CS 390
Chapter 5 Homework Solutions

5.1 `A CPU-scheduling algorithm determines ...`

Determining an execution schedule for n processes is the same problem as placing n items in an array according to some placement algorithm. The first element could be filled by any of the n processes. The second element could be filled by any of the remaining $n-1$ processes. The third element by any of the remaining $n-2$ processes, and so on. Each possible placement of processes in the list corresponds to a different schedule, so there are $n!$ schedules possible.

5.2 `Explain the difference between preemptive ...`

There are four types of events that cause a process to relinquish control of the CPU:

1. The process makes a system call.
2. The arrival of interrupt from the timer indicating that the process's time slice has expired.
3. A device interrupt arrives indicating that the device needs attention.
4. The process terminates.

In preemptive scheduling, the CPU scheduler runs after each of these events. In non-preemptive scheduling, the CPU scheduler does not run after a timer or device interrupt. Instead, the kernel services the interrupt and returns CPU control to the same process.

5.3 `Suppose that the following processes ...`

a. $10.5\overline{3}$ ms[1]

b. $9.5\overline{3}$ ms

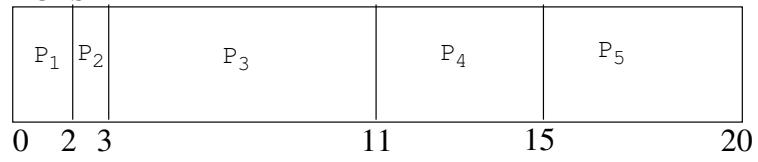
c. $6.8\overline{6}$ ms

[1]Notice that our author committed a huge faux-pas by not stating the units of the problem. I am assuming milliseconds. That just goes to show you that even Yale professors make mistakes.

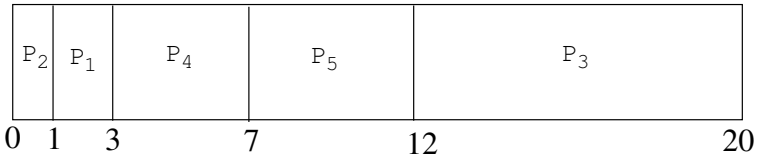
5.4 Consider the following set of ...

a. Draw four Gantt charts that ...

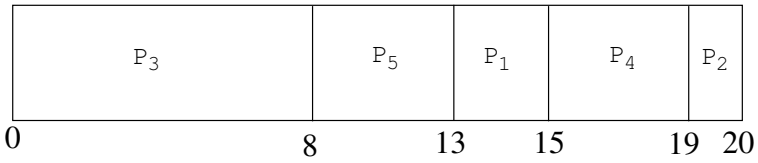
FCFS



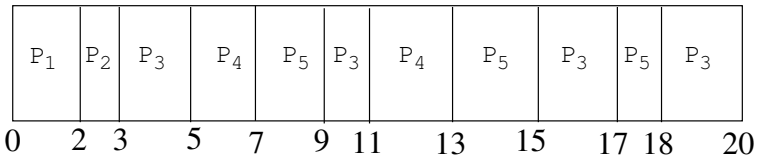
SJF



Priority



RR (Q = 2)



b. What is the turnaround time ...

Turnaround Time				
	Scheduling Algorithm			
Process	FCFS	SJF	Priority	RR
1	2	3	15	2
2	3	1	20	3
3	11	20	8	20
4	15	7	19	13
5	20	12	13	18

c. What is the waiting time ...

Waiting Time				
	Scheduling Algorithm			
Process	FCFS	SJF	Priority	RR (Q = 2)
1	0	1	13	0
2	2	0	19	2
3	3	12	0	12
4	11	3	15	9
5	15	7	8	13

d. `Which of the algorithms in ...`

- FCFS: 6.2 msecs average waiting time
- SJF: 4.6 msecs average waiting time
- Priority: 11 msecs average waiting time
- RR: 7.2 msecs average waiting time

Thus, SJF results in the minimum average waiting time, as would be expected since SJF is provably optimal when we wish to minimize waiting time and turnaround time.

Additional Exercises

A A scheduling algorithm wants to make a prediction of the CPU burst times of a process. Fill in the following table showing the estimated burst time at each time, using the formula for exponential average with $\alpha = 0.5$. Use 1 ms for your initial guess τ_0. In each case, calculate the error in your guess - that is, how different the estimate is from the actual observed value $(\tau_n - t_n)$.

Time n	τ_n	**Observed CPU Burst** t_n	**Error**
0		2	
1		4	
2		8	
3		1	
4		1	
5		-	-

B Why is it important for the CPU scheduler to distinguish IO-bound programs from CPU-bound programs?

C Consider the exponential average formula used to predict the length of the next CPU burst. What would be the implications of setting $\alpha = 0$ and τ_0 to 100 ms? What would be the implications of setting $\alpha = 0.999$ and τ_0 to 10 ms?

D Which of the following scheduling algorithms can result in starvation? FCFS, SRTF, Round-Robin, Priority.

Answers to Additional Exercises

A

Time n	τ_n	**Observed CPU Burst** t_n	**Error**
0	1	2	-1
1	1.5	4	-2.5
2	2.75	8	-5.25
3	5.375	1	4.375
4	3.1875	1	2.1875
5	2.09375	-	-

B Many scheduling algorithms require the scheduler to make an estimate of the length of a process's next CPU burst. At a minimum, this estimate should take into account the past behavior of the process. A crude estimate of this past behavior can be obtained if we can distinguish between I/O-bound processes, which tend to have short CPU-bursts, and CPU-bound processes, which tend to have longer CPU-bursts.

In addition, if the scheduler favors I/O-bound processes over CPU-bound processes, interactive programs will show better response times. We can use the time while the user of the interactive program is typing to schedule CPU-bound processes. You may be a fast typist, but the kernel can schedule and execute hundreds of processes between the time it takes you to type two characters on the keyboard.

C Setting the history parameter α to 0 means that the most recent CPU-burst length will not effect our estimate of any future CPU-burst length. Thus, future estimates will depend solely on our past estimate. Since our initial estimate was 100 msecs, each of the future estimates will also be 100 msecs.

Setting the history parameter to 0.999 means that our estimate of the next CPU-burst will be weighted very heavily by the length of the

previous CPU burst. In CPU-bound processes, this results in accurate estimates that quickly converge to the correct value. However, even CPU-bound processes have the occasional short IO-burst, and when this happens the estimate for the next burst after IO is done will be poor.

D FCFS and Round-Robin cannot result in starvation. SRTF and Priority can result in starvation.