ELEC 377 – Operating Systems

Week 4 – Class 3

Last Class

- Scheduling Criteria
- Algorithms
- Evaluation
- Sample Evaluations

Today

- Quiz 1 Review
- Continue Scheduling
- Deadlock

Quiz 1

- Process
  a program in execution and all of the resources associated with the executing instance of the program such as memory, registers, open files, etc.
- Cache
  high speed memory between the processor and main memory, alternatively, main memory that has been reserved for a local copy of information on the hard drive

Quiz 1

- Context Switch
  - changing the running process by saving the registers of the current process and restoring the registers of the new process.

Quiz 1

- User and Kernel Mode
  The modes are distinguished by a bit in one of the registers of the CPU. All of the processes operate in User mode, while the Operating system code executes in Kernel Model. The mode is changed by an Interrupt or a trap. Modes are used to protect the OS by allowing only code running in Kernel mode to execute I/O instructions or to change memory management or timer registers.

Quiz 1

- PCB
  - The PCB is used to store all information that is specific to a given process. Each process has its own PCB.

Information in a PCB includes (4 of 7)

1. Queuing information
2. Process ID
3. Status
4. Registers
5. Memory Management information
6. Open files
7. Accounting information

Quiz 1

- Interrupt and Trap
  Interrupts are generated by hardware devices, traps are generated by the process (e.g. devision by zero, system call). Traps are used to implement system calls

Quiz 1

- Three Criteria
  Mutual Exclusion - only one

  Progress - if there is no process in a critical section, and more than one process want to enter their critical section, then the selection of a process cannot be postponed indefinitely
  Bounded Waiting - once a process is waiting, the other processes can only enter and leave a bounded number of times (no starvation)

Algorithm Evaluation

- Queuing Models

◊ each component of the OS is a server with an entry queue
- cpu (with ready queue), IO devices with device queues

◊ probability distributions for each request
- hyper exponential distribution for burst time
- distributions for I/O devices

◊ n = λ ✖ W (Little's formula)

n = average number of processes

λ = average arrival

W = average wait time

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

Algorithm Evaluation

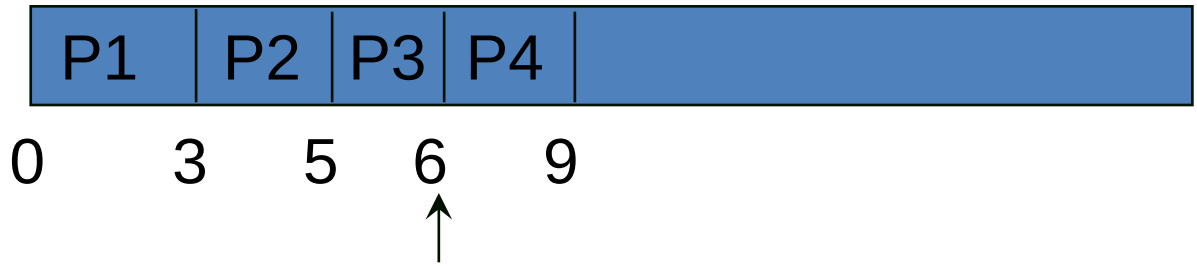
- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

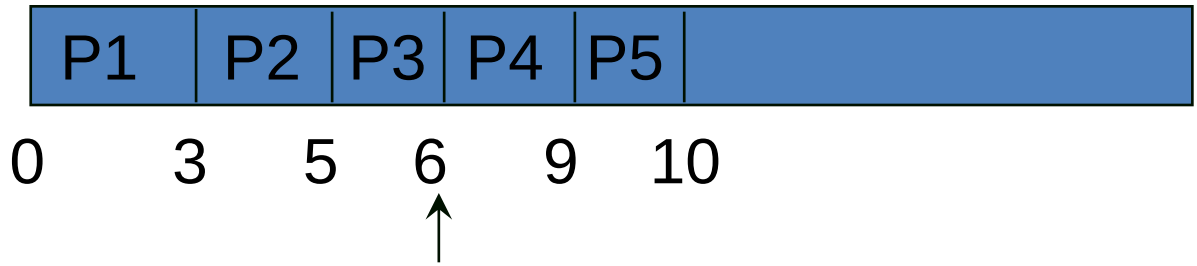
Round Robin scheduling, q=3:



Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

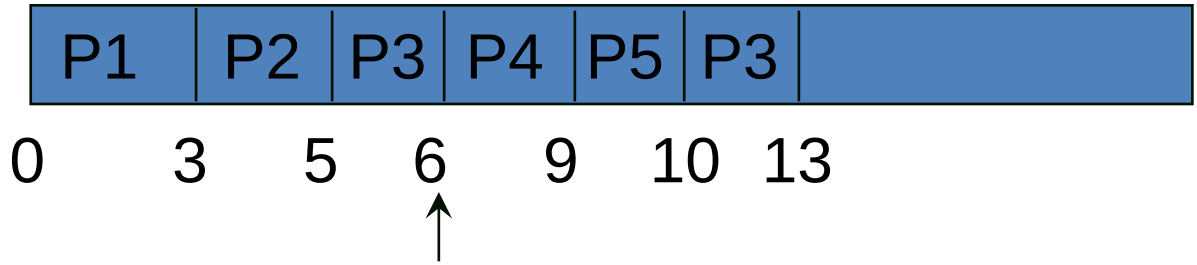
Round Robin scheduling, q=3:



Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

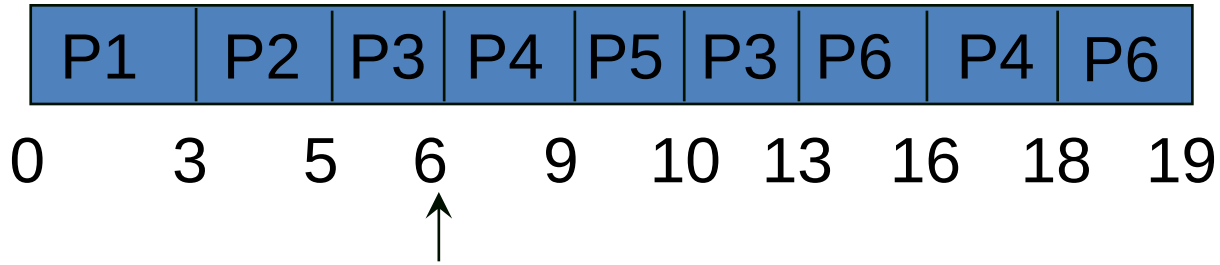
Round Robin scheduling, q=3:



Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

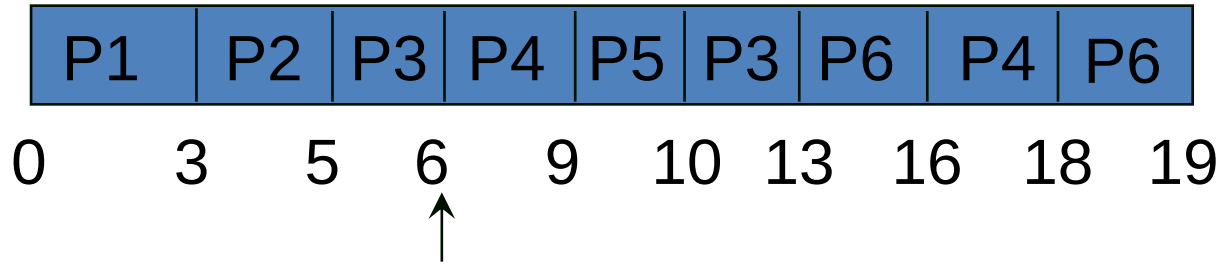
Round Robin scheduling, q=3:



Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:

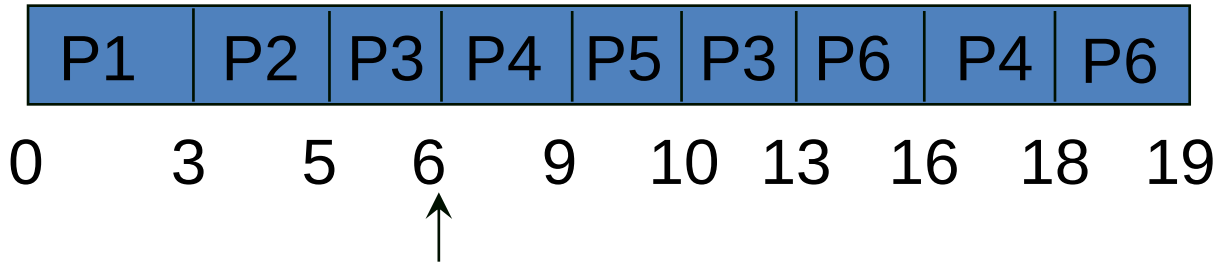


- Waiting time:
  - P1 - 0ms, P2 - 3ms, P3 - 9 ms, P4 - 13ms, P5-9, P6-9
- average = 43/6 = 7.66 ms

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms
  P6 becomes available at time 6 with 4ms

Round Robin scheduling, q=3:



- Waiting time:
  - P1 - 0ms, P2 - 3ms, P3 - 9 ms, P4 - 13ms, P5-9, P6-9
- average = 43/6 = 7.16 ms
- Turnaround
  - P1 - 3ms, P2-5ms, P3 - 13 ms, P4 - 18ms, P5-10,P6-13
- average = 62/6 = 10.333 ms

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling

0

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling

Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling

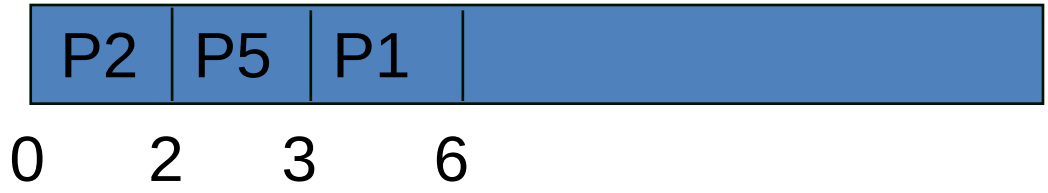
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



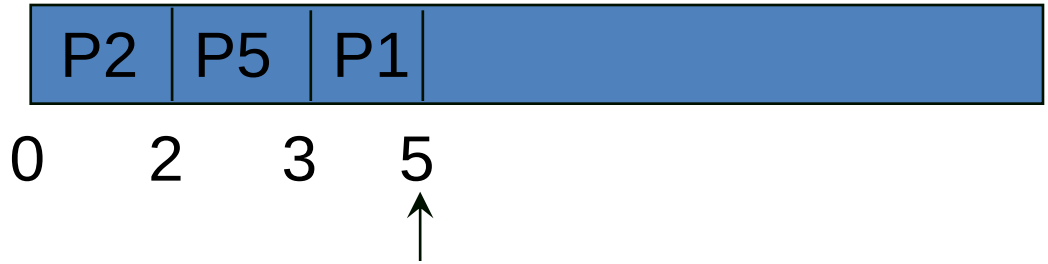
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



