Java Threads →

Deadlocks, Livelocks & Starvation

Resources:

- https://www.baeldung.com/cs/deadlock-livelock-starvation

ada.diaconescu@telecom-paris.fr

Competition for Resources

- More than one process may compete for a shared set of resources

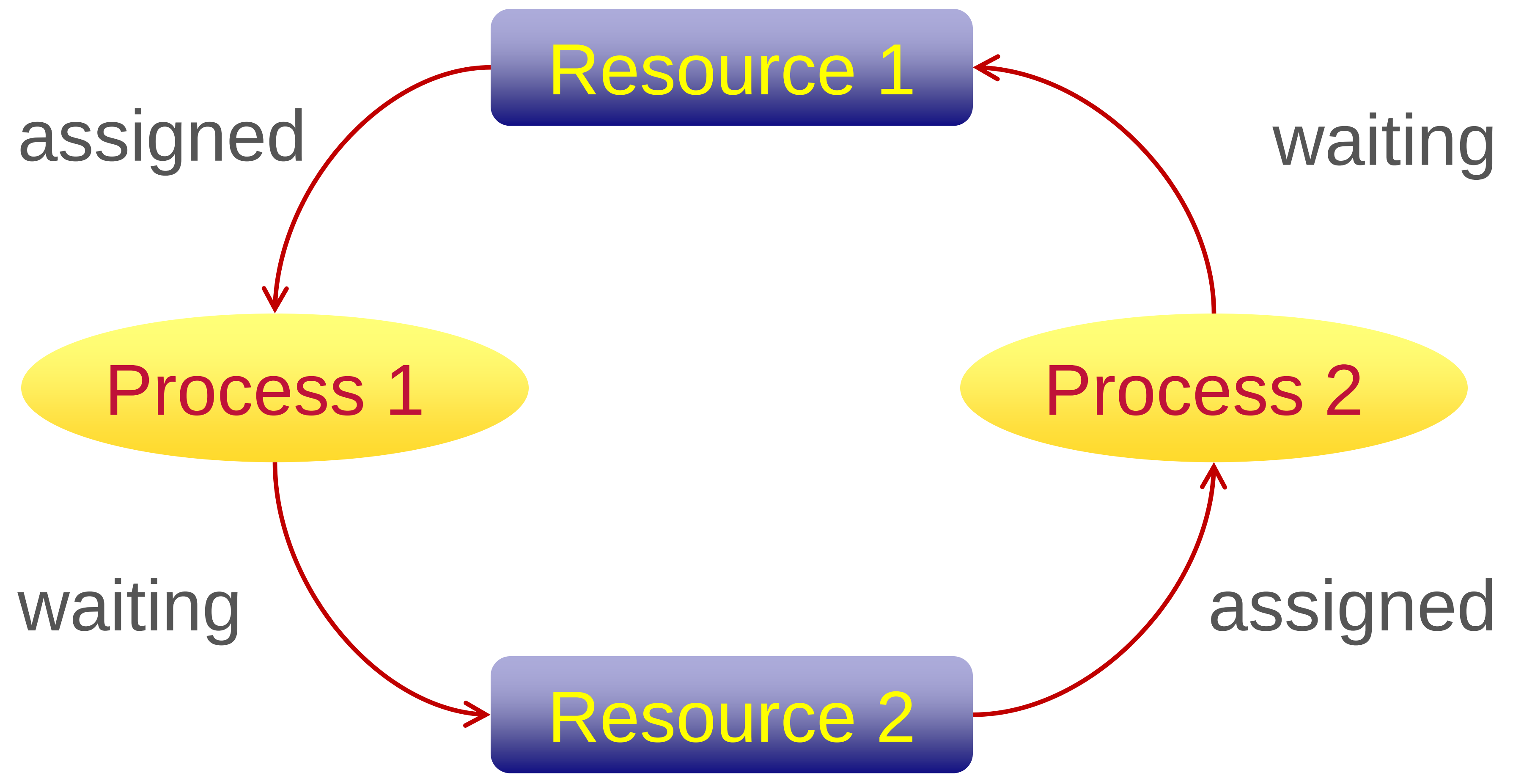
- If a process requests a resource that is unavailable

  then the process waits for the resource to become available

- In some cases, a process may never succeed to get access to the resource
  - Deadlock
  - Livelock
  - Stravation

Deadlock – what is it?

- Processes block each other in a circular manner due to resource acquisition
- None of the processes can make any progress
  as they wait for a resource that is held by another process



Deadlock – when can it happen?

