Homework 7

ComS 352 - gmonroe

8.12) Consider the traffic deadlock depicted in Figure 8.12.
Show that the four necessary conditions for deadlock hold in this example.
State a simple rule for avoiding deadlocks in this system.

- Mutual exclusion condition applies, since only one vehicle can be on a section of the street at a time.
- Hold-and-wait condition applies, since each vehicle is occupying a section of the street, and waiting to move on to the next section of the street.
- No-preemption condition applies, since a section of the street that is a section of the street that is occupied by a vehicle cannot be taken away from it.
- Circular wait condition applies, since each vehicle is waiting on the next vehicle to move. That is, each vehicle in the traffic is waiting for a section of street held by the next vehicle in the traffic.

Dealing with Deadlock Problem
In general, there are four strategies of dealing with deadlock problem:
1. The Ostrich Approach

Just ignore the deadlock problem altogether.

2. Deadlock Detection and Recovery

Detect deadlock and, when it occurs, take steps to recover.

3. Deadlock Avoidance

Avoid deadlock by careful resource scheduling.

4. Deadlock Prevention

Prevent deadlock by resource scheduling so as to negate at least one of the four conditions.
Now we consider each strategy in order of decreasing severity.

Deadlock Prevention
Havender in his pioneering work showed that since all four of the conditions are necessary for deadlock to occur, it follows that deadlock might be prevented by denying any one of the conditions.

8.16)The program example shown in Figure 8.1 doesn't always lead to dead- lock. Describe what role the CPU scheduler plays and how it can con- tribute to deadlock in this program.

Part of the CPU scheduler is essentially to distribute CPU time to diverse forms and strings. Within the over program, When thread1 tries to execute work_one, it checks whether first_mutex is accessible implies moment string isn't doing any work and after that locks it. At that point it checks the second_mutex is available means the moment string isn't doing any work and after that locks it. There are two checks since there are two diverse mutex locks. on the off chance that any check is unsuccessful, thread1 will hold up for moment string to total work. When both locks are accessible and both checks are passed, thread1 will do its work and open the both mutex a while later. Additionally the over handle is happening for thread2. CPU scheduler can lead to halt. Consider the circumstance where thread1 will execute work_one line 1 i.e. it locks first_mutex. At that point the string is delayed and the CPU is apportioned to thread2. Additionally thread2 will execute work_two line 1 i.e. it locks second_mutex. Presently since first_mutex is bolted, line 2 will not execute assist , thread2 will wait. When CPU will be designated to thread1, line 2 will not execute advance, it'll too hold up since second_mutex isn't free(locked). Both strings will hold up inconclusively for each other. This will result in halt. In this way the CPU scheduler will result in halt in this case.

8.18) Which of the six resource-allocation graphs shown in Figure 8.12 illus- trate deadlock? For those situations that are deadlocked, provide the cycle of threads and resources. Where there is not a deadlock situation, illustrate the order in which the threads may complete execution.

(b) Deadlock
T1 is waiting for the resource R3.
R3 is allocated to T3, which is waiting for R1 -> R1 is allocated to T1.
(d) Deadlock
T1 and T2 area allocated R1 which has two instances for which T3 and T4 are waiting.
But, T1 is waiting for R2 which is two instances which in turn allocated to T3 and T4.
(f) DeadLock
Resource R2 has two instances. But, it is allocated to 3 process called T2,T3 and T4.

(a) T2-->T3-->T1. or T2-->T1-->T3

(c) T2-->T3-->T1 or T3-->T2-->T1

(e) T2-->T1-->T3-->T4

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

A system consisting of four resources of the same type that are shared by three threads each of which need at most two resources we have to show that the system is deadlock free follow the explanation given below:
The system is only deadlocked if the process cannot access the necessary amount of resources if needed. If three process need a maximum of 2 resources total that means one of the process ,must needs to be completed after that process completes more resources are freed and the older processes can complete too

8.24)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.

To prevents deadlock follow the below rule:

When a philosopher makes a request for the first chopstick, simply, do not grant the request if there is no other philosopher with two chopsticks and if there is only one chopstick remaining.

8.28 a)Illustrate that the system is in a safe state by demonstrating an order in which the threads may complete.

Need T2 < available so, T2 can take all resources, Available = (2 2 2 4) + (2 4 1 3) (Allocation of T2) = 4 6 3 7

Need (T3) < Available so, T3 will go next, Available = (4 6 3 7) + (4 1 1 0) = (8 7 4 7) Likewise next to T0, T1, T4, will get resources, so now safe sequences is T2, T3, T0, T1, T4

8.28 b)If a request from thread T4 arrives for (2, 2, 2, 4), can the request be granted immediately?

Yes, it can be request and generated immediately, As by request (T4). Request From T4 is (2 2 2 4) and Available is (2 2 2 4)

8.28 c)If a request from thread T2 arrives for (0, 1, 1, 0), can the request be granted immediately?

Yes, it can be request and generated immediately, As by request (T3). Request From T3 is (2 2 1 2) and Available is (2 2 2 4)