CS471: Operating System Concepts
Fall 2006
(Lecture: TR 11:25-12:40 PM)
Homework #4
Points: 20
Due: September 26, 2006
Solution

Question 1 [Points 10] Exercise 7.11

Consider the following snapshot of a system:

	Allocation	Max	Available
	A B C D	A B C D	A B C D
P0	0 0 1 2	0 0 1 2	1 5 2 0
P1	1 0 0 0	1 7 5 0	
P2	1 3 5 4	2 3 5 6	
P3	0 6 3 2	0 6 5 2	
P4	0 0 1 4	0 6 5 6	

Answer the following questions using the banker's algorithm:

a) What is the content of the need matrix ***Need*****?**

The content of matrix *Need* is defined as *Max – Allocation.*

	Need			
	A	**B**	**C**	**D**
P0	**0**	**0**	**0**	**0**
P1	**0**	**7**	**5**	**0**
P2	**1**	**0**	**0**	**2**
P3	**0**	**0**	**2**	**0**
P4	**0**	**6**	**4**	**2**

b) Is the system in a safe state?

The available number of resources of A B C D are 1 5 2 0 respectively.

Since process P0 doesn't need any further resource to complete, let P0 be completed and the available resources after P0 finishes are 1 5 3 2.

After P0 available resources are 1 5 3 2.

The available resource 1 5 3 2 can satisfy the needs of either process P2 or P3.
Let process P2 given the resources that are needed.

After P2 finishes, the available resources are 2 8 8 6.

The available resources 2 8 8 6 can satisfy the needs of the process P3.

Let process P3 given the resources that are needed.

After P3 finishes, the available resources are 2 14 11 8.

The available resources 2 14 11 8 can satisfy the needs of the process P1.

Let process P1 given the resources that are needed.

After P1 finishes, the available resources are 3 14 11 8.

The available resources 3 14 11 8 can satisfy the needs of the process P4.

Let process P4 given the resources that are needed.

After P4 finishes the available resources are 3 14 12 12.

Since all the processes can be satisfied, the system is in a safe state.

The sequence <P0, P2, P3, P1, P4> satisfies the safety criteria.

C) If a request from process P1 arrives for (0,4,2,0) can the request be satisfied immediately?

The process P1 request can be granted immediately. After granting the P1 request, for the sequence <P0, P2, P1, P3, P4> the system is in safe state.
Justification: Suppose this request is granted, the new need matrix is:

	Need			
	A	**B**	**C**	**D**
P0	**0**	**0**	**0**	**0**
P1	**0**	**3**	**3**	**0**
P2	**1**	**0**	**0**	**2**
P3	**0**	**0**	**2**	**0**
P4	**0**	**6**	**4**	**2**

Available resources: 1 1 0 0

Once again, P0 can finish, releasing <0 0 1 2>. So the **new available is <1 1 1 2>**
P2 can now finish. The **new available is <2 4 6 6>.**
With this, P1 can finish; then P3, then P4 (for example).
So they can finish in the order <P0, P2, P1, P3, P4>.
So it is a safe state.

Question 2 [Points 5] Exercise 7.5

a) Increase ***Available*** (new resources added).

If there was no deadlock before, there will be no deadlock after.

b) Decrease ***Available*** **(resource permanently removed from the system)**

Only safe if the banker's algorithm tells us that the system is in a safe state with the reduced number of resources.

c) Increase ***Max*** for one process (the process needs more resources than allowed, it may want more)

Only safe if the system will be in a safe state with the increased value of ***Max.***

d) Decrease the ***Max*** for one process (the process decides it does not need that many resources).

If it was safe before, it will be safe afterwards.

e) Increase the number of processes.
Increased processes may lea to starvation for some but not deadlock.

f) Decrease the number of processes.

If safe before, it will be safe after.

Question 3 [Points 5] Exercise 7.6

Consider a system consisting of four resources of the same type that are shared by three processes, each of which needs at most two resources. Show that the system is deadlock free.

Solution:

So there are four instance of resource R.
Three processes P1, P2, and P3 are contending for it.
Each needs at most two of R.

Possible ways in which all 4 resources may be allocated to the 3 processes is:

P0	P1	P2	Execution
1	1	2	P2 finishes; then P0 and P1
1	2	1	P1 finishes; then P0 and P2
2	0	2	P0 and P2 finish; then P1
2	1	1	P0 finishes; then P1 and P2
2	2	0	P0 and P1 finish; then P2

Thus, it is deadlock free. In summary, since at least one process can always get its maximum resource requirement fulfilled, it will definitely finish. In that case, once it finishes it will release its resources. Then, there would be 4 resources for 2 processes; hence, both can get it; so there is no deadlock.