CS 414 – Multimedia Systems Design

Lecture 31 – Process Management (Part 1)

Klara Nahrstedt

Spring 2009

Outline

- **MP4 is out**, Start early
- **Discussion Session, April 14, 7pm in 3401 SC**
- Competition on **May 1, 2009, 5-6:30pm in 216 SC**
- Deadline April 30 (pre-competition to decide on the finalists)
  - Exact rules, scenarios of the competition will be posted next Monday, April 20.
- All should come, pizza and 1,2,3rd prices of the competition will be provided between 6:30-7pm on May 1 in 0216 SC

CPU Scheduling

- Maintain many programs in memory
- Determine how to best schedule them
- Requires information about the processes and an understanding of the system goals
- Switch among processes rapidly so each user perceives direct ownership of system

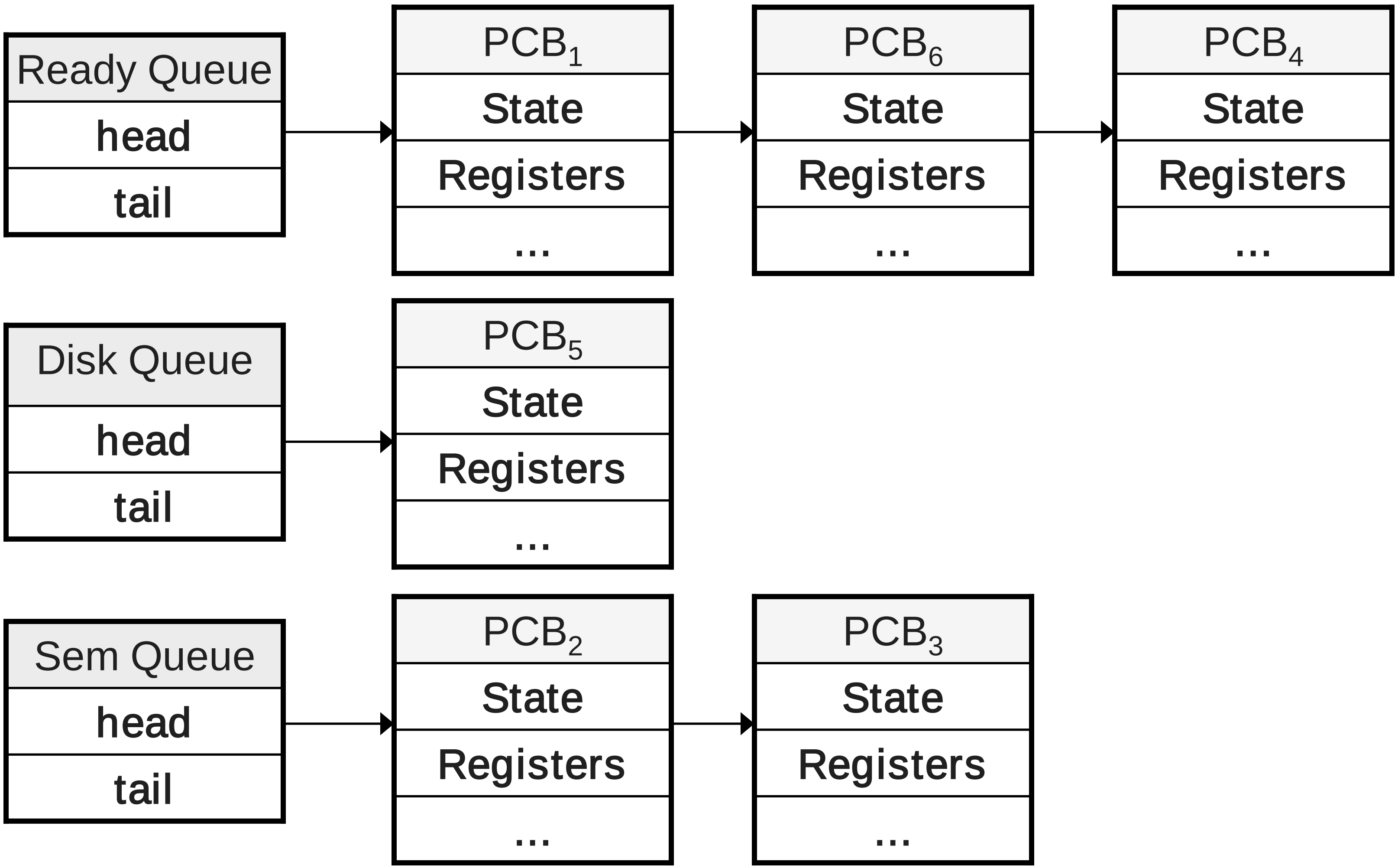
Process State

- *New* – being created
- *Terminated* – finishing execution
- *Running* – executing on the CPU
- *Waiting* – blocking on an event
- *Ready* – waiting for the CPU