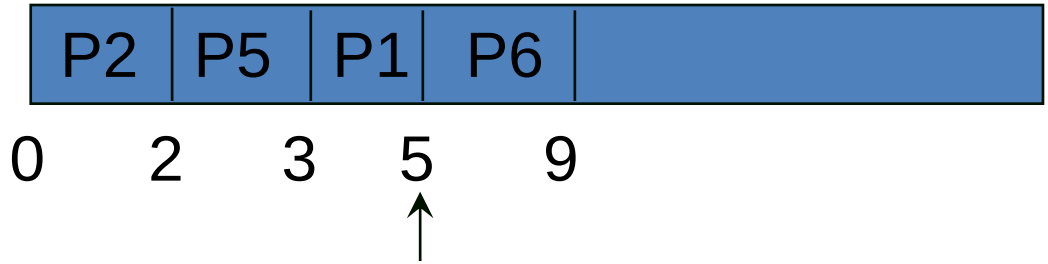
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



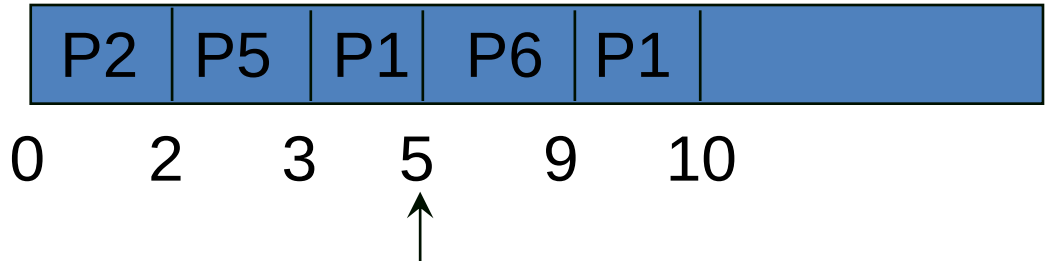
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



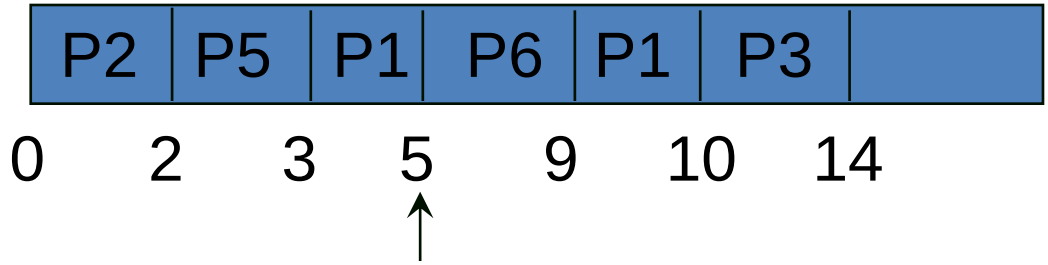
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



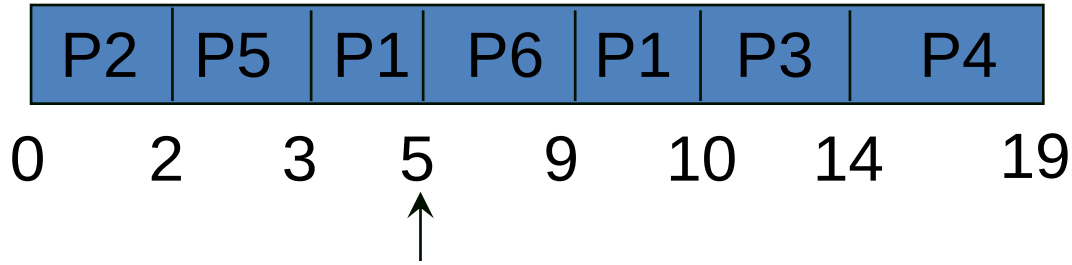
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



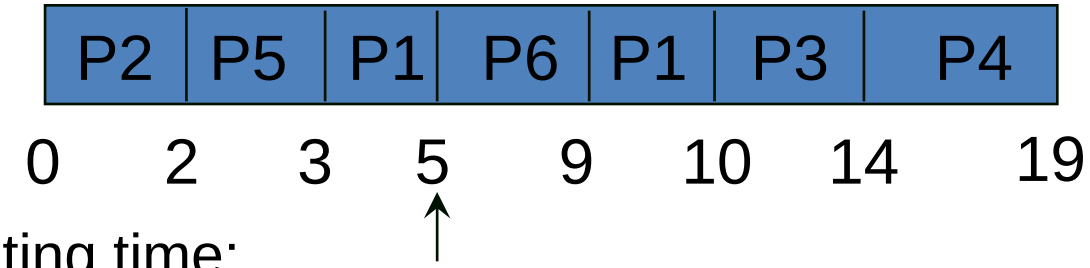
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

Preemptive Priority Scheduling



- Waiting time:
  - P1 - 7ms, P2 - 0ms, P3 - 10 ms, P4 - 14ms, P5-2, P6=0
- average = 33/6 = 5.5 ms

