中断的抢占式优先权调度算法
5. 立即抢占(Immediate Preemption)的优先权调度算法

3.常用的几种实时调度算法

1. 最早截止时间优先即EDF(Earliest Deadline First)算法:根据系统任务的开始截止时间来确定任务的优先级，截止时间越短，优先级越高，可用于抢占式与非抢占式
2. 最低松弛度优先即LLF(Least Laxity First)算法：该算法是根据任务紧急(或松弛)的程度，来确定任务的优先级。任务的紧急程度愈高，为该任务所赋予的优先级就愈高， 以使之优先执行。一任务在400 ms时必须完成，它本身需要运行 150 ms，则其松弛程度为 250 ms。
3. 产生死锁的原因和必要条件
4. 产生死锁的原因：

(1)竞争资源。

(2)进程间推进顺序非法。

2.产生死锁的必要条件

(1)互斥条件

(2)请求和保持条件

(3)不剥夺条件

(4)环路等待条件

3.处理死锁的基本方法

(1)预防死锁。在运行前检测

(2)避免死锁。在运行前检测

(3)检测死锁。在运行时检测，不做任何处理

(4)解除死锁。在运行时检测，并做相应的处理