Process Control Block

- Represents a process in the OS
  - process state
  - program counter
  - CPU registers
  - scheduling information
  - virtual memory information (e.g., page tables)
  - I/O status information

Resource Queues



CPU Scheduler

- When CPU becomes idle, select next process from the ready queue

- CPU may become idle when:
  - allotted time slice expires
  - interrupt occurs (timer, I/O, user input)
  - application makes an I/O request
  - application terminates

CPU Scheduling Evaluation Criteria

- *Throughput*: processes completed per unit time
- *Turnaround*: from submission to termination
- *Wait*: time waiting in the ready queue
- *Response*: from user request to system response
- *Utilization*: how busy the CPU is over time

- Want predictable system behavior

Scheduling Algorithms

- FCFS
- Shortest Job First
- Priority scheduling
- Round-robin
- Multi-level queue

For General Purpose Process CPU Scheduling

- Rate monotonic
- Earliest deadline first

For RT/Multimedia Process CPU Scheduling

Real-time/Multimedia Processing Requirements

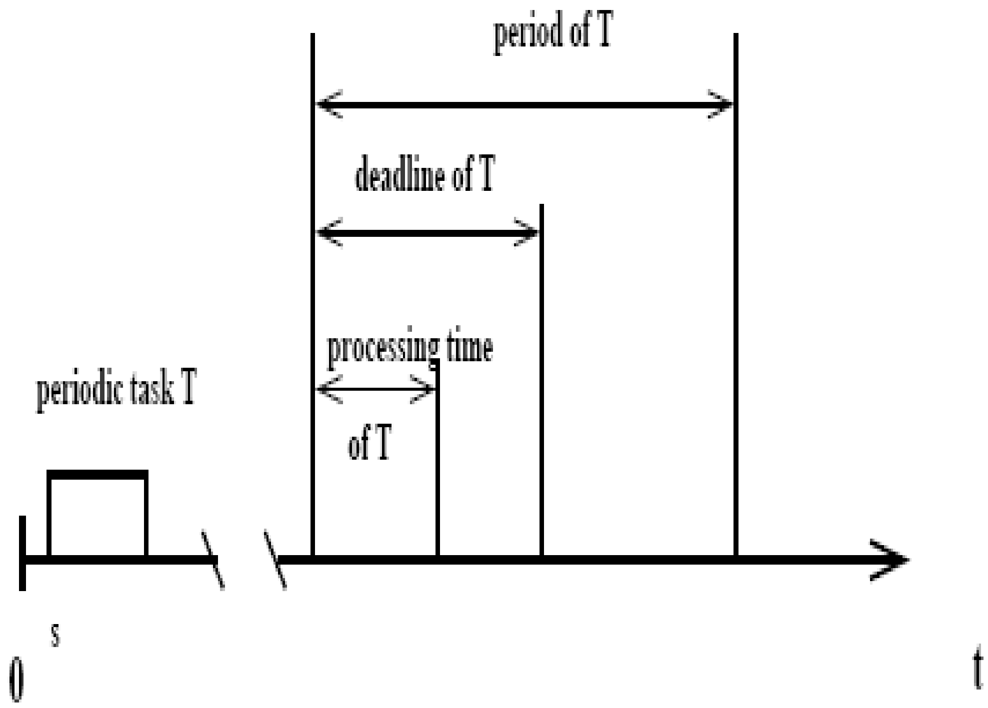
- Need to process continuous multimedia data
  - Processing occurs in predetermined, usually periodic intervals
  - Processing must be completed by certain deadlines
- Need RT process manager
  - Perform admission control
  - Determine schedule
  - Perform reservation
  - Schedule to give processing guarantees

RT/Multimedia Processing Requirements

- Main Problem: How to find a feasible schedule?
- Conflicting Goals/Problems:
- How do we schedule multimedia (RT) processes so that
  1. non-RT processes do not starve when RT process is running
  2. RT process is not subject to priority inversion

Model in RT Scheduling

- Task (Process) – schedulable unit
- Task characterized by
  - **Timing constraints**
  - Resource requirements
- Assumptions
  - **Periodic tasks without precedence relations**
- Time constraints
  - s – task starting point
  - e – task processing time
  - d – task deadline
  - p – task period



Model of RT Scheduling

- Congestion avoidance deadline
  - If period at (k-1) step is equal to ready (start) time of period k
- Tasks: preemptive vs. non-preemptive
- Main goal of RT Scheduling:
  - find feasible schedule of all periodic tasks so that newly arriving task and all previous admitted tasks finish processing in every period according to their deadline

