Deadlock Prevention And Avoidance

Difficulty Level : Easy Last Updated : 21 Sep, 2021

Deadlock Characteristics

As discussed in the previous post, deadlock has following characteristics.

1. Mutual Exclusion
2. Hold and Wait
3. No preemption
4. Circular wait

Deadlock Prevention

We can prevent Deadlock by eliminating any of the above four conditions.

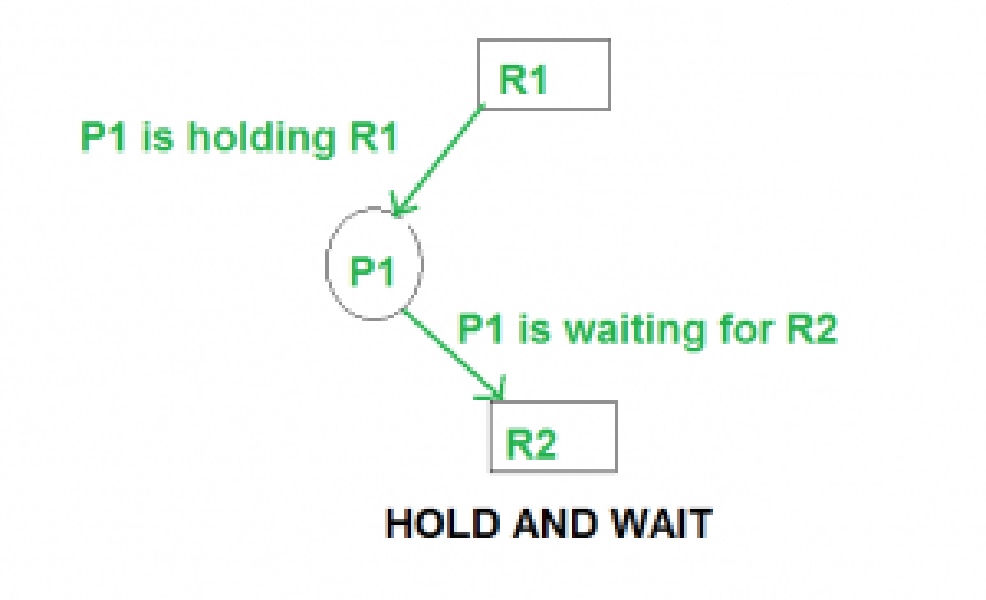
Eliminate Mutual Exclusion

It is not possible to dis-satisfy the mutual exclusion because some resources, such as the tape drive and printer, are inherently non-shareable.

Eliminate Hold and wait

1. Allocate all required resources to the process before the start of its execution, this way hold and wait condition is eliminated but it will lead to low device utilization. for example, if a process requires printer at a later time and we have allocated printer before the start of its execution printer will remain blocked till it has completed its execution.

2. The process will make a new request for resources after releasing the current set of resources. This solution may lead to starvation.



Eliminate No Preemption

Preempt resources from the process when resources required by other high priority processes.

Eliminate Circular Wait

Each resource will be assigned with a numerical number. A process can request the resources increasing/decreasing. order of numbering.

For Example, if P1 process is allocated R5 resources, now next time if P1 ask for R4, R3 lesser than R5 such request will not be granted, only request for resources more than R5 will be granted.

Deadlock Avoidance

Deadlock avoidance can be done with Banker's Algorithm.

Banker's Algorithm

Bankers's Algorithm is resource allocation and deadlock avoidance algorithm which test all the request made by processes for resources, it checks for the safe state, if after granting request system remains in the safe state it allows the request and if there is no safe state it doesn't allow the request made by the process.

Inputs to Banker's Algorithm:

1. Max need of resources by each process.
2. Currently, allocated resources by each process.
3. Max free available resources in the system.

The request will only be granted under the below condition:

1. If the request made by the process is less than equal to max need to that process.

2. If the request made by the process is less than equal to the freely available resource in the system.

Example:

```
Total resources in system:
A B C D
6 5 7 6



Available system resources are:
A B C D
3 1 1 2



Processes (currently allocated resources):
    A B C D
P1  1 2 2 1
P2  1 0 3 3
P3  1 2 1 0



Processes (maximum resources):
    A B C D
P1  3 3 2 2
P2  1 2 3 4
P3  1 3 5 0



Need = maximum resources - currently allocated resources.
Processes (need resources):
    A B C D
P1  2 1 0 1
P2  0 2 0 1
P3  0 1 4 0
```

Note: Deadlock prevention is more strict than Deadlock Avoidance.