The following four conditions must hold simultaneously:

- Mutual Exclusion:
  - at least one resource must be held by a process; all other processes must wait for it
- Hold & Wait:
  - A process must hold one resource
    and ask for another resource that is held by other processes
- No Pre-emption:
  - A resource cannot be released by force from a process; the process must release it voluntarily
- Circular Wait:
  - A set of processes p0, p1, ..pn are in a state where p0 is waiting for a resource held by p1; p1 waits for a resource held by p2; ...; pn waits for a resource held by p0.

ada.diaconescu@telecom-paris.fr

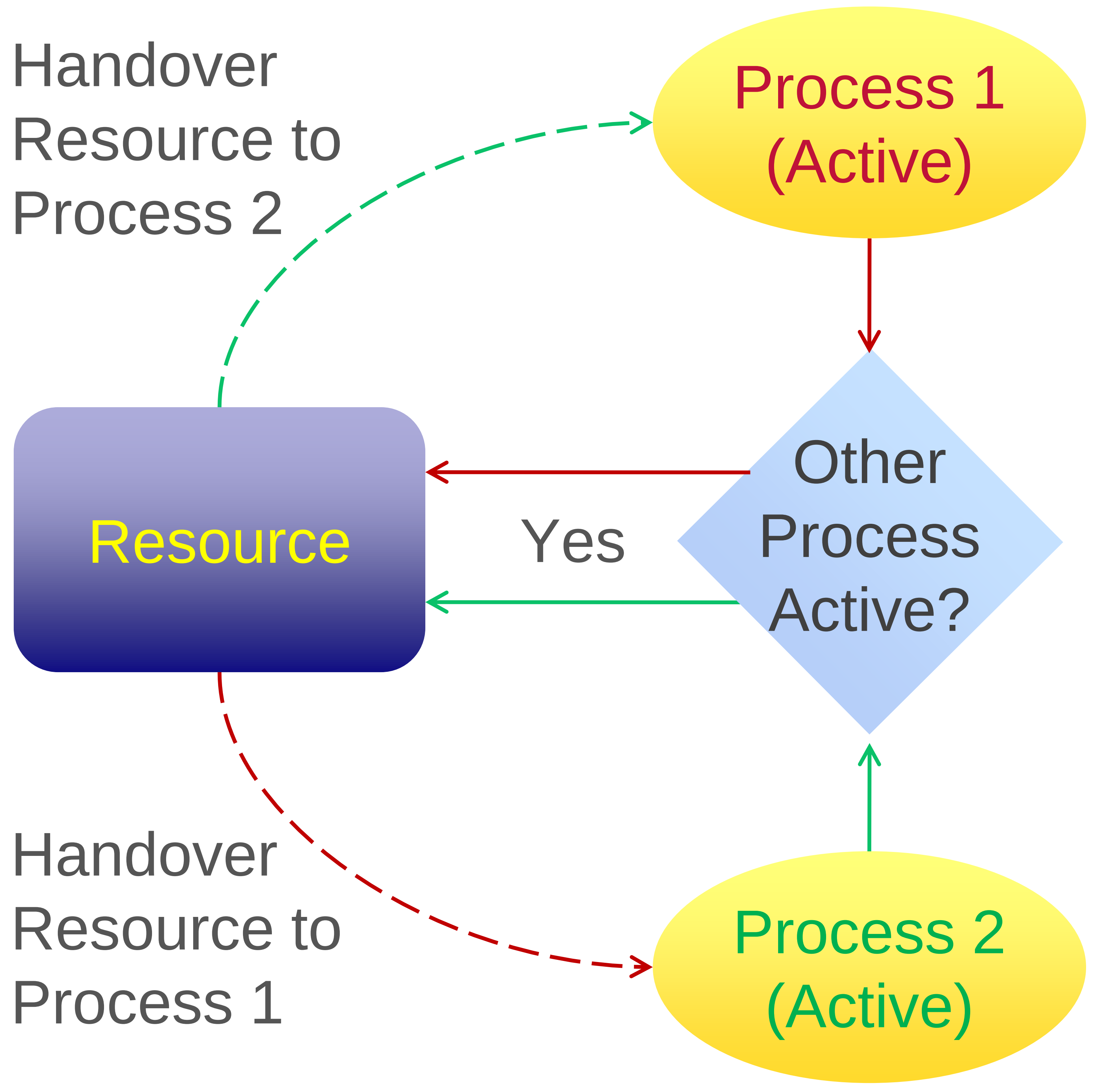
Deadlock – how to prevent it?

