3.2.3 先来先服务和短作业优先调度算法

【作业调度举例】

例1：在一个单道批处理系统中，有一组作业的提交时刻和运行时间如右表所示

作业名	A	B	C	D
提交时刻	8.0	8.5	9.0	9.1
运行时间	1.0	0.5	0.2	0.1

当系统采用作业先来先服务的调度算法时，计算该组作业的平均周转时间和平均带权周转时间是多少。

作业FCFS调度顺序	A	B	C	D	平均
作业开始执行时刻	8.0	9.0	9.5	9.7	
作业完成时刻	9.0	9.5	9.7	9.8	
周转时间	1.0	1.0	0.7	0.7	0.85
带权周转时间	1.0	2.0	3.5	7.0	3.375

- 平均周转时间T=（1.0+1.0+0.7+0.7）/4=0.85
- 平均带权周转时间W=(1.0+2.0+3.5+7.0)/4=3.375

3.2.3 先来先服务和短作业优先调度算法

例2：在一个单道批处理系统中，有一组作业的提交时刻和运行时间如右表所示

作业名	A	B	C	D
提交时刻	8.0	8.5	9.0	9.1
运行时间	1.0	0.5	0.2	0.1

当系统采用短作业优先的调度算法时，计算该组作业的平均周转时间和平均带权周转时间是多少。

作业SJF调度顺序	A	C	D	B	平均
作业开始执行时刻	8.0	9.0	9.2	9.3	
作业完成时刻	9.0	9.2	9.3	9.8	
周转时间	1.0	0.2	0.2	1.3	0.675
带权周转时间	1.0	1.0	2.0	2.6	1.65

平均周转时间T＝（1.0+0.2+0.2+1.3）/4=0.675

平均带权周转时间W=(1.0+1.0+2.0+2.6)/4=1.65

3.2.3 先来先服务和短作业优先调度算法

SJ(P)F调度算法也存在不容忽视的缺点：

(1) 该算法对长作业不利，如作业C的周转时间由10增至16，其带权周转时间由2增至3.1。更严重的是，如果有一长作业(进程)进入系统的后备队列(就绪队列)，由于调度程序总是优先调度那些(即使是后进来的)短作业(进程)，将导致长作业(进程)长期不被调度。

(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。

2. 高响应比优先调度算法（Highest Response ratio Next, HRRN）

优先权的变化规律可描述为：

$$\text{优先权} = \frac{\text{等待时间} + \text{要求服务时间}}{\text{要求服务时间}}$$

由于等待时间与服务时间之和，就是系统对该作业的响应时间，故该优先权又相当于响应比R_p。

据此，又可表示为：

$$\text{优先权} = \frac{\text{等待时间} + \text{要求服务时间}}{\text{要求服务时间}} = \frac{\text{响应时间}}{\text{要求服务时间}}$$

3.2.4 优先级调度算法和高响应比优先调度算法

由公式可知：

(1) 如果作业的等待时间相同，则要求服务的时间愈短，其优先权愈高，因而该算法有利于短作业。

(2) 当要求服务的时间相同时，作业的优先权决定于其等待时间，等待时间愈长，其优先权愈高，因而它实现的是先来先服务。

(3) 对于长作业，作业的优先级可以随等待时间的增加而提高，当其等待时间足够长时，其优先级便可升到很高， 从而也可获得处理机。

3.2.4 优先级调度算法和高响应比优先调度算法

9.5时，计算响应比

9.0时，计算响应比

例3：在一个单道批处理系统中，有4个作业A、B、C、D，其到达时刻和运行时间如右表所示。

作业名	A	B	C	D
提交时刻	8.0	8.5	9.0	9.1
运行时间	1.0	0.5	0.2	0.1

C的响应比=1+0/0.2=1

D的响应比

当系统采用作业响应比高优先的调度算法时，计算该组作业的平均周转时间和平均带权周转时间是多少。

作业HRF调度顺序	A	B	D	C	平均
作业开始执行时刻	8.0	9.0	9.5	9.6	
作业完成时刻	9.0	9.5	9.6	9.8	
周转时间	1.0	1.0	0.5	0.8	0.825
带权周转时间	1.0	2.0	5.0	4.0	3

平均周转时间T=（1.0+1.0+0.5+0.8）/4=0.825

平均带权周转时间W=(1.0+2.0+5.0+4.0)/4=3

3.3.1 进程调度的任务、机制和方式

1、进程调度的任务

- 保存现场
- 按某种算法选取进程
- 将处理器分配给进程

2. 进程调度机制

- 排队器：将就绪进程插入到相应的就绪队列
- 分派器：从就绪队列挑选进程运行
- 上下文切换：当前进程->分派器->新选进程

（为节省切换时间，可采用2组或多组寄存器）

2) 抢占方式(Preemptive Mode)