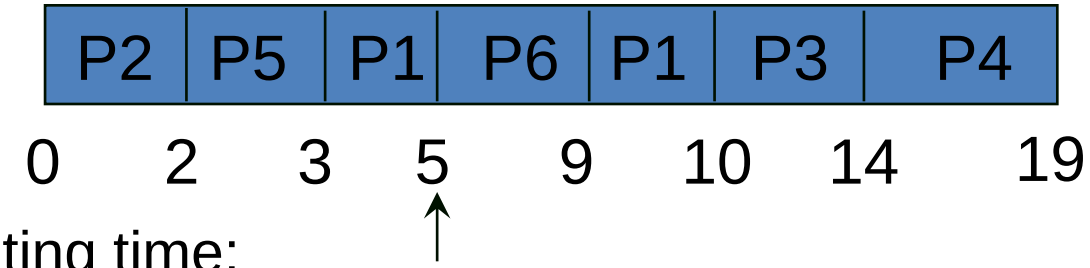
Algorithm Evaluation

- P1 - 3ms, P2 - 2ms, P3 - 4 ms, P4 - 5 ms, P5 - 1ms

pri 3 1 4 5 2

P6 becomes available at time 5 with 4ms, pri 2

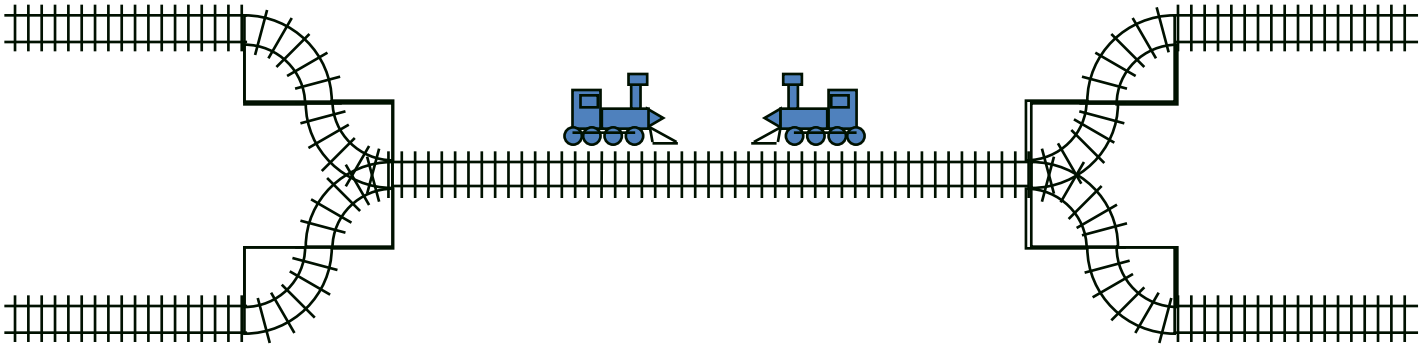
Preemptive Priority Scheduling



- Waiting time:
  - P1 - 7ms, P2 - 0ms, P3 - 10 ms, P4 - 14ms, P5-2, P6=0
- average = 33/6 = 5.5 ms
- Turnaround
  - P1 - 10ms, P2 - 2ms, P3 - 14 ms, P4 - 19ms, P5-3, P6=4
- average = 52/6 = 8.66 ms

What is Deadlock?

- A set of process, each holding a resource that another process in the set needs



- Common track is a resource
- Starvation
- rollback?

System Model

- Resource Types R_1, R_2, …, R_η
  - Each resource has a number of instances (W_i)
  - Resource instances are indistinguishable – doesn't matter which one you get.

- Process resource protocol
  - request
  - use
  - release

Deadlock Conditions

- four conditions necessary for deadlock:
  - ◊ **mutual exclusion**: only a limited number (usually one) process at a time can use a resource

Deadlock Conditions

- four conditions necessary for deadlock:
  - ◊ **mutual exclusion**: only a limited number (usually one) process at a time can use a resource
  - ◊ **hold and wait**: a process has (at least) one resource and is waiting for another

Deadlock Conditions

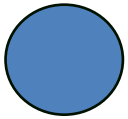
- four conditions necessary for deadlock:
  - ◊ **mutual exclusion**: only a limited number (usually one) process at a time can use a resource
  - ◊ **hold and wait**: a process has (at least) one resource and is waiting for another
  - ◊ **no preemption**: we can’t take a resource away from a process

Deadlock Conditions

- four conditions necessary for deadlock:
  - ◊ **mutual exclusion**: only a limited number (usually one) process at a time can use a resource
  - ◊ **hold and wait**: a process has (at least) one resource and is waiting for another
  - ◊ **no preemption**: we can't take a resource away from a process
  - ◊ **circular wait**: P_0 waits for a resource held by P_1, which waits for a resource held by P_2, … P_n, which waits for a resource held by P_0

