Deadlocks

- Detection & Recovery

① How?

② How?

→ Graph

(Process & Resources

→ Resource Graph

or

Resource Allocation Graph (RAG)

Methodology

Maintain RAG ① Construct RAG — How?

└ Updates.

Detection ② At any time instance check RAG has a cycle.

Recovery ③ Recovery - How to break the cycle?

└ 2 methods

RAG (Resource Allocation Graph)

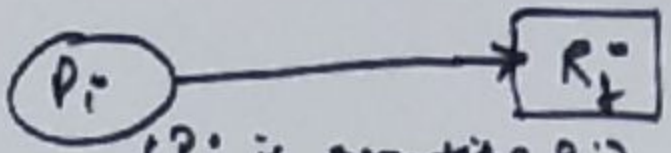
- Two elements → Node
  - → Edge
- Types of Nodes
- " " Edges
- Rules for Edges

→ Two Types

- Process Node → (Pi)
- Resource Node → Number → [Ri]
  - → Types of Resources
    - Ways to represent Resource Nodes
      - Manual ← eg 10 printers : [R1] .... [R10]
      - Automated → [⋮⋮⋮] R ✓

Types of Edges

↳ Connection — Directional between process & Resource Nodes



Request edge

(P_i is requesting R_j)

Assignment edge

(R_j is assigned to P_i)

Allocation edge.

Cycle — Across All P_i & R_j

Updation of RAG

- Process node gets deleted, on process termination.
- Resource node & process node assignment edges get deleted after resource usage.
- If a resource is removed from the system, the corresponding resource node is removed from RAG.

— Recovery

Process based (or) Resource based

— On detection of a cycle in RAG,

P_i, P_j, P_k — killing a process

Two ways ① Kill all process in the cycle ✗

② Incremental kill / destroy
Give a criteria to select a process to kill

repeated.

— Resource based — ① Remove Request / Allocation edges

② Remove Assigned edge. is incremental

— In case of assignment edges,
- a process that has a resource assigned, but has not started using the resource, the removal of that assignment edge is preferred.

Incremental = Ensuring that a process has minimal loss of computation and resource usage
⇒
(preferred)