处理机调度经典算法比较

（1）作业调度

1、先来先服务（FCFS）

基本思想：按进程（作业）进入就绪（后备）队列的先后次序来分配处理机（为其创建进程）。

一般采用非剥夺的调度方式。

例子：

作业号 提交时间 运行时间 开始时间 等待时间 完成时间 周转时间 带权周转时间

1 8 2 8 0 10 2 1

2 8.4 1 10 1.6 11 2.6 2.6

3 8.8 0.5 11 2.2 11.5 2.7 5.4

4 9 0.2 11.5 2.5 11.7 2.7 13.5

平均等待时间 t = (0+1.6+2.2+2.5)/4=1.575

平均周转时间 T = (2+2.6+2.7+2.7)/4=2.5

平均带权周转时间 W = (1+2.6+5.4+13.5)/4=5.625

FCFS 调度算法的平均作业周转时间与作业提交的顺序有关。

FCFS 调度算法的特点：

简单，但效率不高。

有利于 CPU 繁忙型作业。

不利于 I/O 繁忙型作业。

现在操作系统中，已很少用该算法作为主要调度策略，尤其是在分时系统和实时系统中。但它常被结合在其它调度策略中使用。

2、短作业优先（SJF）

用于作业调度

主要任务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。类似地，用于进程调度的是短进程优先调

优点

能有效降低作业的平均等待时间。

能有效缩短进程的平均周转时间。

提高了吞吐量。

缺点

对长作业不利。

没有考虑作业的紧迫程度。

作业执行时间、剩余时间仅为估计。

SJF 算法虽然是优化的，但在 CPU 调度中很难实现。

3优先级调度算法（PSA）

优先级调度算法(Priority-Scheduling Algorithm, PSA) 以作业的紧迫程度为优先级。

系统选择优先级最高的几个作业装入内存。

优先级调度算法也用于进程调度，系统在可运行的进程中选择优先级最高者使其投入运行。

优先级的类型：

静态优先级

动态优先级

静态优先级

　　优先权在创建进程时确定，且在进程的整个运行期间保持不变。一般用整数表示，小表示优先级高。

　　确定原则：

　　进程类型（系统进程 > 用户进程）

　　进程对资源的需求（要求少的有较高的优先权）

　　用户要求（紧急程度和付费情况）

　　优点：简单，开销小。

　　缺点：公平性差（对低优先权进程）

动态优先级

　　动态优先级在进程的存在过程中不断发生变化。

　　动态优先级的变化取决于：进程的等待时间、进程的运行时间、进程使用资源的情况

　　动态优先权确定方法的资源利用率高，公平性好，但开销较大，实现较为复杂。

　　高响应比优先算法 (HRRN)采用动态优先权。

4高响应比优先调度算法（HRRN）

FCFS 只考虑了作业的等待时间，忽略了运行时间。SJF 只考虑了作业的运行时间，忽略了等待时间。

高响应比优先调度算法(Highest Response Ratio Next, HRRN) 既考虑了作业的等待时间，也考虑了作业的运行时间，是一种动态优先级调度算法。

优先权 =（等待时间 + 要求服务时间）/要求服务时间

响应比RP =(等待时间 + 要求服务时间)/要求服务时间 =响应时间/要求服务时间

周转时间：从提交到完成的时间间隔。

响应时间：在交互式系统中，从提交请求到产生首次响应的时间，而不是到产生输出结果所需的时间

根据优先权公式可知

如等待时间相同，则要求服务时间越短其优先权越高 →SJF。

如要求服务时间相同，优先权决定于等待时间 →FCFS。

对长作业，若等待时间足够长，优先权也高，也能获得 CPU

（2）进程调度

1、最短剩余时间调度算法（SRT）

最短剩余时间调度算法(Shortest Remaining Time First, SRT) 是针对SPF 增加了抢占机制的一种调度算法。它总是选择预期剩余时间最短的进程。只要新进程就绪，且有更短的剩余时间，调度程序就可能抢占当前正在运行的进程。

SRT 不像 FCFS 偏向长进程，也不像轮转法（下个算法）产生额外的中断，从而减少了开销。

就此例而言，平均周转时间和带权平均周转时间说明了SRT 好于前面的任何一个算法（因为它具有抢占的特点）。

缺点：必须记录过去的服务时间，从而增加了开销

2、时间片轮转调度算法（RR）

时间片轮转调度算法：系统将所有就绪进程按FCFS的原则，排成一个队列依次调度，把 CPU 分配给队首进程，并令其执行一个时间片，通常为10-100ms。时间片用完后，系统的计时器发出时钟中断，该进程将被剥夺 CPU并插入就绪队列末尾。RR 是一种非常公平的算法。

进程切换时机

若一个时间片尚未用完进程便已经完成，就立即再调度就绪队列中队首进程运行，并启动一个新的时间片。如果在一个时间片用完时进程尚未运行完毕，则剥夺 CPU，调度程序把它

送往就绪队列的末尾。

响应时间T = 时间片q × 就绪队列进程数n

3、多级队列调度算法

优先级调度算法的类型

非抢占式优先级调度算法：一旦把处理机分配给就绪队列中优先级最高的进程后，该进程便一直执行下去直至完成，或者因该进程发生某事件而放弃处理机时，系统方可将处理机重新分配给另一优先级最高的进程。

抢占式优先级调度算法：只要出现了另一个其优先级更高的进程，调度程序就将处理机分配给新到的优先级最高的进程

4、多级反馈队列调度算法（MFQ）

多级反馈队列调度算法是时间片轮转算法和优先级调度算法的综合和发展，通过动态调整进程优先级和时间片大小，不必事先估计进程的执行时间。

FCFS + 优先级 + RR + 抢占

多级反馈队列可兼顾多方面的系统目标，是目前公认的一种较好的进程调度算法

（3）实时调度

1、非抢占式调度算法用于非周期实时任务

2、抢占式调度算法用于周期实时任务

3、最低松弛度优先算法（LLF）

最低松弛度优先算法（LLF，Least Laxity First）

低松弛 = 高紧急

算法是根据任务紧急（或松弛）的程度，来确定任务的优先级。任务的紧急度越高，其优先级越高，并使之优先执行。

算法采用抢占调度方式，可用于调度具有完成截止时间的周期性实时任务。

松弛度 = 必须完成时间 - 本身剩余运行时间 - 当前时间