抢占的原则有：

(1) 优先权原则。

(2) 短进程优先原则。

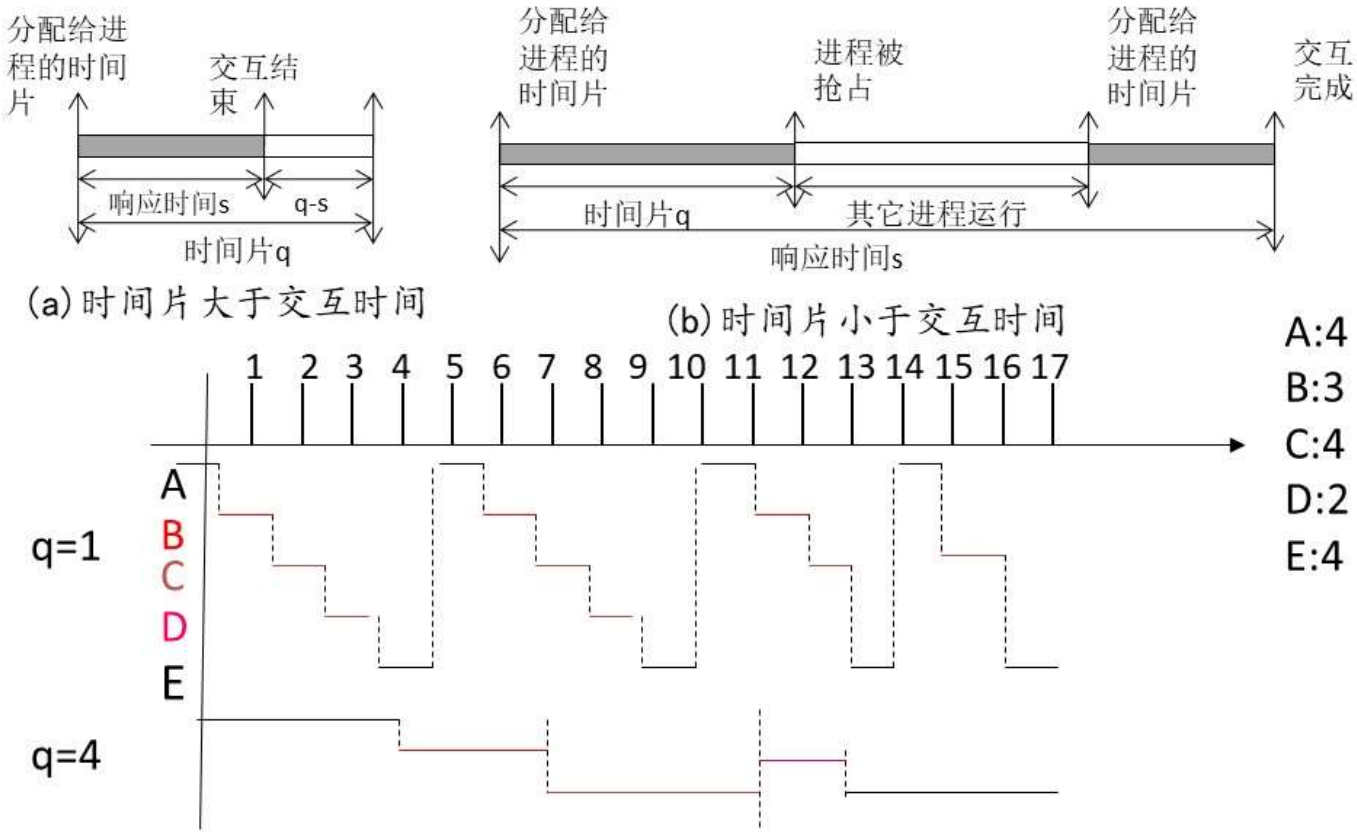
(3) 时间片原则。

3.3.2 轮转调度算法

1. 轮转法的基本原理

在早期的时间片轮转法中，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。时间片的大小从几ms到几百ms。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。这样就可以保证就绪队列中的所有进程，在一给定的时间内，均能获得一时间片的处理机执行时间。

3.3.2 轮转调度算法


分配给进程的时间片
交互结束
响应时间s
q-s
时间片q
(a) 时间片大于交互时间
分配给进程的时间片
进程被抢占
分配给进程的时间片
交互完成
时间片q
其它进程运行
响应时间s
(b) 时间片小于交互时间
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
A
B
C
D
E
q=1
q=4
A:4
B:3
C:4
D:2
E:4

3.3.2 轮转调度算法

作业情况 / 时间片	进程名	A	B	C	D	E	平均
	到达时间	0	1	2	3	4	
	服务时间	4	3	4	2	4	
RR q=1	完成时间	15	12	16	9	17	
	周转时间	15	11	14	6	13	11.8
	带权周转时间	3.75	3.67	3.5	3	3.33	3.46
RR q=4	完成时间	4	7	11	13	17	
	周转时间	4	6	9	10	13	8.4
	带权周转时间	1	2	2.25	5	3.33	2.5

