Study Set for Lecture 08: Deadlocks

Due Oct 19 at 11:59pm	**Points** 10	**Questions** 12	**Available** after Oct 13 at 12pm
Time Limit None	**Allowed Attempts** Unlimited		

Instructions

Review lecture notes from lect08_Deadlocks.pdf.

Then answer the questions from this study set and submit them to gain access to the further part of the course.

Take the Quiz Again

Attempt History

	Attempt	Time	Score
KEPT	Attempt 3	2 minutes	0 out of 10
LATEST	Attempt 3	2 minutes	0 out of 10
	Attempt 2	9 minutes	0 out of 10 *
	Attempt 1	2,867 minutes	0 out of 10 *

* Some questions not yet graded

Correct answers are hidden.

Score for this attempt: **0** out of 10
Submitted Oct 21 at 8:58am
This attempt took 2 minutes.

Question 1

0 / 0 pts

What are the conditions necessary for a deadlock to occur?

Do they all need to be satisfied? Will just one be sufficient?

Support your answer with details and examples.

Your Answer:

Deadlock:

- The computer system uses may types of resource which are then used by various processes to carry out their individual functions.
- But problem is that the amount of resources available is limited and many process needs to use it.
- A set of process is said to be in a deadlocked state when every process in the set is waiting for an event that can be caused only by another process in the set. The event can

be resource acquisition, resource release etc. The resource can be physical (printers, memory space) or logical (semaphores, files)

Mutual Exclusion
only one process at a time can use a resource
Hold and wait
a process holding at least one resource is waiting to aquire additional resources held by other processes.
No preemption
a resource can be release only voluntarily by the process holding it after that process has completed its task
Circular wait
There exists a set {P0,P1,...,Pn} of waiting processes such that P0 is waiting for a resource that is held by P1,P1 is waiting for a resource that is held by P2, ..., Pn-1 is waiting for a resource that is held by Pn, and Pn is waiting for a resource that is held by P0.

Question 2

0 / 0 pts

Define what a resource allocation graph is.

Let there will be a cycle in a resource allocation graph. Can you tell if there is a deadlock?

Support your answer with all necessary details and an example.

Your Answer:

A resource allocation graph is a set of vertices and edges such that the vertices are partitioned into a set of processes and a set of all resource types in the system.

If there's a cycle and all instances of all resources in the set are taken up by the processes in the cycle, then there's a deadlock.

However, a cycle doesn't necessarily mean there's a deadlock, cause there might be extra instances of resources that are being used by processes that aren't in the cycle, which will be returned after the processes exit, resolving the deadlock.

Question 3

0 / 0 pts

Define deadlock prevention and deadlock avoidance.

Explain what is the difference between the two.

Your Answer:

Deadlock prevention - engineer the system in a way that eliminates deadlocks completely

Deadlock avoidance - Manage system so it avoids dangerous states, such as ones that may lead to deadlock

Deadlock prevention - implements strict rules that restrict user programs, which are often expensive on the overhead of the system. Deadlock avoidance is often less expensive and less strict but is a bit riskier, so not often used in mission-critical systems.

Question 4

0 / 0 pts

Discuss the use of resource numbering scheme to prevent deadlocks. Describe how the effectiveness of the scheme can be proven formally.

Then, illustrate the theory by designing a deadlock prevention algorithm based on resource numbering for the the Dining Philosophers problem.

Your Answer:

The resource numbering scheme gives each resource a number which indicates which process can access it. A process can request a resource only if the number of it's currently held resource has a lower number than the one it's requesting. It can be proven by contradiction, where we assume that we got a deadlock, which means that a process got a hold of a resource of a lower number than the one it currently was holding since it needs to be in a cycle for there to be a deadlock, which would have been impossible given that the process can't do that with those rules.

If you consider that each chopstick is given a number from 0-5, the fifth philosopher couldn't grab the 0 chopstick if they already held the 5th chopstick, thus no deadlock will occur because the circular wait condition won't hold.

Question 5

0 / 0 pts

The following is a resource allocation graph:

```
V={P1, P2, P3, P4, P5, R1, R2, R3, R4}
E={(R1, P1), (R2, P2), (R3, P5), (R4, P4), (P1, R2), (P2, R3), (P5, R4), (P4, R1), (P3,
R1), (P3, R3)}
```

Using these data create a an adjacency table for the graph. Then, write a pseudocode for the algorithm that uses the adjacency table to detect whether there is a deadlock in the system.

Your Answer:

R1-> P1

P1-> R2

R2->P2

R3->P5

P5-> R4

R4->P4

P4->R1

P3->R3

P3->R1

There will be a deadlock in this system because P3 is requesting resources from R1 and R3 where P4 is asking for resources from R1 and P2 is asking resources from R3, this causes an issue because P3 would have to wait until R3 and R1 are released to be able to do something.

