**进程调度算法**

**基本调度算法**：

1. **先来先服务FCFS**：既可以作为作业调度算法也可以作为进程调度算法；按作业或者**进程到达的先后顺序依次调度**；因此对于长作业比较有利。

**算法优点**：易于理解且实现简单，只需要一个队列(FIFO)，且相当公平   
**算法缺点**：比较有利于长进程，而不利于短进程，有利于CPU 繁忙的进程，而不利于I/O 繁忙的进程

1. **短作业优先SPF**：作业调度算法，**算法从就绪队列中选择估计时间最短的作业进行处理，**直到得出结果或者无法继续执行；

**算法优点**：相比FCFS 算法，该算法可改善平均周转时间和平均带权周转时间，缩短进程的等待时间，提高系统的吞吐量。   
**算法缺点**：对长进程非常不利，可能长时间得不到执行，且未能依据进程的紧迫程度来划分执行的优先级，以及难以准确估计进程的执行时间，从而影响调度性能。

1. \*\* 高优先权优先HRRF\*\*：既可以作为作业调度也可以作为进程调度算法；调度作业时，从**就绪队列中选择优先级最高的作业进行处理**；由于涉及到了优先级，因此可以分为抢占式和非抢占式；而且优先级的确定也可以分为静态优先级（事先根据进程类型，进程对资源的需求，用户要求等方面确定一个固定值）；动态优先级（随进程的推进或者等待时间而增加或者减少）。
2. **最高响应比优先HRN**：FCFS可能造成短作业用户不满，SPF可能使得长作业用户不满，于是提出HRN，**选择响应比最高的作业运行。（考虑等待和处理时间因素）**响应比=1+作业等待时间/作业处理时间。

**算法优点**：由于长作业也有机会投入运行，在同一时间内处理的作业数显然要少于SJF法，从而采用HRRN方式时其吞吐量将小于采用SJF 法时的吞吐量。   
**算法缺点**：由于每次调度前要计算响应比，系统开销也要相应增加。

1. 时间片轮转**（RR，Round-Robin）** ：按到达的先后对进程放入队列中，然后给队首进程分配CPU时间片，时间片用完之后计时器发出中断，暂停当前进程并将其放到队列尾部，循环。

**算法优点**：时间片轮转调度算法的特点是简单易行、平均响应时间短。   
**算法缺点**：不利于处理紧急作业。在时间片轮转算法中，时间片的大小对系统性能的影响很大，因此时间片的大小应选择恰当   
怎样确定时间片的大小：

**时间片大小的确定**   
1.系统对响应时间的要求   
2.就绪队列中进程的数目   
3.系统的处理能力

1. **多级反馈队列：**目前公认较好的调度算法；设置多个就绪队列并为每个队列设置不同的优先级，第一个队列优先级最高，其余依次递减**。优先级越高的队列分配的时间片越短，进程到达之后按FCFS放入第一个队列，**如果调度执行后没有完成，那么放到第二个队列尾部等待调度，如果第二次调度仍然没有完成，放入第三队列尾部…。只有当前一个队列为空的时候才会去调度下一个队列的进程。

**在多级反馈队列调度算法中，如果规定第一个队列的时间片略大于多数人机交互所需之处理时间时，便能够较好的满足各种类型用户的需要。**