Deadlock

General Definition

- A set of processes S is *deadlocked* if each process in S is waiting for an event that can be caused only by a process in S.

Resource Control

- Process waits for one or more resources held by others
- Necessary condition for deadlock:
  - **Mutual exclusion**
  - **Hold-and-wait**
  - **No preemption**
  - **Circular wait**

Principles of dealing with deadlock

- Deadlock *prevention*
- Deadlock *avoidance*
- Deadlock *detection* and *recovery*
- Ignore (hope for the best)

Deadlock Prevention

Idea

- To introduce *structural restrictions* that eliminates deadlock risk

Methods

- Mutual exlusion
  Enable simultaneous use, e.g. by spooling [not general]
- Hold-and-wait
  Reserve all resources at once [low utilization, risk of starvation]
- No-preemption
  Allow preemption, e.g. of CPU and memory [not general]
- Circular wait
  Assign ranks to resource types:

  A process may only request resources having strictly higher rank than already allocated ones.

Deadlock Avoidance

Idea

- To use *behavioural information* to dynamically avoid deadlock.

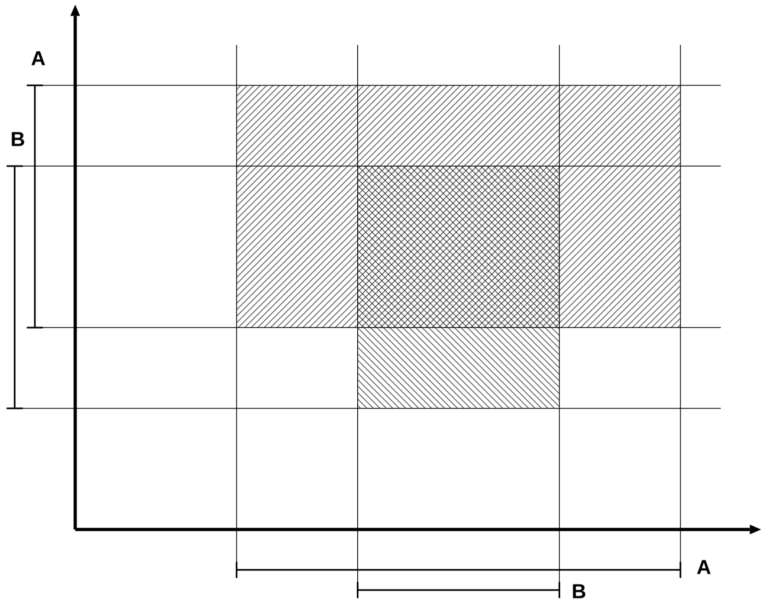
Prerequisites

- Behavioural information must be available for all processes
- Examples:
  - Max resource claim for each resource type
  - Resource usage pattern

Method

- A *safe* state is a state from which there exists a way to terminate all processes (according to usage information).
- **Banker's Algorithm** A resource request is granted only if the resulting state remains safe.

Safe/unsafe States



Deadlock Detection

Idea

- To detect deadlocks and handle them by automatic recovery

Deadlock Detection

- Maintain global allocation state and perform deadlock detection:
  - Regularly
  - When some process seems not to make progress
- Assume deadlock if no progress for a while

Recovery

- Select one or more victims based on *cost factors*
- Kill victim or *roll-back* to *check-point*
- Risk of starvation