Question 6

0 / 0 pts

Consider the following state of the system:

```
Process   Allocation     Max       Available
            A B C       A B C        A B C
  P0        0 1 0       7 5 3        3 3 2
  P1        2 0 0       3 2 2
  P2        3 0 2       9 0 2
  P3        2 1 1       2 2 2
  P4        0 0 2       4 3 3
```

Prove that the sequence <P_1, P_3, P_4, P_2, P_0> satisfies the safety criteria.

Use the following Google Sheets **Baker's Algorithm Task Template** **(https://docs.google.com/spreadsheets/d/1Td2hK8dYLoDk4O2TOpfF5HjMW9H4ahELMuxF6O7LOcI/template/previe** to perform the analysis. When you open the template, you need to click on the "Use Template" button to create your own version. After you are done, select "Download" from the "File" menu, then choose PDF, and then submit the downloaded PDF file as your answer.

Banker's Algorithm Problem 1.pdf **(https://cilearn.csuci.edu/files/2199969/download)**

Question 7

0 / 0 pts

Consider the following state of the system:

```
Process   Allocation     Max       Available
            A B C       A B C        A B C
  P0        0 1 0       7 5 3        3 3 2
  P1        2 0 0       3 2 2
  P2        3 0 2       9 0 2
  P3        2 1 1       2 2 2
  P4        0 0 2       4 3 3
```

Prove that a request for (3, 3, 0) cannot be granted to P_4.

Use the following Google Sheets **Baker's Algorithm Task Template (https://docs.google.com/spreadsheets/d/1Td2hK8dYLoDk4O2TOpfF5HjMW9H4ahELMuxF6O7LOcI/template/previe** to perform the analysis. When you open the template, you need to click on the "Use Template" button to create your own version. After you are done, select "Download" from the "File" menu, then choose PDF, and then submit the downloaded PDF file as your answer.

Banker's Algorithm Problem 1.pdf (https://cilearn.csuci.edu/files/2199970/download)

Question 8

0 / 0 pts

Consider the following state of the system:

```
Process   Allocation     Max       Available
            A B C       A B C        A B C
  P0        0 1 0       7 5 3        3 3 2
  P1        2 0 0       3 2 2
  P2        3 0 2       9 0 2
  P3        2 1 1       2 2 2
  P4        0 0 2       4 3 3
```

Prove that granting a request (0, 2, 0) to P_0 will end up in a safe state.

Use the following Google Sheets **Baker's Algorithm Task Template (https://docs.google.com/spreadsheets/d/1Td2hK8dYLoDk4O2TOpfF5HjMW9H4ahELMuxF6O7LOcI/template/previe** to perform the analysis. When you open the template, you need to click on the "Use Template" button to create your own version. After you are done, select "Download" from the "File" menu, then choose PDF, and then submit the downloaded PDF file as your answer.

Banker's Algorithm Problem 1.pdf (https://cilearn.csuci.edu/files/2199971/download)

Question 9

0 / 0 pts

Invent a deadlock avoidance scheme based on the Banker's Algorithm that can be used to solve the Dining Philosophers problem. Provide an example that illustrates how the scheme works.

Provide a formal proof that the implementation indeed prevents deadlocks.

Do not submit code; rather, plain English description of the algorithm and why it will work.

Your Answer:

Given the need of each philosopher and the availible number of chopsticks, use work and finish to determing if allocating a number of chopsticks to a particular philosopher would lead to an unsafe state, if it doesn't, continue, else, stop that particular philosopher from grabbing chopsticks so that others can successfully eat without a deadlock.

If each philosopher tries to grab two chopsticks, the algorithm will eventually detect that one philosopher is trying to grab 2 chopsticks when only 1 is available, so it will block that philosopher until more chopsticks are put down by the other philosophers.

Question 10

0 / 0 pts

The system that you are designing cannot afford the overhead of deadlock prevention and deadlock avoidance. What is needed to implement deadlock detection scheme?

Discuss the options available in dealing with deadlocked systems?

Your Answer:

In order to implement a deadlock detection scheme, you can either use a wait-for graph, which is a resource-allocation graph in which the nodes are processes and the edge indicates that one process is waiting for another process's resource to be released. Either that or you can use the bankers algorithm to detect a deadlock instead of avoid one.

You could try to reclaim a process that's deadlocked by using a rollback, which returns the process to some previous safe state, but that could be expensive. The only other solution is termination, but you could improve that method by strategically choosing which process to terminate with the least amount of collateral.

Question 11

0 / 0 pts

Explain with details and examples why resource allocation graph cannot be used for deadlock avoidance in systems that have resources with multiple instances.

Your Answer:

A cycle in a resource allocation graph doesn't always mean that there's a deadlock. Especially if there are multiple instances of resources, where there could be a deadlock if there is a cycle, but we can't tell. For example, if there is a cycle between two resources with both having two instances, but the second instance of each resource is being allocated by separate processes that aren't a part of the cycle, then those processes will eventually free the resources they used, breaking the deadlock. If the two processes were a part of the cycle it would result in a deadlock.

Question 12

10 / 10 pts

I have submitted answers to all questions in this study set.

[✓ True]

False

Quiz Score: **0** out of 10