KARNAVATI UNIVERSITY

Operating Systems

Course Code: **71203002004**

Introduction to Deadlocks

by -

Minal Rajwar

Deadlock in Operating System

A **deadlock** is a state where processes are permanently stuck because each is waiting for a resource held by another, and none can proceed.

How Deadlock Occurs

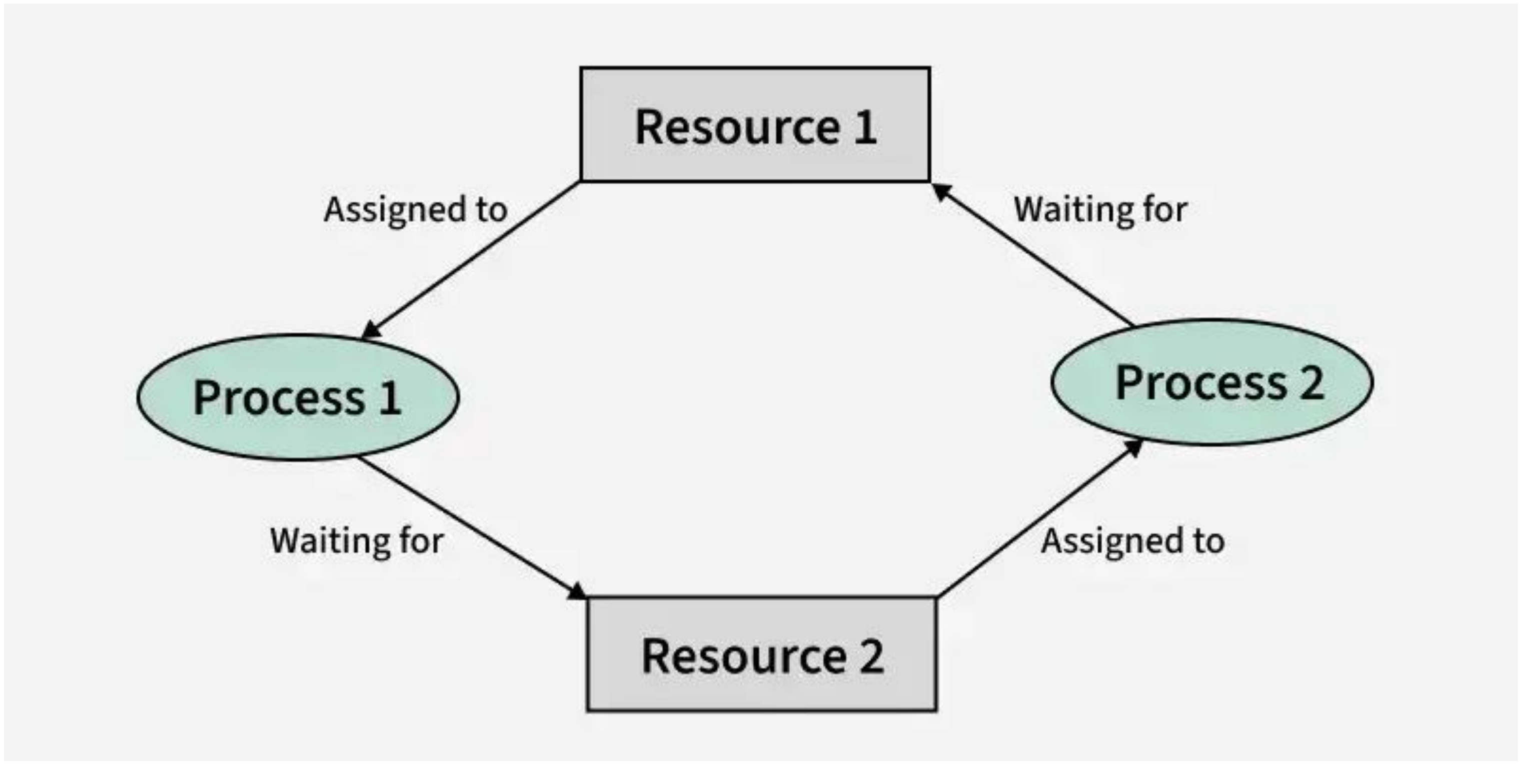
Processes generally:

1. Request a resource
2. Use the resource
3. Release the resource

Deadlock occurs if processes hold some resources while waiting for others.

Example:

- P1 holds R1 and requests R2.
- P2 holds R2 and requests R1.
  → Both are stuck.



Real-Life Examples

- Two processes each hold one tape drive and need another.
- Two processes wait on semaphores A and B, blocking each other.
- Processes request memory in a sequence where each waits for more, causing a deadlock.

How Deadlock Occurs

Consider **two processes (P1, P2)** and **two resources (R1, R2):**

1. P1 acquires R1.
2. P2 acquires R2.
3. P1 now requests R2 (held by P2).
4. P2 requests R1 (held by P1).

How Deadlock Occurs…

Both processes are stuck:

- P1 waits for R2 from P2.
- P2 waits for R1 from P1.

Since neither can release its resource before finishing, both are in **deadlock**.

To resolve this, the OS may intervene (e.g., terminate a process).
Deadlocks waste resources and reduce system performance, so **detection, avoidance, and prevention algorithms** are used.