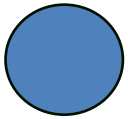
Resource Allocation Graph

- Process

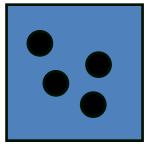


Resource Allocation Graph

- Process



- Resource Type
  - ◊ 4 instances

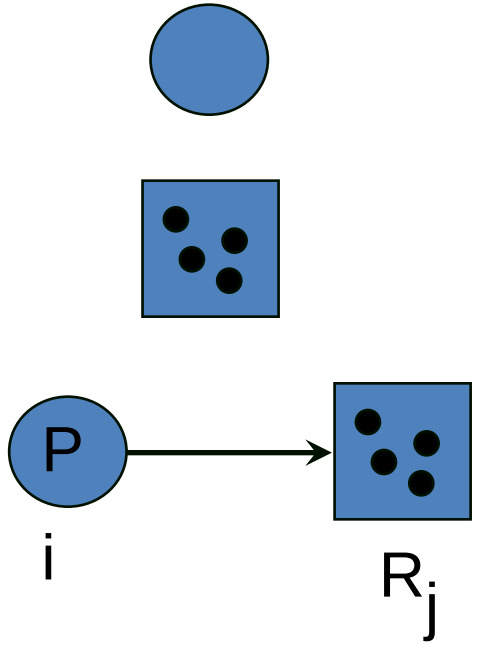


Resource Allocation Graph

- Process

- Resource Type
