

CSC 262

Homework 1 *Solutions*

1. This problem deals with job scheduling. Assume that there are 4 jobs:

Job 1: 25 sec. CPU

Job 2: 10 sec. CPU, 5 sec. I/O, 5 sec. CPU, 10 sec. I/O

Job 3: 5 sec. CPU, 2 sec. I/O, 10 sec. CPU, 2 sec. I/O

Job 4: 2 sec. CPU, 3 sec. I/O, repeated three more times

- **Machine A:** Create a schedule for a serial batch system. That is, each job, once started must run to completion before another job can be scheduled. (Be sure to include the total time)

All the schedules for this machine result in the same running time

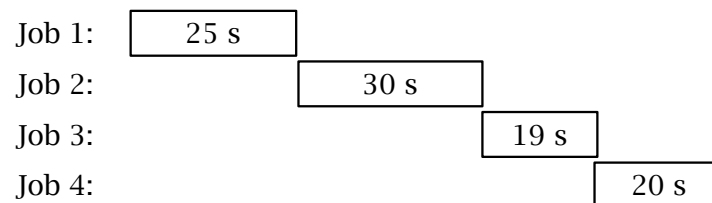


Figure 1: Total Time: 94s

- **Machine B:** Create a schedule for a simple multiprogrammed system that can have one job utilizing the CPU and one utilizing I/O devices at a time. Assume that the OS has 5 sec time slices (this is highly unrealistic). Also assume round-robin scheduling, that is, the next job scheduled is the one that ran least recently (for 3 jobs all using cpu this would mean a schedule of 1,2,3,1,2,3,etc. on the CPU).(Be sure to include the total time)

There are many possible valid schedules for this simplistic machine. This schedule results if you start with Job 1 and assume that the round robin queue was {1,2,3,4} when job 1 was scheduled.

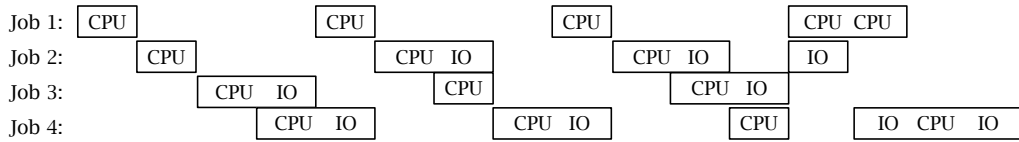


Figure 2: Total Time: 80s

2. Rather than explicitly chop up processes it can be helpful to think of them as a pair of values. This pair consists of total time and the percentage of that time spent using the CPU. For instance, Job 1 would be (25 sec, 100%).

(a) What is the pair for Job 2? (30s, 50%)

(b) Job 3? (19s, 79%)

(c) Job 4? (20s, 40%)

3. Using the pairs calculated above, it is possible to estimate the best possible running time for a collection of jobs. Assuming a machine that can only run one process at a time, but that has an OS with a perfect scheduler and no overhead (**highly** unrealistic), what is the best possible completion time for running all 4 jobs? *Hint: remember that the OS can run another process while one is waiting for I/O to complete. Simplify the jobs into a solid compute chunk followed by an I/O chunk and think about running the jobs back to back (overlapping compute and I/O)*

This is a rough way to estimate the best possible running time. Basically, you simplify your view of the process into two parts, 'work' and 'blocking for I/O'. Then you assume that any other process can overlap its work parts with any other processes blocking parts. In this simplified model, all the CPU times added together are 63 seconds. All the I/O times added together are 31 seconds. In this case all the I/O can be 'hidden' by the CPU time. A slightly more rigorous approach would actually sequence the processes, but since jobs 2-4 all end with I/O you can estimate that the I/O 'tail' would be either 2,3, or 10 seconds. Allowing a result between 65 and 73 seconds.

4. *Throughput* is a measure of the amount of jobs a system can complete, and *utilization* is a measure of how efficiently system resources are being used.

$$Utilization = \frac{\text{time used}}{\text{total time elapsed}}$$

$$Throughput = \frac{\text{jobs completed}}{\text{total time elapsed}}$$

(a) What is the Utilization and throughput for your schedule for Machine A?

Utilization is basically the amount of CPU cycles actually used divided by the time it took to use them. $Util = \frac{25+15+15+8}{94} = \frac{63}{94} = 0.67$

Throughput is a measure of jobs per second, in this case $Throughput = \frac{4}{94} = 0.043 \frac{Jobs}{sec}$

(b) Machine B?

$$Util = \frac{63}{80} = 0.79$$

$$Throughput = \frac{4}{80} = 0.05 \frac{Jobs}{sec}$$

5. Assuming that there only Job 1 and Job 2 need to be scheduled how many schedules are possible for Machine A?

*This is just a little combinatorics. With four jobs, there are 4 possible choices for the first job to run on the machine. When that's done, then there are 3 jobs to choose from. Then 2, then 1. This results in a total number of combinations of $4 * 3 * 2 * 1 = 24$. For two jobs, the total number of combinations is just two (either 1;2 or 2;1).*