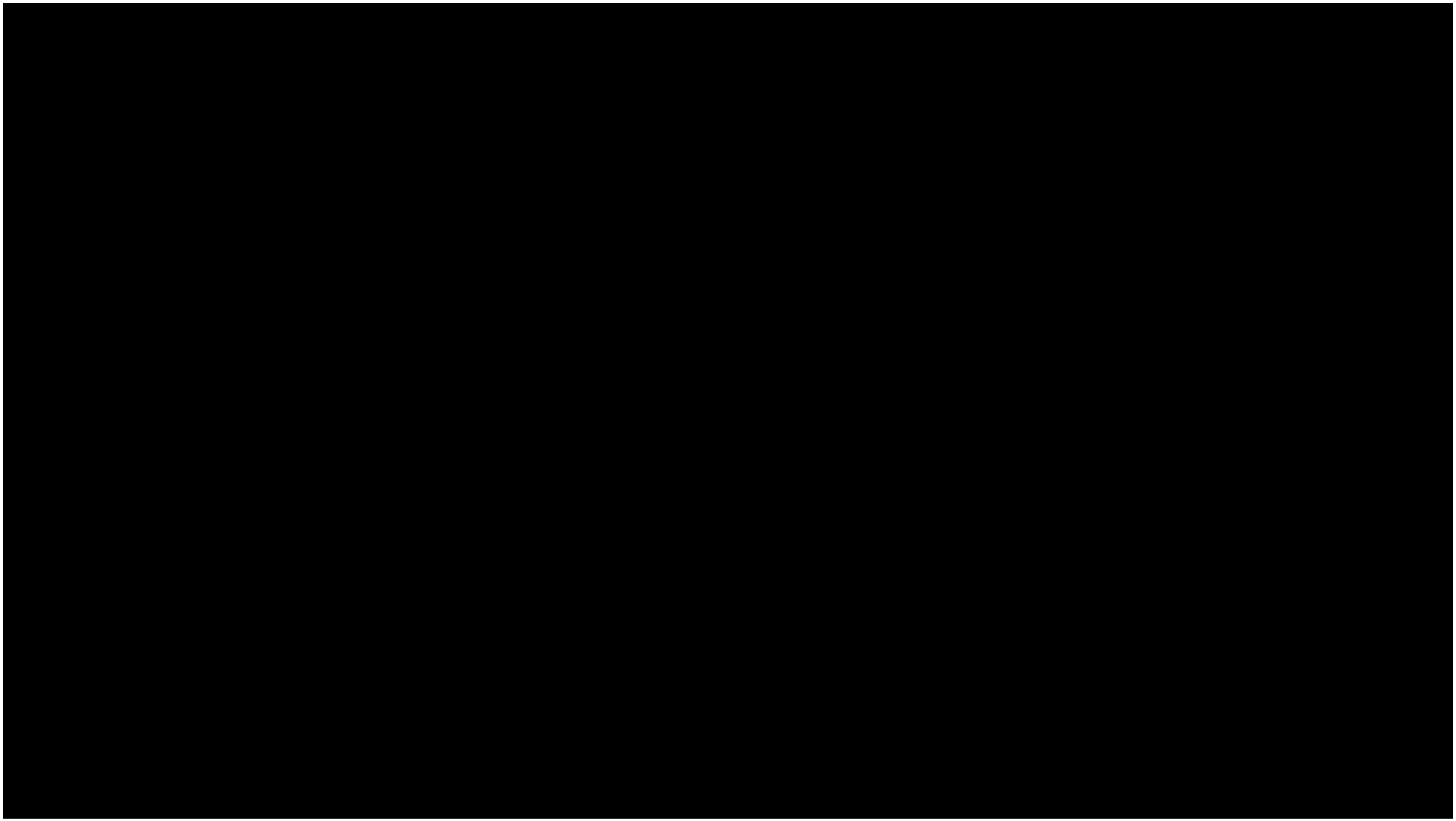


CS3844 Operating Systems

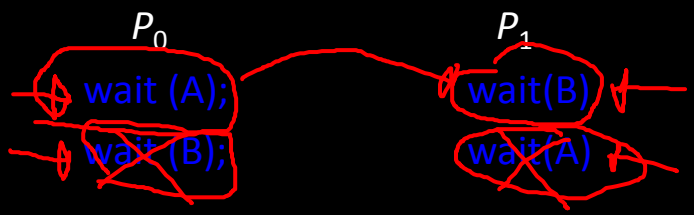
Constructing Resource Allocation Graphs

Objectives

- Construct a resource allocation graph.
- List the conditions necessary for a deadlock to occur.

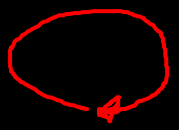
The Deadlock Problem

- A set of blocked processes each holding a resource and waiting to acquire a resource held by another process in the set
- Example
  - System has 2 disk drives
  - P_1 and P_2 each hold one disk drive and each needs another one
- Example
  - semaphores *A* and *B*, initialized to 1



Deadlock Characterization

- **Mutual exclusion:**
  - only one process at a time can use a resource
- **Hold and wait:**
  - a process holding at least one resource is waiting to acquire additional resources held by other processes
- **No preemption:**
  - a resource can be released only voluntarily by the process holding it, after that process has completed its task
- **Circular wait:**
  - there exists a set $\{P_0, P_1, \ldots, P_0\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by P_2, …, P_{n-1} is waiting for a resource that is held by P_n, and P_0 is waiting for a resource that is held by P_0.

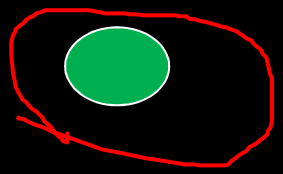


Modeling Deadlocks using Discrete Math

- Vertices represent processes and resources
  - $P = \{P_1, P_2, \ldots, P_n\}$, the set consisting of all the processes in the system
  - $R = \{R_1, R_2, \ldots, R_m\}$, the set consisting of all resource types in the system
- request edge – directed edge $P_1 \rightarrow R_j$
- assignment edge – directed edge $R_j \rightarrow P_i$

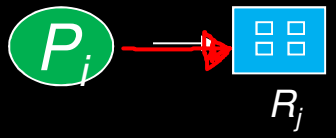
Resource-Allocation Graph

- Process



Resource Type with 4 instances

- P_i requests instance of R_j



- P_i is holding an instance of R_j

Resource Allocation Graph Example