**OS第五次作业： 直接发助教邮箱：wth\_os@163.com**

5.1 Why is it important for the scheduler to distinguish I/O-bound programs from CPU-bound programs?

因为不同类型的进程对I/O和CPU的需求不同，I/O-bound进程大部分时间再进行I/O，而CPU-bound的进程大部分的时间使用CPU。一个良好的 I/O-bound和 CPU-bound进程的比例能够提升系统的效率。如果I/O-bound进程的数目过多，会导致进程等待I/O的时间变长而CPU出于闲置状态，之会导致等待CPU的时间过长而I/O处于闲置状态，出于这个原因scheduler必须将 IO-bound 和 CPU-bound的进程分开。

5.2 Discuss how the following pairs of scheduling criteria conflict in certain settings.

a. CPU utilization and response time

b. Average turnaround time and maximum waiting time

c. I/O device utilization and CPU utilization

(a) 如果要求CPU执行效率，就意味着一个任务在不必要的时候(必要的时候指必须进行IO操作)能够不被打扰，以减少进程切换的开销，但是 响应时间短意味着系统要及时地响应用户的操作，就意味着CPU要经常中断当前正在执行的进程转到UI进程去响应用户的操作。导致增大进程切换的开销。所以CPU utilization 和 reponse time 是矛盾的。（继承切换需要保存当前的 contex 和读取待执行进程的 contex 这段之间CPU没有执行任务，产生额外开销。）

（b）平均周转时间指的是进程被提交到进程结束的时间，周转时间等于总的等待时间+总的执行时间。最小化平均周转时间就等价于需要最小化等待时间，因此需要用”短作业优先调度”。但是使用短作业优先调度可能产生饥锇现象，导致长作业得不到执行，这时候的 maximum waiting time为长作业的等待时间，将会变得很长。产生了矛盾。

(c) 要提升IO设备的利用率，需要CPU及时地处理IO中断，极端情况下采用查询的方式进行IO, 此时IO设备的利用率极高，但是会导致CPU时间浪费在等待IO操作上，无法执行任务这会导致CPU利用率下降。这就带来了矛盾。

5.4 Consider the following set of processes,with the length of the CPU-burst time given in milliseconds:

Process Burst Time Priority

P1 10 3

P2 1 1

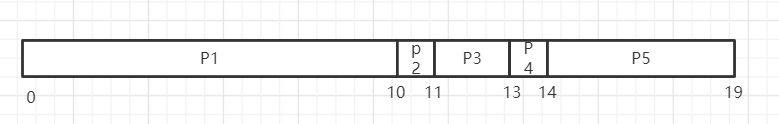
P3 2 3

P4 1 4

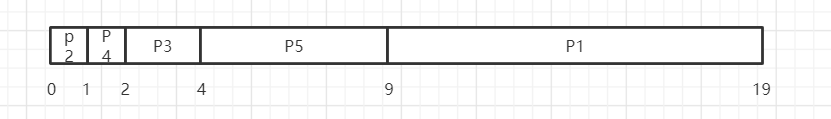
P5 5 2

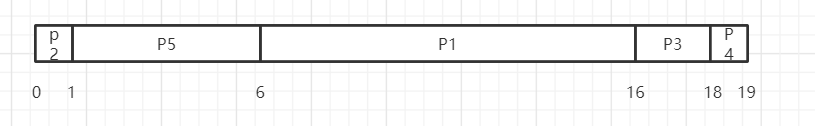
The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

a. Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a nonpreemptive priority (a smaller priority number implies a higher priority), and RR (quantum = 1) scheduling.

FCFS : 

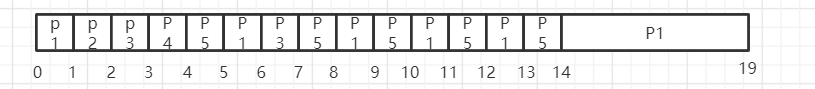
SJF:



Priority: 

(P1排在P3前面，因为P1先到)

时间片轮转：



b. What is the turnaround time of each process for each of the scheduling algorithms in part a?

周转时间：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | P1 | P2 | P3 | P4 | P5 |
| FAFS | 10 | 11 | 13 | 14 | 19 |
| SJF | 19 | 1 | 4 | 2 | 9 |
| Priority | 16 | 1 | 18 | 19 | 6 |
| RR | 19 | 2 | 7 | 4 | 14 |

c. What is the waiting time of each process for each of the scheduling algorithms in part a?

等待时间：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | P1 | P2 | P3 | P4 | P5 |
| FAFS | 0 | 10 | 11 | 13 | 14 |
| SJF | 9 | 0 | 2 | 1 | 4 |
| Priority | 6 | 0 | 16 | 18 | 1 |
| RR | 0+4+2+1+1+1=9 | 1 | 2+3=5 | 3 | 4+2+1+1+1 =9 |

d. Which of the schedules in part a results in the minimal average waiting time (over all processes)?

平均等待时间:

FAFS : (10+ 11+13+14)/5 = 9.6

SJF : (9+2+1+4)/5 = 3.2

Priority : (6+16+18+1)/5 = 8.2

RR : (9+1+5+3+9)/5 = 5.4

所以是 SJF 的平均等待时间最短

5.5 Which of the following scheduling algorithms could result in starvation?

a. First-come, first-served

b. Shortest job first

c. Round robin

（b）短作业优先可能导致进程饿死，因为执行时间长的作业必须等待执行时间短的作业完成，才能轮到它。假如执行时间短的进程持续不断地被提交，那么执行时间长的进程可能永远得不到执行，就会被”饿死”

5.10 Explain the differences in the degree to which the following scheduling algorithms discriminate in favor of short processes:

a. FCFS b. RR c. Multilevel feedback queues

对短作业支持程度(从高到低) c > b >a

a 先来先服务调度算法，它不区分短作业和长作业，根据到来的时间进行服务，如果执行时间长的进程先到达，会发生护航效应，导致执行时间短的进程得不到执行，对短作业的支持最差。

b 时间片轮转。给每个进程分配一个时间片，短作业执行时间短能够在较少的时间片内完成，同时一旦时间片耗尽，长作业也会被强制切换，使得其它短作业得到执行。但是在执行顺序上短作业和长作业取决于到来的顺序。这种方案对短作业的支持程度高于a 但是低于c。

c 多级反馈队列。这种方式采用多级队列，每个队列分配的时间片不同，每个队列都有优先级，只有优先级高的队列里的进程执行完了才会执行优先级低的队列的进程。和b一样，一旦时间片耗尽，进程就会被切换，这确保了长作业之后的短作业能够得到执行。但是和 b 不同的是，如果时间片结束还没有执行完，进程就会被放置到更低优先级的队列中，这意味着随着时间片的轮转，长作业的优先级会越来越低，短作业能够越来越早地得到执行(因为他们的队列的优先级较高)。所以其对短作业的支持程度最高。