  - ◊ 4 instances

- P_i requests an instance of R_j



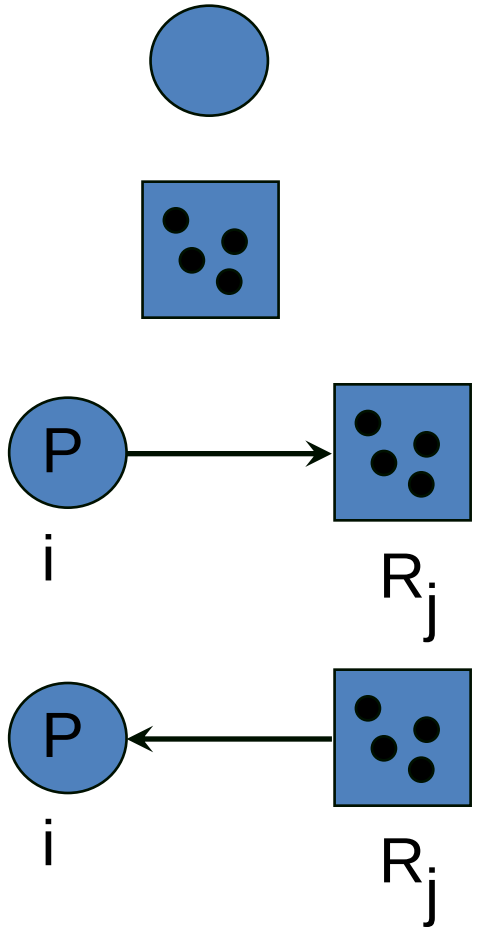
Resource Allocation Graph

- Process

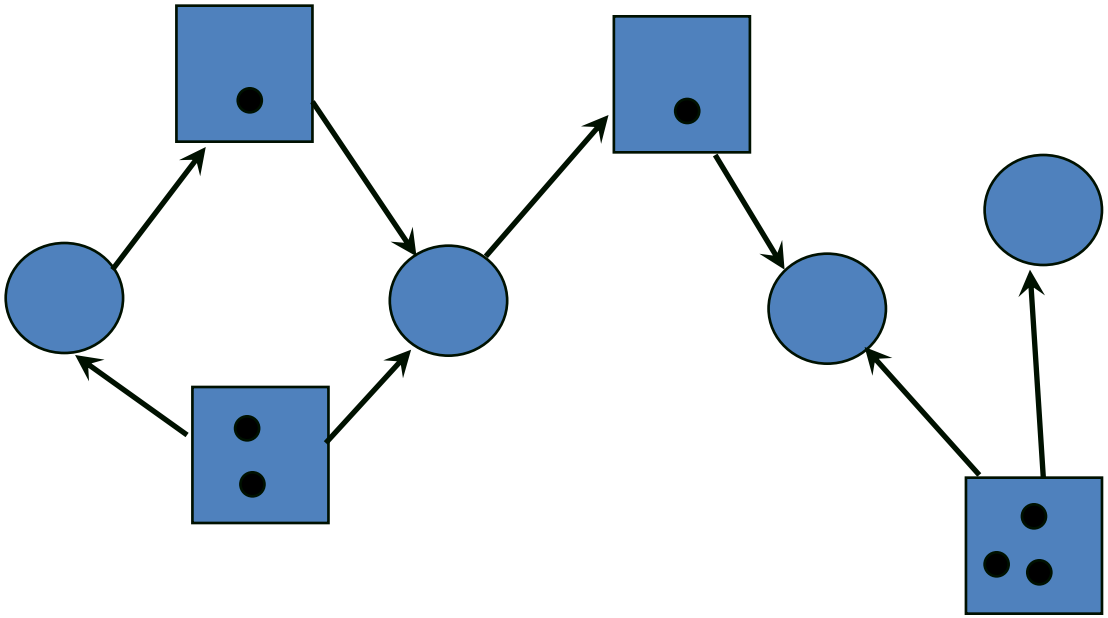
- Resource Type
  - 4 instances

- P_i requests an instance of R_j

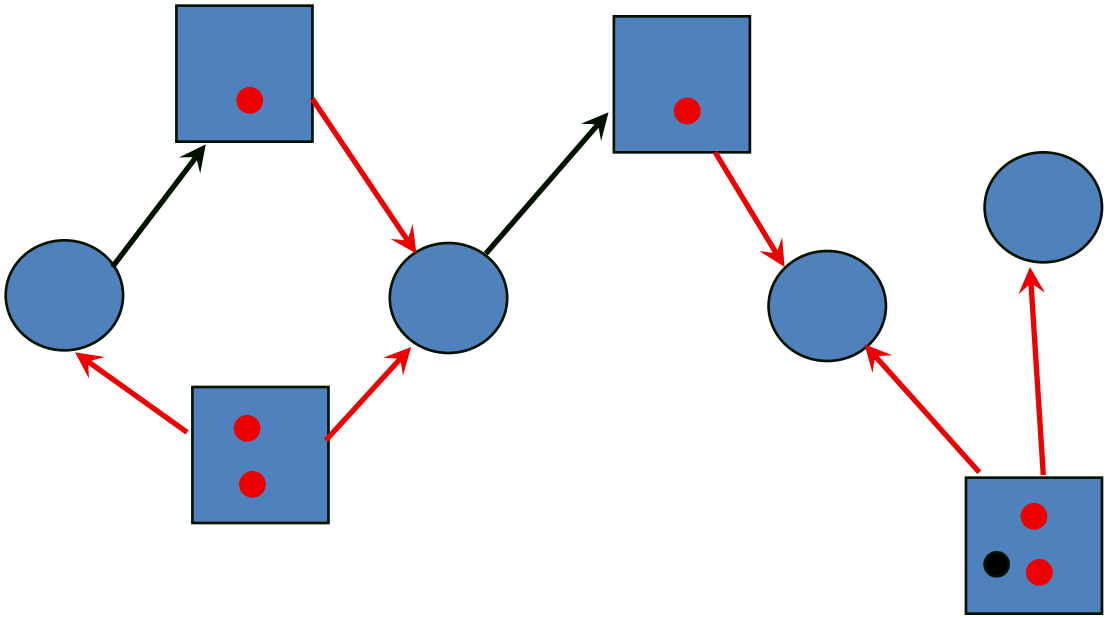
