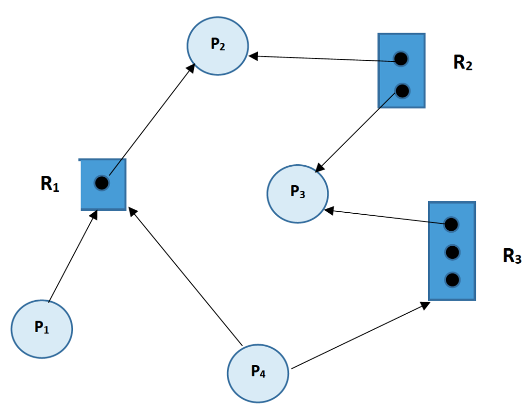
Tutorial 5
Chapter 7 – Deadlocks
Operating Systems CS 3SH3 Term 2, Winter 2022
Prof. Neerja Mhaskar

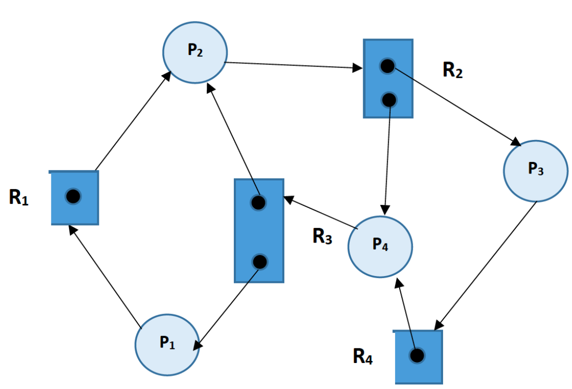
Tutorials are not mandatory. They are simply a tool for you to understand the course concepts better.
Tutorial Format: The questions will be posted a day before or on the day of the tutorial on the course website. You can choose to solve these problems before hand and come in with your solutions. I or one of the TAs helping me will check your solutions. If you have all of the questions correct you can choose to leave. If you have any of them incorrect, it is recommended that you stay and understand the solutions.
Solutions to the tutorial will not be posted online.

Questions:

1. Consider the below resource allocation graph. Is the system in a deadlock state? If so, report the cycle(s) causing deadlock. If not, explain the order in which processes access the resources requested and complete execution.



2. Consider the below resource allocation graph. Is the system in a deadlocked state? If so, report the cycle(s) causing deadlock. If not, explain the order in which processes access the resources requested and complete execution.

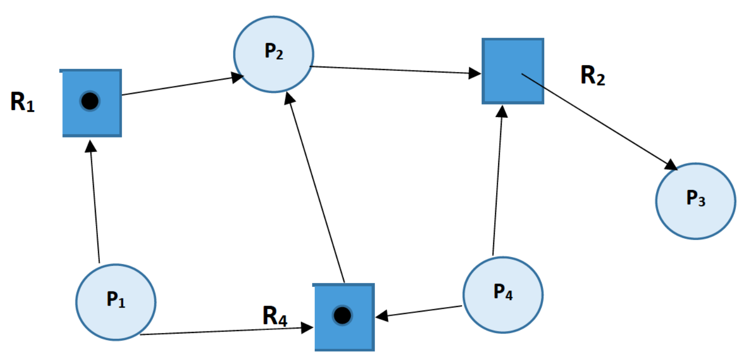


3. Consider the following snapshot of a system:
   Answer the following questions using the banker's algorithm:

	Allocation	*Max*	*Available*
	A B C D	*A B C D*	*A B C D*
P_0	0 0 1 2	0 0 1 2	1 5 2 0
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 6 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

a. What is the content of the matrix Need?
b. Is the system in a safe state?
c. If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately?

4. Consider the below resource allocation graph. Construct the corresponding wait-for graph. Is the system in deadlock? If so, provide the cycle causing deadlock.



5. Consider the following snapshot of a system at time *T*0:

   Five processes ***P*0** through ***P*4**.

   Three resource types A (10 instances), *B* (3 instances), and *C* (6 instances)

   Snapshot at time ***T*0**:

	Allocation	*Request*	*Available*
	A B C	*A B C*	*A B C*
*P*0	2 1 1	0 0 0	0 0 0
*P*1	2 1 2	2 0 2	
*P*2	4 0 0	0 0 1	
*P*3	2 1 1	1 0 0	
*P*4	0 0 2	0 0 2	

a) Is the system in deadlocked state? If no, provide a sequence of processes that show that shows the no deadlock. If yes, explain why and list the processes involved in the deadlock.

b) Suppose process P1 makes an additional request of resource type B, the Request matrix is modified as follows:

	Request
	A B C
*P*0	0 0 0
*P*1	2 1 2
*P*2	0 0 1
*P*3	1 0 0
*P*4	0 0 2

c) Is the system in deadlocked state? If no, provide a sequence of processes that show that shows the no deadlock. If yes, explain why and list the processes involved in the deadlock.

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.

7. Consider the version of the dining-philosophers problem in which the chopsticks are placed at the center of the table and any two of them can be used by a philosopher. Assume that requests for chopsticks are made one at a time. Describe a simple rule for determining whether a particular request can be satisfied without causing deadlock given the current allocation of chopsticks to philosophers.