CG2271 Real Time Operating Systems

Tutorial 6

Question 1

a. Explain why a process using DMA does not require the CPU when it is waiting for I/O, and is therefore safe to put into a blocked state.

   A process using DMA cannot progress until the DMA completes. It is therefore not using CPU and is safe to put into the blocked state.

b. Suppose we have a device that requires the process to transfer data on its own, byte by byte, using an algorithm like this:

   i. Initiate read on device.
   ii. Wait for byte or END signal.
   iii. if END signal received, goto step vi.
   iv. Transfer byte to buffer.
   v. Goto step ii.
   vi. Exit.

Explain the impact of such a device on a system that has to run multiple processes. In particular, what further impact is there on the system if each byte of data must be read within a limited time span?

If the process has to transfer data on its own, it has to use the CPU to do this. This means that this process is not idling and cannot be moved into the blocked state. More importantly if the data must be read within a limited time span, pre-empting the process can cause data loss and corruption.

Question 2

a. Given that the probability of a process accessing an I/O device (via DMA or other similar mechanism) is p, explain why the probability of n processes accessing an I/O device is p^n.

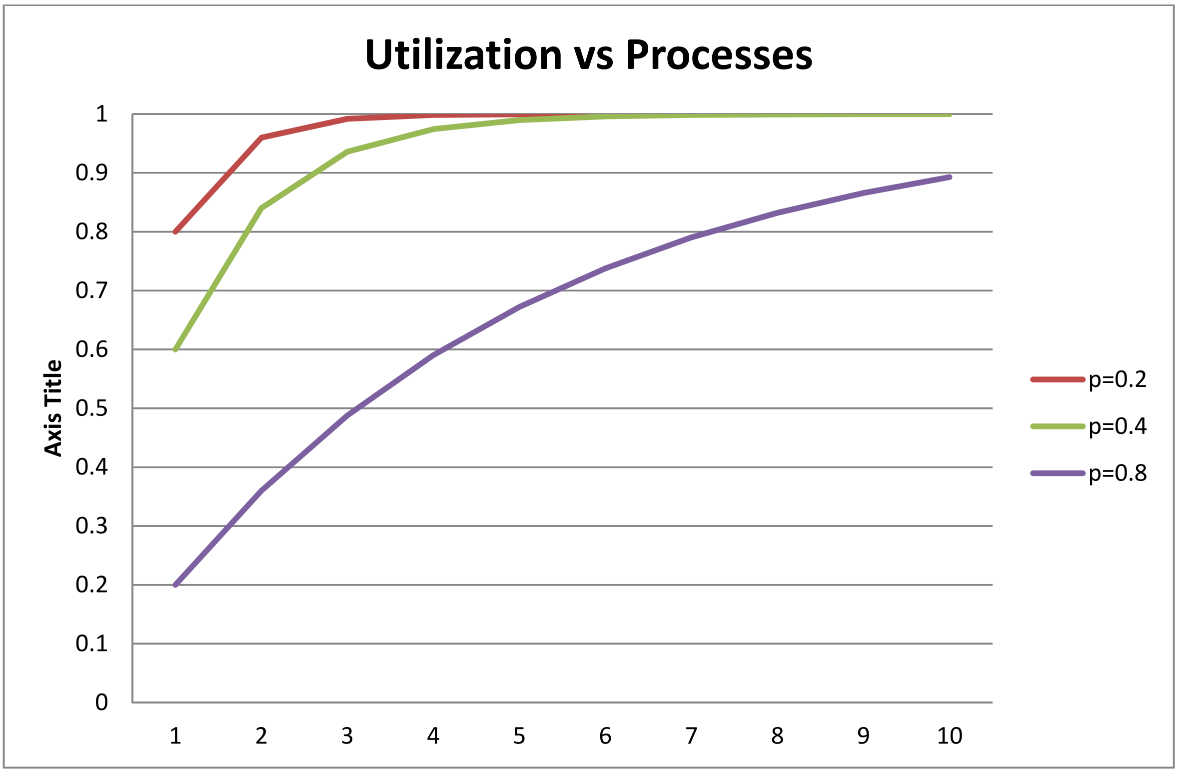
This follows from standard probability. If P1 and P2 have probability *p* of accessing I/O, then the probability of P1 AND P2 accessing I/O at the same time is $p*p=p^2$. Likewise for P1..PN, the probability of all processes accessing I/O at the same time is $p*p*..*p = p^n$.

b. Now explain why the CPU utilization would be given by $1-p^n$.

Since the probability that *n* processes using I/O at the same time is p^n, then the probability of at least one process not using I/O is $1-p^n$. Since we assume that a task is either doing I/O or using CPU, this gives us a utilization of $1-p^n$.

Question 3

a. Using Excel or any similar software, plot the CPU utilization chart for $n=1..10$, for $p=0.2$ using the formula $Utilization=1-p^n$. Similarly plot charts for $p=0.4$, $p=0.8$ on different charts.



b. Based on the charts that you obtain, explain how n and p affect the CPU utilization.

Higher n increases utilization since there's more "opportunity" for execution. However higher p places a lower cap on utilization because the processes are likelier to be waiting on I/O and therefore cannot be run on the CPU.