TECHNO INTERNATIONAL BATANAGAR

OPERATING SYSTEM TERM PAPER

TOPIC :: *Deadlock*

Name : Sachin Sinha

Roll : 33200118018

Year : 3rd Year

Semester: 5th

Dept. : Computer Science Engineering

Introduction of Deadlock in Operating System

A process in operating systems uses different resources and uses resources in the following way.
1) Requests a resource
2) Use the resource
2) Releases the resource

Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.
Consider an example when two trains are coming toward each other on the same track and there is only one track, none of the trains can move once they are in front of each other. A similar situation occurs in operating systems when there are two or more processes that hold some resources and wait for resources held by other(s). For example, in the below diagram, Process 1 is holding Resource 1 and waiting for resource 2 which is acquired by process 2, and process 2 is waiting for resource 1.

Deadlock can arise if the following four conditions hold simultaneously

(Necessary Conditions) Mutual Exclusion: One or more than one resource are non-shareable (Only one process can use at a time)
Hold and Wait: A process is holding at least one resource and waiting for resources.
No Preemption: A resource cannot be taken from a process unless the process releases the resource.
Circular Wait: A set of processes are waiting for each other in circular form.

Methods for handling deadlock

There are three ways to handle deadlock
1) Deadlock prevention or avoidance: The idea is to not let the system into a deadlock state. One can zoom into each category individually, Prevention is done by negating one of above mentioned necessary conditions for deadlock.
Avoidance is kind of futuristic in nature. By using strategy of “Avoidance”, we have to make an assumption. We need to ensure that all information about resources which process will need are known to us prior to execution

of the process. We use Banker's algorithm (Which is in-turn a gift from Dijkstra) in order to avoid deadlock.

2) Deadlock detection and recovery: Let deadlock occur, then do preemption to handle it once occurred.

3) Ignore the problem altogether: If deadlock is very rare, then let it happen and reboot the system. This is the approach that both Windows and UNIX take.

4) Difference between Deadlock Prevention and Deadlock Avoidance

1. Deadlock Prevention :

Deadlock prevention means to block at least one of the four conditions required for deadlock to occur. If we are able to block any one of them then deadlock can be prevented.

The four conditions which need to be

blocked are:- Mutual Exclusion
Hold and Wait
No
Preemption
Circular
Wait
Spooling and non-blocking synchronization algorithms are used to prevent the above conditions.In deadlock prevention all the requests are granted in a finite amount of time.

2. Deadlock Avoidance :

In Deadlock avoidance we have to anticipate deadlock before it really occurs and ensure that the system does not go in unsafe state.It is possible to avoid deadlock if resources are allocated carefully. For deadlock avoidance we use Banker's and Safety algorithm for resource allocation purpose. In deadlock avoidance the maximum number of resources of each type that will be needed are stated at the beginning of the process.

Program for Deadlock free condition in Operating System

Given: A system has R identical resources, P processes competing for them and N is the maximum need of each process. The task is to find the minimum number of Resources required So that deadlock will never occur.

Formula:

R >= P * (N - 1) + 1

Examples:

Input : P = 3,
N = 4 Output :
R >= 10

Input : P = 7,
N = 2 Output :
R >= 8

Approach:

Consider, 3 process A, B
and C. Let, Need of each
process is 4

Therefore, The maximum resources require will be 3 * 4 = 12 i.e, Give 4 resources to each
Process.

And, The minimum resources required will be 3 * (4 – 1) + 1 = 10.

i.e, Give 3 Resources to each of the Process, and we are left out with 1 Resource.

That 1 resource will be given to any of the Process A, B or C.
So that after using that resource by any one of the Process, It left the resources and that resources will be used by any other Process and thus Deadlock will Never Occur.

Recovery from Deadlock in Operating
System Prerequisite – Deadlock Detection
And Recovery
When a Deadlock Detection Algorithm determines that a deadlock has occurred in the system, the system must recover from that deadlock. There are two approaches of breaking a Deadlock:

1. Process Termination:
To eliminate the deadlock, we can simply kill one or more processes. For this, we use two methods:

(a). Abort all the Deadlocked Processes:
Aborting all the processes will certainly break the deadlock, but with a great expenses. The deadlocked processes may have computed for a long time and the result of those partial computations must be discarded and there is a probability to recalculate them later.
(b). Abort one process at a time untill deadlock is eliminated:
Abort one deadlocked process at a time, untill deadlock cycle is eliminated from the system. Due to this method, there may be considerable overhead, because after aborting each process, we have to run deadlock detection algorithm to check whether any processes are still deadlocked.

2. Resource Preemption:
To eliminate deadlocks using resource preemption, we preepmt some resources from processes and give those resources to other processes. This method will raise three issues-

(a). Selecting a victim:
We must determine which resources and which processes are to be preempted and also the order to minimize the cost.
(b). Rollback:
We must determine what should be done with the process from which resources are preempted. One simple idea is total rollback. That means abort the process and restart it.

(c). Starvation:
In a system, it may happen that same process is always picked as a victim. As a result, that process will never complete its designated task. This situation is called Starvation and must be avoided. One solution is that a process must be picked as a victim only a finite number of times. Attention reader! Don’t stop learning now. Get hold of all the important CS Theory concepts for SDE interviews with the CS Theory Course at a student-friendly price and become industry ready.