1. 优先级调度算法的类型

1）非抢占式优先权算法

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）抢占式优先权调度

在采用这种调度算法时，是每当系统中出现一个新的就绪进程i时，就将其优先权P_i与正在执行的进程j的优先权P_j进行比较。如果$P_i \leqslant P_j$，原进程P_j便继续执行；但如果是$P_i > P_j$，则立即停止P_j的执行，做进程切换，使i进程投入执行。显然，这种抢占式的优先权调度算法，能更好地满足紧迫作业的要求，故而常用于要求比较严格的实时系统中。

3.3.4 多队列调度算法

一个就绪队列无法满足不同用户对进程调度策略的不同要求，多处理机系统该缺点更加突出。

在采用这种调度算法时，将不同类型或性质的进程固定分配在不同的就绪队列，分别采用不同的调度算法。

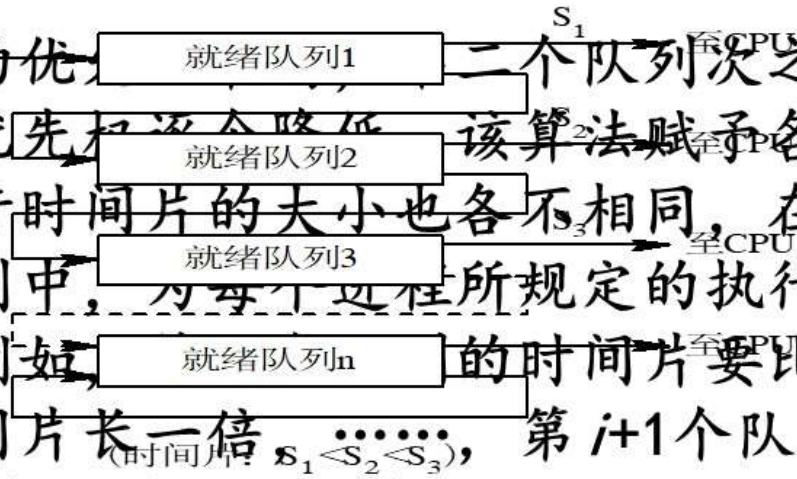
在多处理机系统中，可以为每个CPU设置一个就绪队列，每个处理机可以采用不同的调度算法。

3.3.5 多级反馈队列

1. 调度机制

(1) 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[illegible]二个队列次之，其余各队列的优先[illegible]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。例如，[illegible]的时间片要比第一个队列的时间片长一倍，……，第 $i+1$ 个队列的时间片要比第 i 个队列的时间片长一倍。



(时间片: $S_1 < S_2 < S_3$)

图 3-4 是多级反馈队列算法的示意。

3.3.5 多级反馈队列

1. 调度机制

(1) 应设置多个就绪队列，并为各个队列赋予不同的优先级。

(2) 每个队列都采用FCFS算法

当一个新进程进入内存后，首先将它放入第一队列的末尾，按FCFS原则排队等待调度。当轮到该进程执行时，如它能在该时间片内完成，便可准备撤离系统；如果它在一个时间片结束时尚未完成，调度程序便将该进程转入第二队列的末尾，再同样地按FCFS原则等待调度执行；如果它在第二队列中运行一个时间片后仍未完成，再依次将它放入第三队列，……，如此下去，当一个长作业(进程)从第一队列依次降到第n队列后，在第n队列中便采取按时间片轮转的方式运行。

3.3.5 多级反馈队列

1. 调度机制

(1) 应设置多个就绪队列，并为各个队列赋予不同的优先级。

(2) 每个队列都采用FCFS算法

(3) 按队列优先级调度

仅当第一队列空闲时，调度程序才调度第二队列中的进程运行； 仅当第1~(i-1) 队列均空时，才会调度第i队列中的进程运行。如果处理机正在第i队列中为某进程服务时，又有新进程进入优先权较高的队列(第1~(i-1)中的任何一个队列)，则此时新进程将抢占正在运行进程的处理机，即由调度程序把正在运行的进程放回到第i队列的末尾，把处理机分配给新到的高优先权进程。

2. 调度算法的性能

(1) 终端型作业用户。

(2) 短批处理作业用户。

(3) 长批处理作业用户。

3.4.1 实现实时调度的基本条件

1. 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。

1. 可重用性资源和消耗性资源

1) 可重用性资源　　性质如下：

(1) 每一个可重用性资源中的单元只能分配给一个进程使用；

(2) 遵循可重用资源的使用顺序：请求、使用、释放；

(3) 系统中每一类可重用性资源中的单元数目是相对固定的，系统中的大多数资源都是可重用的。