OS Solutions to the January 2001 Core Exam

OS1. Consider a system with two kinds of disks: big-sector and small-sector. For every access, each disk requires 6ms for seeking and 4ms for rotational latency and then transfers one sector at 10MB/sec. A big-sector disk has 128KB sectors. A small-sector disk has 1KB sectors. Assume all sectors are scattered randomly across the disk (even if they represent consecutive blocks in a file). If the goal is to transfer 1KB bytes, clearly the small sector disk is faster (transferring the rest of the large sector is wasted). If the goal is to transfer 128KB, clearly the large sector disk is faster. Where is the crossover point? That is, what is the **SMALLEST** request for which the large sector disk is faster (assume all requests must be a multiple of 1KB). Show your work. To ease the arithmetic, assume that 1MB = 10^6 bytes and 1KB = 10^3 bytes. Recall that 1ms = 1 millisecond = 10^{-3} seconds.

Solution. Time for a big-sector transfer is

$$6\text{ms} + 4\text{ms} + \frac{128\text{KB}}{10\text{MB/sec}} =$$

$$10\text{ms} + \frac{128 \times 10^3\text{B}}{10^7\text{B/sec}} =$$

$$10^{-2}\text{sec} + 128 \times 10^{-4}\text{sec} = 22.8\text{ms}$$

Time for a small-sector transfer is

$$6\text{ms} + 4\text{ms} + \frac{1\text{KB}}{10\text{MB/sec}} =$$

$$10\text{ms} + \frac{10^3\text{B}}{10^7\text{B/sec}} =$$

$$10^{-2}\text{sec} + 10^{-4}\text{sec} = 10.1\text{ms}$$

Thus 2 small transfers take 20.2ms and three take 30.3ms. Hence a request for 3KB (three small transfers) is the smallest size for which the large-sector disk is faster.

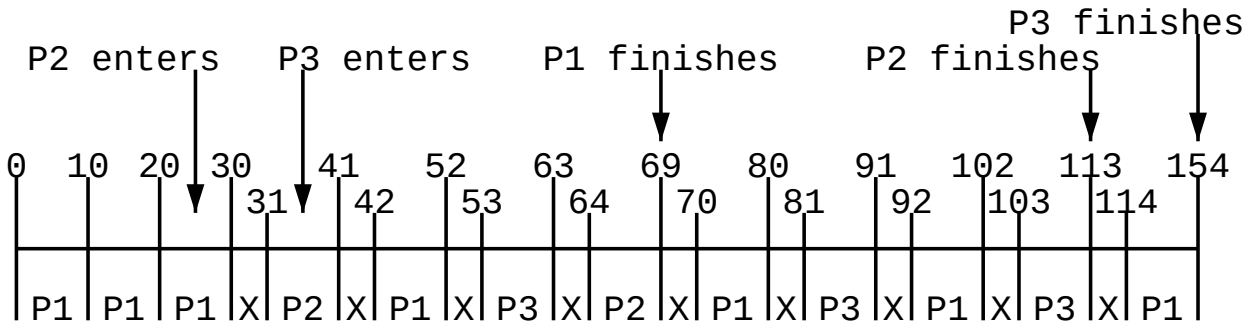
OS2. Consider the following set of processes, each of which performs no I/O (i.e., no process ever blocks). All times are in milliseconds. The CPU time is the total time required for the process. The creation time is the time when the process is created. So P1 is created when the problem begins and P3 is created 35 milliseconds later.

Process	CPU Time	Creation Time
P1	100	0
P2	15	25
P3	30	35

Assume that the round robin quantum is 10ms and that the time required to context switch from one process to a DIFFERENT process is 1ms. If, at the end of a quantum, the system runs the SAME process again, assume that no time is required by the system to make the decision and resume the process.

At what time does each process finish and what is the efficiency achieved? Show your work. Redo the entire problem with a quantum of 100ms.

Solution.



The diagram above is nearly self explanatory: The bottom column indicates the process that is running, with X signifying a 1ms context switches.

Note that when P3 enters at 35 it is placed on the rear of the ready queue behind P1, which was previously placed on the ready queue during the 30-31 context switch. So P1 runs at 42.

P1 finishes at 154.
P2 finishes at 69.
P3 finishes at 113.

The system computes user jobs for 100+15+30=145ms.
The system context switches for 9 ms.
The efficiency is 145ms/(145+9)ms = .94.

The second part is much easier.

P1 starts at zero and finishes at 100.
P2 starts at 101 and finishes at 116.
P3 starts at 117 and finishes at 147.

The efficiency is 145ms/(145+2)ms = .99.