Prevent at least one of the four conditions presented in the previous slide

- Mutual Exclusion:
  - E.g., read-only resources can be shared by several processes simultaneously
- Hold & Wait:
  - E.g., avoid asking for a resource when already holding another resource; Ask for all the resources at once instead
- No Pre-Emption:
  - E.g., a process releases its held resources if the new ones it asks for are unavailable
- Circular Wait:
  - E.g., impose a total ordering of resource demands (ask for r1 first then for r2)

Live lock

- The states of the processes involved change continuously
- The processes depend on each other and can never progress



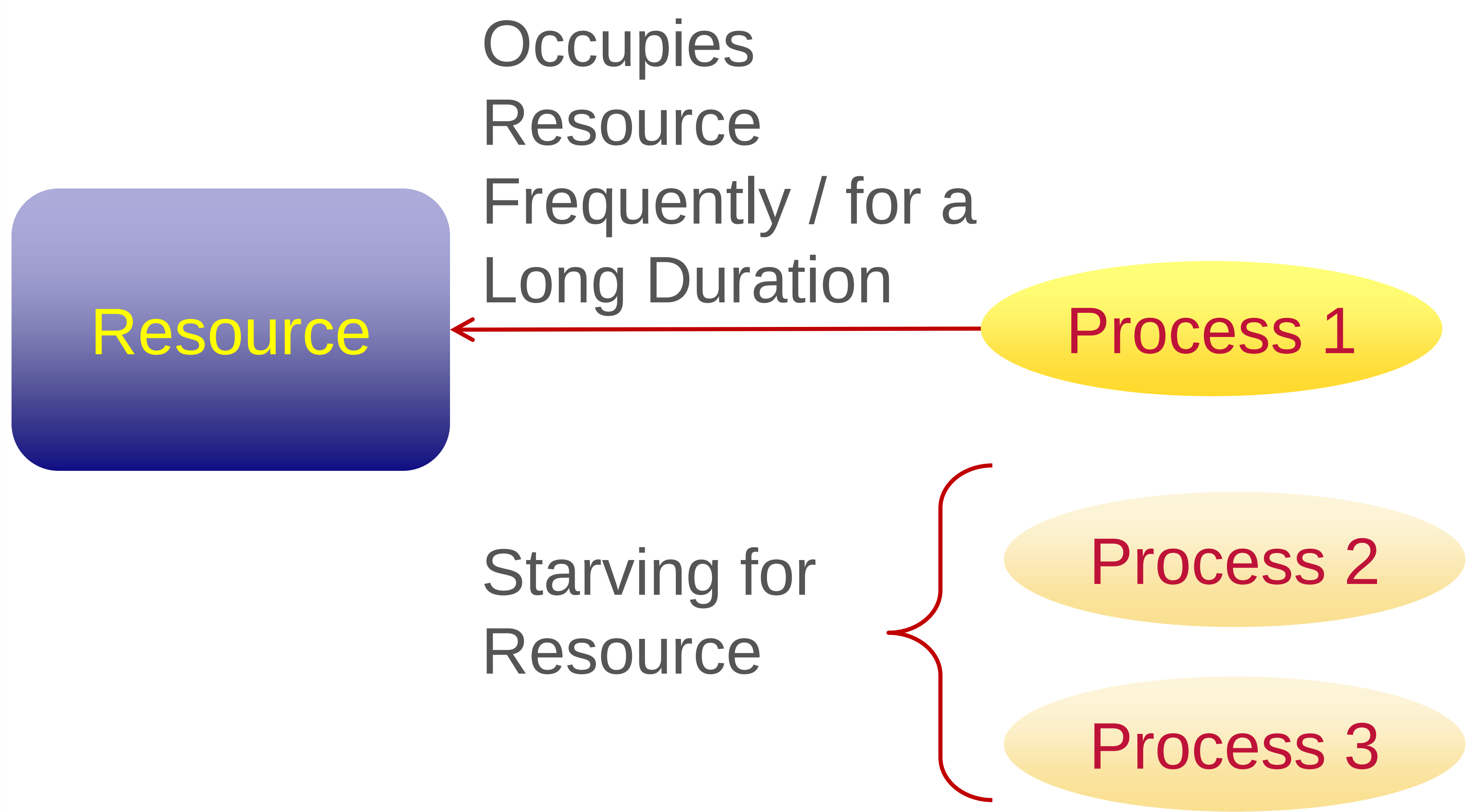
Example:

- Process 1 & Process 2 need a shared Resource
- Each process checks whether the other is in an Active state
- If so then it hands over the Resource to the other process
- As both processes are Active, they keep handing over the Resource to each other indefinitely
- The processes aren't blocked but don't make any progress

ada.diaconescu@telecom-paris.fr

Process Starvation

- The outcome of a process that is unable to gain access to a required resource
  - The process cannot make any progress
- Causes: deadlock, live lock, or a 'greedy' process holding the resource



Avoidance techniques:

- Priority queue that uses aging technique:

  → The more a process waits the higher its priority in the queue

- Round-robin access pattern

  → The resource is allocated fairly to each process