CS 4/53201 # Homework #2 **Operating Systems**

Due in class on Friday 19 October 2001

1. **(Exercise 7.2 from OSC 6th edition, not in 5th edition) Explain why spinlocks are not appropriate for uniprocessor systems yet may be suitable for multiprocessor systems.**

   Spinlocks are not appropriate in a uniprocessor system because they would tie up the CPU doing essentially "useless" work. However, in a multiprocessor system, even if one processor is busy waiting, other processors can still get useful work done. Moreover, if the time spent busy waiting is shorter than the context switch time, it is more efficient to simply busy wait in this situation.

2. **In the Coke Machine example of Lecture 11, consider the code implementing ThirstyPerson. If the order of the fullSlot.P( ) and mutex.P( ) statements were reversed, would the code still work? Explain your answer.**

   The code will generally work, except in one particular situation. The point of the fullSlot.P( ) statement is to ensure that there is at least one full slot (containing a Coke); if there is no full slot (i.e., the machine is empty) ThirstyPerson blocks and waits for the DeliveryPerson to put a Coke into the machine. However, if ThirstyPerson makes this check inside the critical region, then DeliveryPerson can not get inside the critical region to put a Coke into the machine, so the DeliveryPerson will never signal ThirstyPerson that a Coke is available, and the whole system will deadlock.

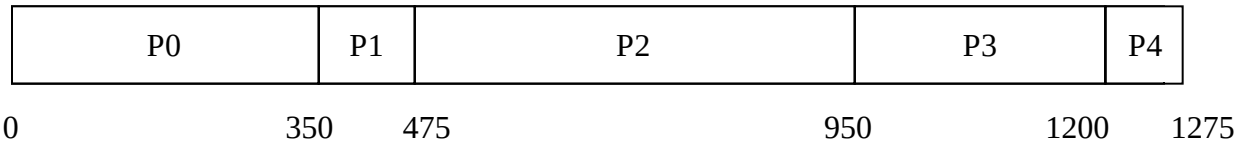
3. **(Exercise 15.2 from OSC 5th and 6th editions) Why do most WANS employ only a partially connected topology?**

   Because WANs span a wide geographic area, it would be prohibitively expensive to employ a fully-connected topology. Think of separate wires running from your PC to each and every person on the Internet — not remotely practical.

4. **(Exercise 15.6 from OSC 5th and 6th editions) Explain why the doubling of the speed of the systems on the Ethernet segment may result in decreased network performance. What changes could ameliorate the problem?**

   Because this could mean that, in a given amount of time, each system could try to send twice as much network traffic, leading to more collisions. One relatively unsatisfactory solution to this problem would be to remove half the nodes. A better solution would be to double the packet length (sending twice as much data each time) so there are fewer attempts to send a "new" packet.

5. Consider five processes that arrive into the ready list at the same time in the following order, with the CPU burst times shown: P0 (350), P1 (125), P2 (475), P3 (250), P4 (75). Draw a Gantt chart illustrating the execution of these processes using a FCFS algorithm, and then compute the average turnaround time (show your work in this computation).



Note that this question asked for average ***turnaround*** time, not average wait time (turnaround time is the time taken for the process to complete once it enters the system).

ATT = (350 + 475 + 950 + 1200 + 1275) / 5 = 4250/5 = 850

If you had been asked to compute the average waiting time, the answer would have been:

AWT = (0 + 350 + 475 + 950 + 1200) / 5 = 2975/5 = 595