- P_i holds an instance of R_j



Resource Allocation Graph Example

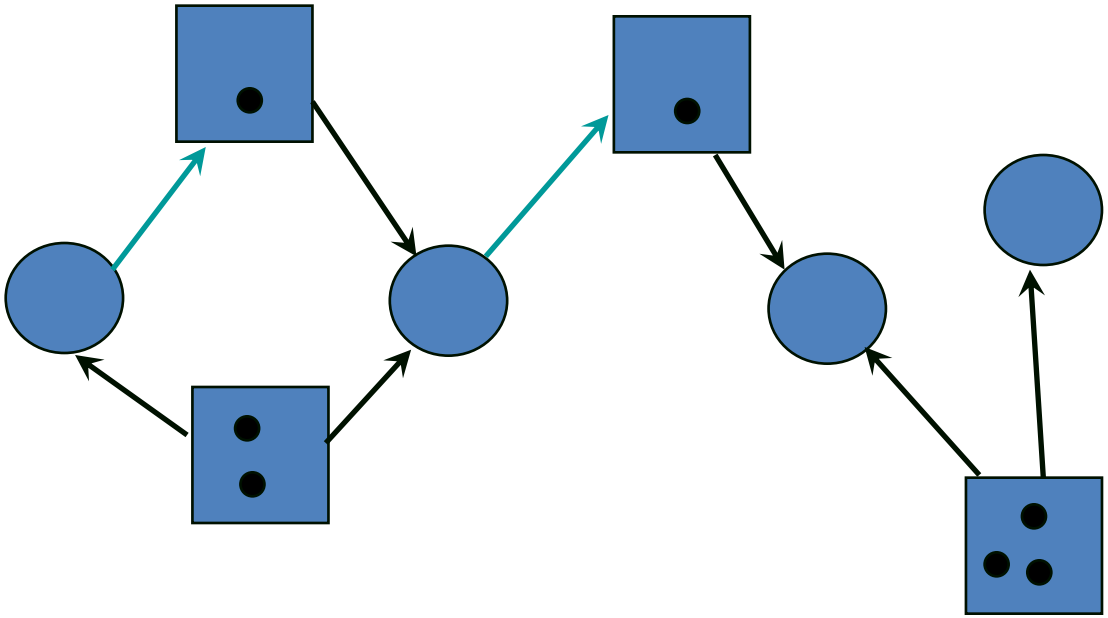


Resource Allocation Graph Example



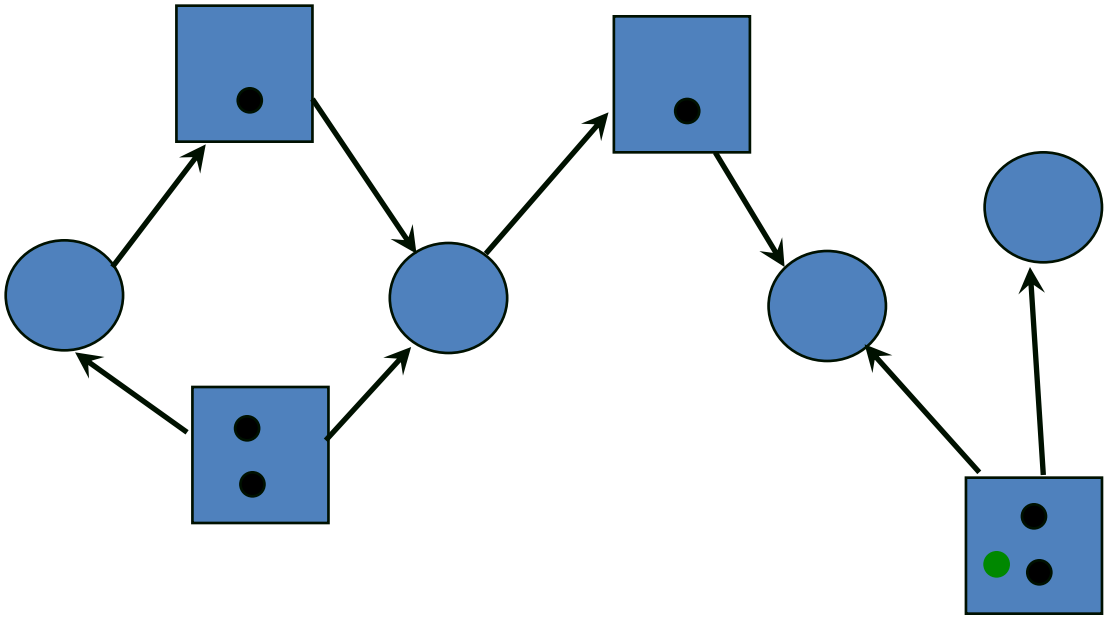
Allocated Resources

Resource Allocation Graph Example



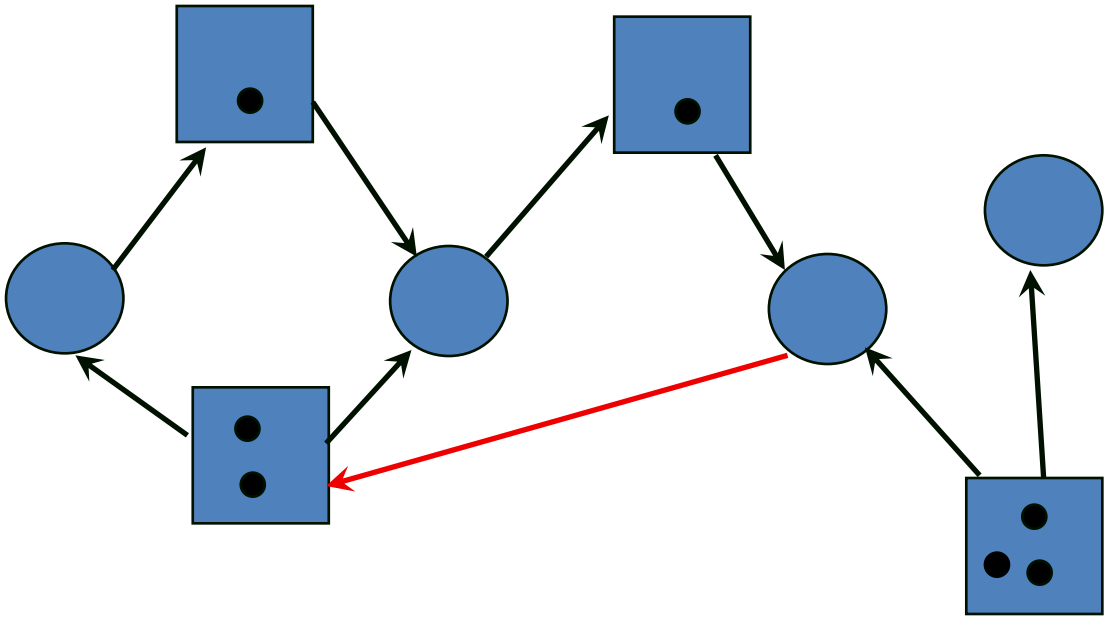
