Vivian Do
CSC 4320
Mar 26 2023

Spring 2023 – 4320/6320 Section 006 Operating Systems

Homework 4: Due 03/26/2022 at 11:59 PM

Your programs – if requested – must compile with gcc and execute on snowball.cs.gsu.edu! Please see https://cscit.cs.gsu.edu/sp/guide/snowball for more details. You may use whatever IDEs / text editors you like, but you must submit your responses on iCollege.

1. **(10 Points)** Provide a deadlock-prone pseudocode for two processes each accessing two semaphores A and B.

My Pseudocode:

Process 1 = p1 Semaphore 1 = A
Process 2 = p2 Semaphore 2 = B

p1 locks A, waiting for B
p1 acquires B
p2 locks B, waiting for A
p2 acquires A

p1 releases A
p1 releases B

p2 releases B
p2 releases A

2. **(20 Points)** Prove the correctness or give a counter-example for each of the following statements. You must state whether the statement is true or false and then show your arguments. ("->" means "implies").

   a. Cycle -> Deadlock

   b. Knot -> Deadlock

a) Cycle -> Deadlock = False
In a resource allocation graph, it is possible that there is a cycle but there are no deadlocks since there are more than one instance of resource. This makes a cycle a bad condition for a deadlock.

b) Knot -> Deadlock = True
In a connected subgraph of a directed graph, it is impossible to leave the knot following the edge of the graph when starting from any node in the subset. This makes a knot a good condition for a deadlock.

3. **(20 Points)** Consider the following maximum-claim reusable resource system with four processes (P_0, P_1, P_2, P_3) and three resource types (R_0, R_1, R_2). The maximum claim matrix C is given by

	R_0	R_1	R_2
P_0	4	1	4
P_1	3	1	4
P_2	5	6	13
P_3	1	1	6

where C_{ij} denote maximum claim of process i for resource j. The total number of units of each resource type is given by the vector

R0	R1	R2
5	8	15

The current allocation of resources is given by the matrix A

	R_0	R_1	R_2
P_0	0	1	4
P_1	2	0	1
P_2	1	2	1
P_3	1	0	3

where $A_{i,j}$ denotes the units of resources of type j currently allocated to process i. For the state shown above, determine if a new request by process P_1 for 1 unit of resource R_1 can be safely granted. (Remark: You can assume that the system state above does not yet take into account this request.)

Process 1 = p1 Resource 1 = r1

p1 Max Need - p1 Current Allocation = Future Need (Requirement)
[3 1 4] - [2 0 1] = [1 1 3]

p1 Requirement = [1 1 3] is requesting of r1
p1 Requirement + r1 Max Need
[1 1 3] + [x 1 x] = [1 2 3]

p1 Final Requirement = [1 2 3]

Therefore, the new request by p1 for one unit of r1 can be safely granted from using the available resources.

My Handwritten Chart/Work (?):