Necessary Conditions for Deadlock

- **Mutual Exclusion** – Only one process can use a resource at a time.
- **Hold and Wait** – A process holds one resource and waits for others.
- **No Preemption** – Resources can’t be forcibly taken; they must be released voluntarily.
- **Circular Wait** – A cycle of processes each waiting for a resource held by the next.

Mutual Exclusion

The **mutual exclusion** condition means at least one resource must be non-shareable—only one process can use it at a time.

Example:

- Process 1 acquires Resource 1.
- Process 2 acquires Resource 2.

Both resources are held exclusively, so no other process can access them simultaneously.

Hold & Wait

A process holds at least one resource while waiting for another.

Example:

- Process 1 holds Resource 1 and waits for Resource 2.
- Process 2 holds Resource 2 and waits for Resource 1.

Both processes are holding one resource and waiting for the other.

No Preemption

The **no preemption** condition states that resources cannot be forcibly taken from a process; they must be released voluntarily after the process finishes.

Example:

- P1 holds Resource R1 and requests R2 (held by P2).
- P1 cannot take R2 until P2 finishes and releases it.
- P1 may later retry requesting both R1 and R2.

This restriction can sometimes lead to other issues like **live lock**.

What is Live Lock?

A **live lock** occurs when processes keep changing their states in response to each other but make no real progress.

Example:

- P1 acquires R1 and needs R2.
- P2 acquires R2 and needs R1.
- Both detect the conflict, release their resources, and try again.
- They repeat this cycle endlessly—resources keep switching, but execution never proceeds.

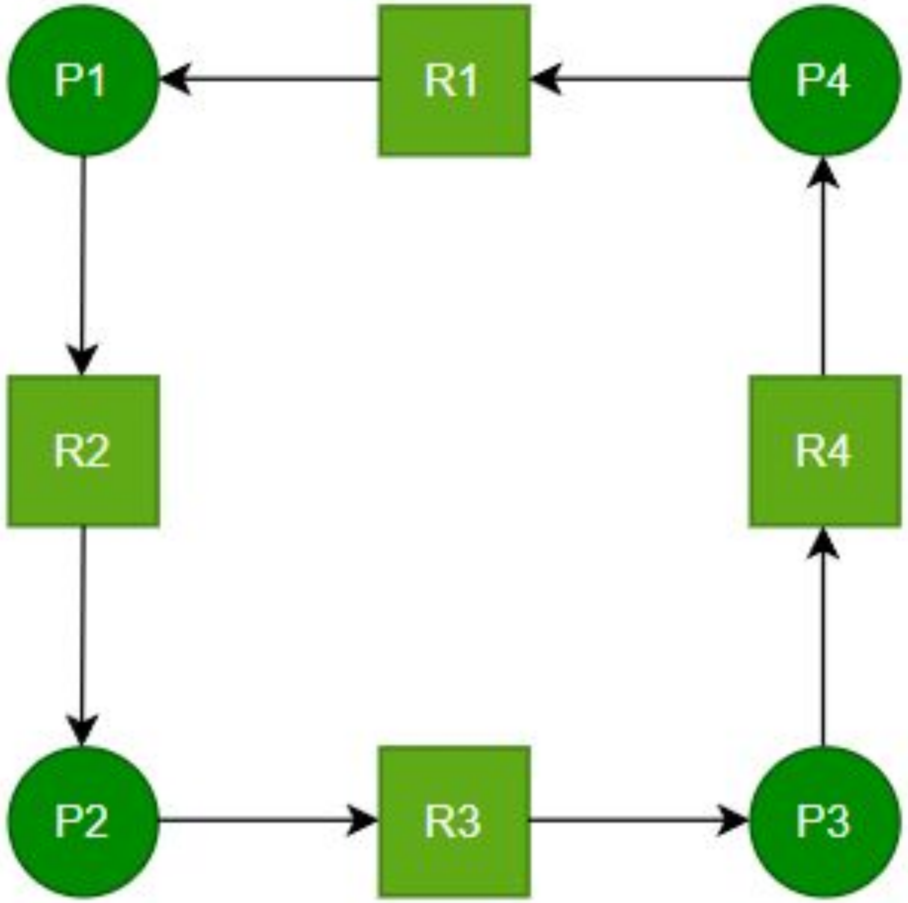
Circular Wait

Circular wait is a condition in which a set of processes are waiting for resources in such a way that there is a circular chain, with each process in the chain holding a resource that the next process needs. This is one of the necessary conditions for a **deadlock** to occur in a system.

Example: Imagine four processes—**P1**, **P2**, **P3**, and **P4**—and four resources—**R1**, **R2**, **R3**, and **R4**.

- **P1** is holding **R1** and waiting for **R2** (which is held by P2).
- **P2** is holding **R2** and waiting for **R3** (which is held by P3).
- **P3** is holding **R3** and waiting for **R4** (which is held by P4).
- **P4** is holding **R4** and waiting for **R1** (which is held by P1).

Circular Wait



DISCUSSION & REVISION

1. Which condition states that only one process can use a resource at a time?
2. In which condition does a process hold one resource while waiting for another?
3. What do we call it when resources cannot be forcibly taken from a process?
4. What condition occurs when processes wait on each other in a cycle?
5. What is the situation called when processes keep changing state without progress?

REFERENCES

- https://www.geeksforgeeks.org/operating-systems/introduction-of-deadlock-in-operating-system/
- https://www.geeksforgeeks.org/operating-systems/conditions-for-deadlock-in-operating-system/
- https://www.baeldung.com/cs/os-deadlock