Resource
Requests

Resource Allocation Graph Example

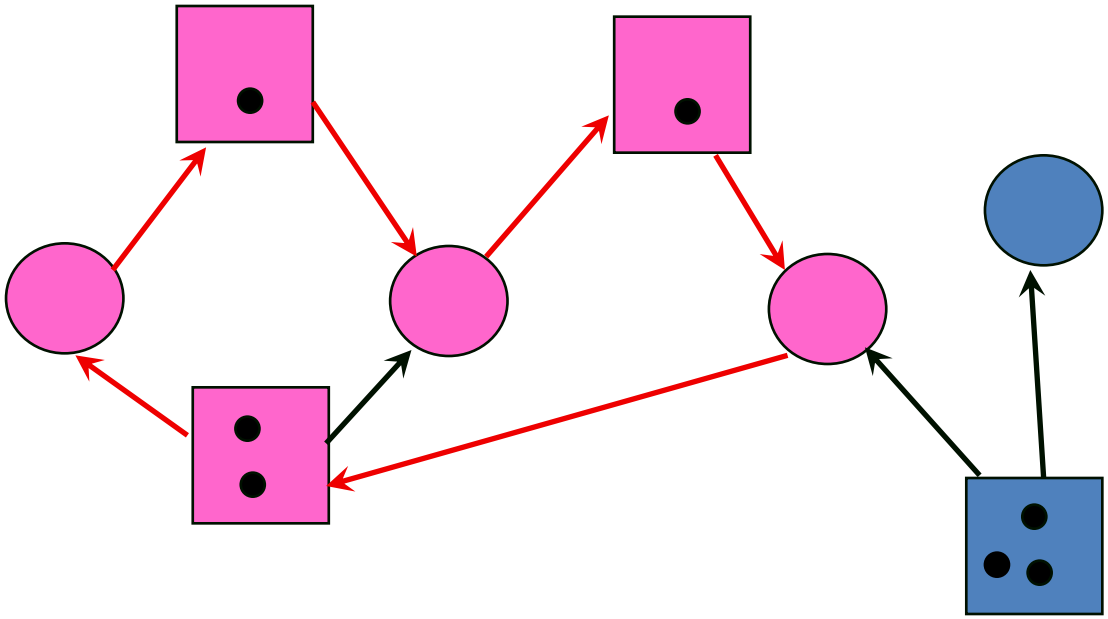


Free
Resources

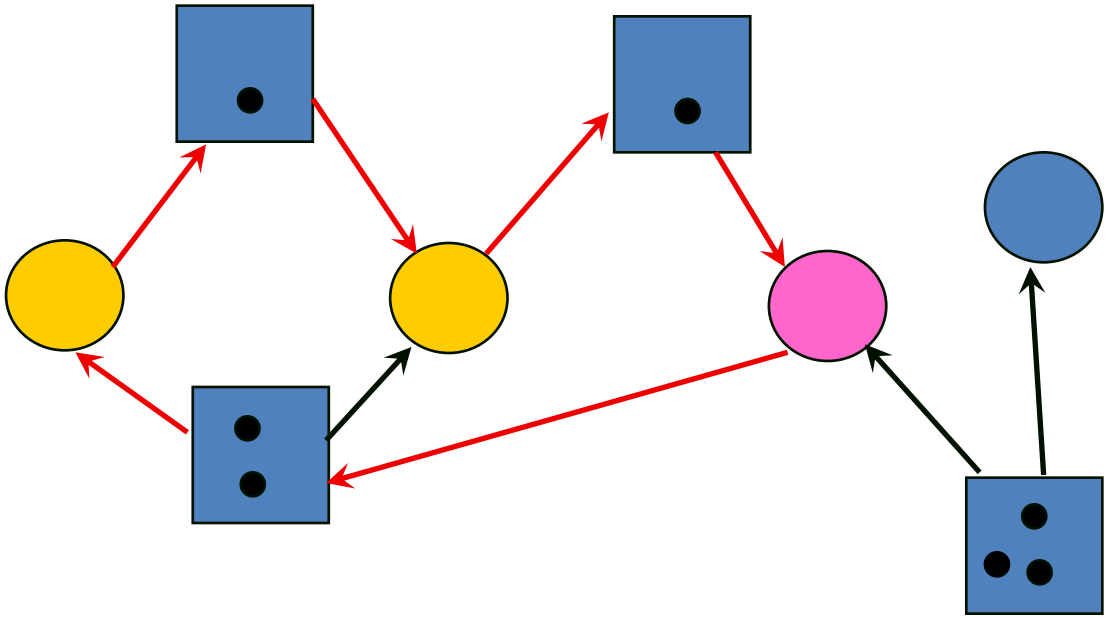
Deadlock Example



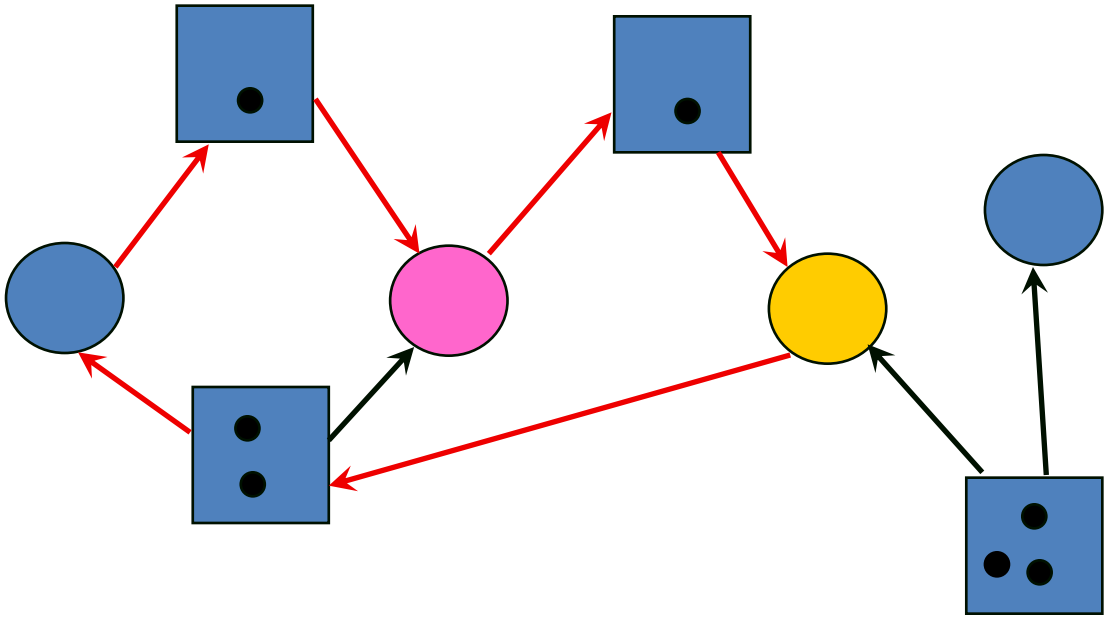
Deadlock Example



Deadlock Example



Deadlock Example



Deadlock Example