CS4328 & CS5305: Homework #3

Due on April 12, 2024

Mina Guirguis

PLEASE READ: You may discuss this problem set with other students. However, you must *write up your answers on your own*. You must also write the names of other students you discussed any problem with. Each problem has a different weight. Please state any assumptions you are making in solving a given problem. *Show your work*. Late assignments will not be accepted with prior arrangements. By submitting this assignment, you acknowledge that you have read the course syllabus.

Problem 1

Consider the following program running on a single processor machine: **[10 pts]**

```
const int n = 50;
int tally;

void Inc() {
   int count;
   for (count = 1; count<=n; count ++)
      tally++
}

void Dec() {
   int count;
   for (count = 1; count<=n; count ++)
      tally--
}

void main() {
   tally = 0;
   parbegin(Inc(),Dec()); // parbegin executes the functions in parallel
   write tally;
}
```

(a) What is the lower bound and upper bound on the final value of the shared variable tally in this concurrent program. Assume processes can execute at any relative speed and that the value can only be incremented/decremented after it has been loaded into a register by a separate machine instruction.

(b) What is the lower and upper bounds if we execute the following process?

```
void main() {
   tally = 0;
   parbegin(Inc(),Inc()); // parbegin executes the functions in parallel
   write tally;
}
```

Problem 2

Show that, if the wait() and signal() semaphore operations are not executed atomically, then mutual exclusion may be violated. **[10 pts]**

Problem 3

The Under-equipped Mechanic Shop problem is a synchronization problem in which 3 mechanics work in an under-equipped shop and are forced to share 3 available tools (A, B and C). The 3 mechanics continuously repair parts and take breaks after fixing a part. Mechanic 1 needs the 3 tools to repair a part, Mechanic 2 only needs A and C to repair a part, and Mechanic 3 only needs B and C to repair a part. Write a program/pseudocode to synchronize between these 3 mechanics. Explain whether a deadlock can occur or not in your program.**[20 pts]**

Problem 4

Consider the following snapshot of a system. Answer the following questions: **[24 pts]**

Current available:

R1	R2	R3	R4
2	1	0	0

Current Allocation:

P-R	R1	R2	R3	R4
P1	0	0	1	2
P2	2	0	0	0
P3	0	0	3	4
P4	2	3	5	4
P5	0	3	3	2

Maximum Claim:

P-R	R1	R2	R3	R4
P1	0	0	1	2
P2	2	7	5	0
P3	6	6	5	6
P4	4	3	5	6
P5	0	6	5	2

(a) What are the total number of resources present in the system?

(b) Compute what each process might still need.

(b) Is this system in a safe or unsafe state? Why?

(d) Is this system currently deadlocked? Why or why not?

(e) Which processes, if any, are or may become deadlocked?

(f) If a request from P3 arrives (0,1,0,0), can that request be safely granted? If granted, what would be the resulting state (safe, unsafe, deadlocked)? Which processes, if any, are or may become deadlocked if this request was immediately granted?

Problem 5

Write a program/pseudo-code to show how solving the dining philosophers problem can be done by allowing each philosopher to grab both chopsticks at once. Discuss any drawbacks of your solution. **[10 pts]**

Problem 6

(a) Three processes share M resources units that can be reserved and released only one at a time. Each process needs a maximum of 3 units. What is the minimum value of M so that a deadlock cannot occur? **[5pts]**

(b) N processes share M resource units that can be reserved and released only one at a time. The maximum need of each process does not exceed M, and the sum of all maximum needs in less than M+N. Show that a deadlock cannot occur.**[5 pts]**