SP19-BCS-078

Saud ul Hassan

OS Theory Assignment 3

# **Task**:

Understand all the deadlock detection and recovery techniques, make a word document on them with simplified words.

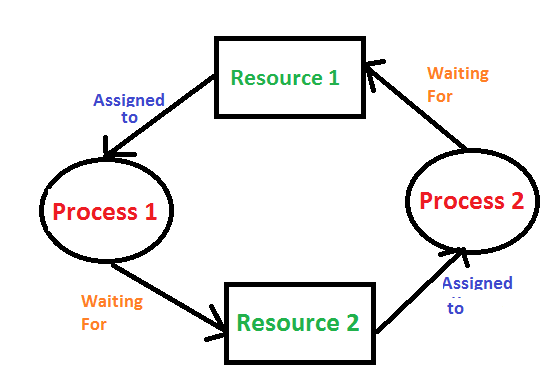
**Deadlock**

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. The four necessary conditions for a deadlock situation are mutual exclusion, no preemption, hold and wait and circular set.

**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).

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.



**Methods:**

There are four methods of handling deadlocks - deadlock avoidance, deadlock prevention, deadlock detection and recovery and deadlock ignorance.

**Deadlock Detection in OS:**

A deadlock occurrence can be detected by the resource scheduler. A resource scheduler helps the OS to keep track of all the resources which are allocated to different processes. So, when a deadlock is detected, it can be resolved using the below-given methods:

**Deadlock Prevention in OS**

It’s important to prevent a deadlock before it can occur. The system checks every transaction before it is executed to make sure it doesn’t lead to deadlock situations. Such that even a small chance that an operation which can lead to Deadlock in the future is never allowed to execute.

It is a set of methods for ensuring that at least one of the conditions cannot hold.

1. **No preemptive action:**

No Preemption – A resource can be released only voluntarily by the process holding it after that process has finished its task.

* If a process which is holding some resources requests another resource that can’t be immediately allocated to it, in that situation, all resources will be released.
* Preempted resources require the list of resources for a process that is waiting.
* The process will be restarted only if it can regain its old resource and a new one that it is requesting.
* If the process is requesting some other resource, when it is available, then it is given to the requesting process.
* If it is held by another process that is waiting for another resource, we release it and give it to the requesting process.

1. **Mutual Exclusion:**

Mutual Exclusion is a full form of Mutex. It is a special type of binary semaphore which is used for controlling access to the shared resource. It includes a priority inheritance mechanism to avoid extended priority inversion problems. It allows current higher priority tasks to be kept in the blocked state for the shortest time possible.

Resources shared such as read-only files never lead to deadlocks, but resources, like printers and tape drives, need exclusive access by a single process.

1. **Hold and Wait:**

In this condition, processes must be stopped from holding single or multiple resources while simultaneously waiting for one or more others.

1. **Circular Wait:**

It imposes a total ordering of all resource types. Circular wait also requires that every process request resources in increasing order of enumeration

**Deadlock Avoidance:**

When a process requests a resource, the deadlock avoidance algorithm examines the resource-allocation state. If allocating that resource sends the system into an unsafe state, the request is not granted.

Therefore, it requires additional information such as how many resources of each type is required by a process. If the system enters into an unsafe state, it has to take a step back to avoid deadlock.

**Avoidance Algorithms:**

The deadlock-avoidance algorithm helps you to dynamically assess the resource-allocation state so that there can never be a circular-wait situation.

A single instance of a resource type.

* Use a resource-allocation graph
* Cycles are necessary which are sufficient for Deadlock

Multiple instances of a resource type.

* Cycles are necessary but never sufficient for Deadlock.
* Uses the banker’s algorithm

**Deadlock Recovery:**

We let the system fall into a deadlock and if it happens, we detect it using a detection algorithm and try to recover.

Some ways of recovery are as follows.

* Aborting all the deadlocked processes.
* Abort one process at a time until the system recovers from the deadlock.
* Resource Preemption: Resources are taken one by one from a process and assigned to higher priority processes until the deadlock is resolved.

**Deadlock Ignorance:**

In the method, the system assumes that deadlock never occurs. Since the problem of deadlock is not frequent, some systems simply ignore it. Operating systems such as UNIX and Windows follow this approach. However, if a deadlock occurs we can reboot our system and the deadlock is resolved automatically.

This is an example of the Ostrich Algorithm. It is a strategy of ignoring potential problems on the basis that they are extremely rare.