Model of RT Scheduling

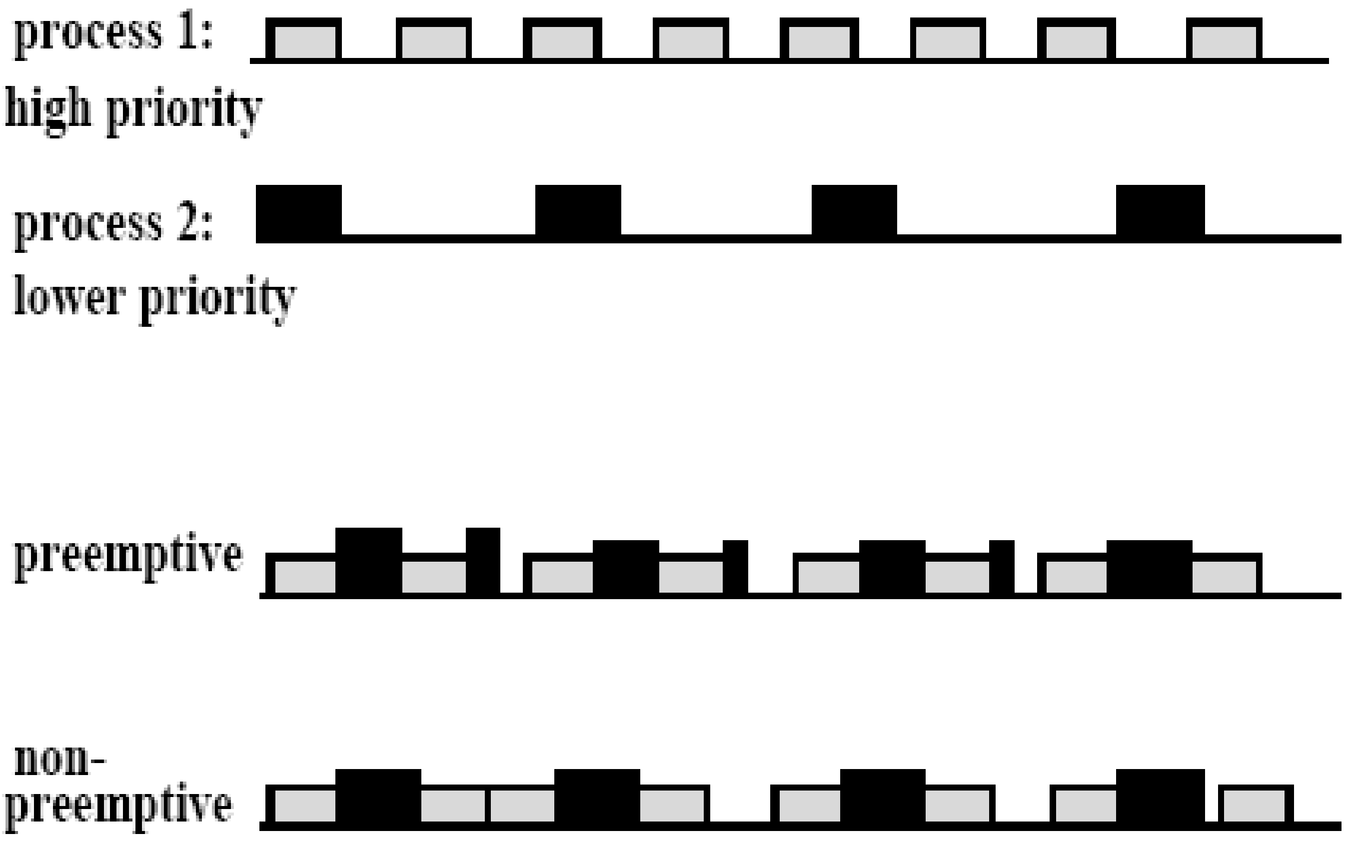
- Must have Schedulability (Admission) Test for RT tasks
- What is the performance metric for RT tasks?
  - Guarantee ratio := number of guaranteed tasks/total number of tasks
  - Process utilization (U):

$$U = \sum_{i=1}^{n} \frac{e_i}{p_i}$$

Scheduling Policies of RT Tasks

- Rate- Monotonic Scheduling (RMS)
  - Designed/proved by C.L. Liu and Layland 1973
  - Policy: task with highest rate has highest priority
  - Static and optimal, priority-driven
    - Optimal means that there no other static algorithm that is able to schedule a RT task which can't be scheduled by RMS algorithm
    - Assumptions:
      - Tasks are periodic
      - Each task must complete before next request
      - All tasks are independent
      - Run-time of each task request is constant
      - Any non-periodic task has no required deadline

Example of RMS

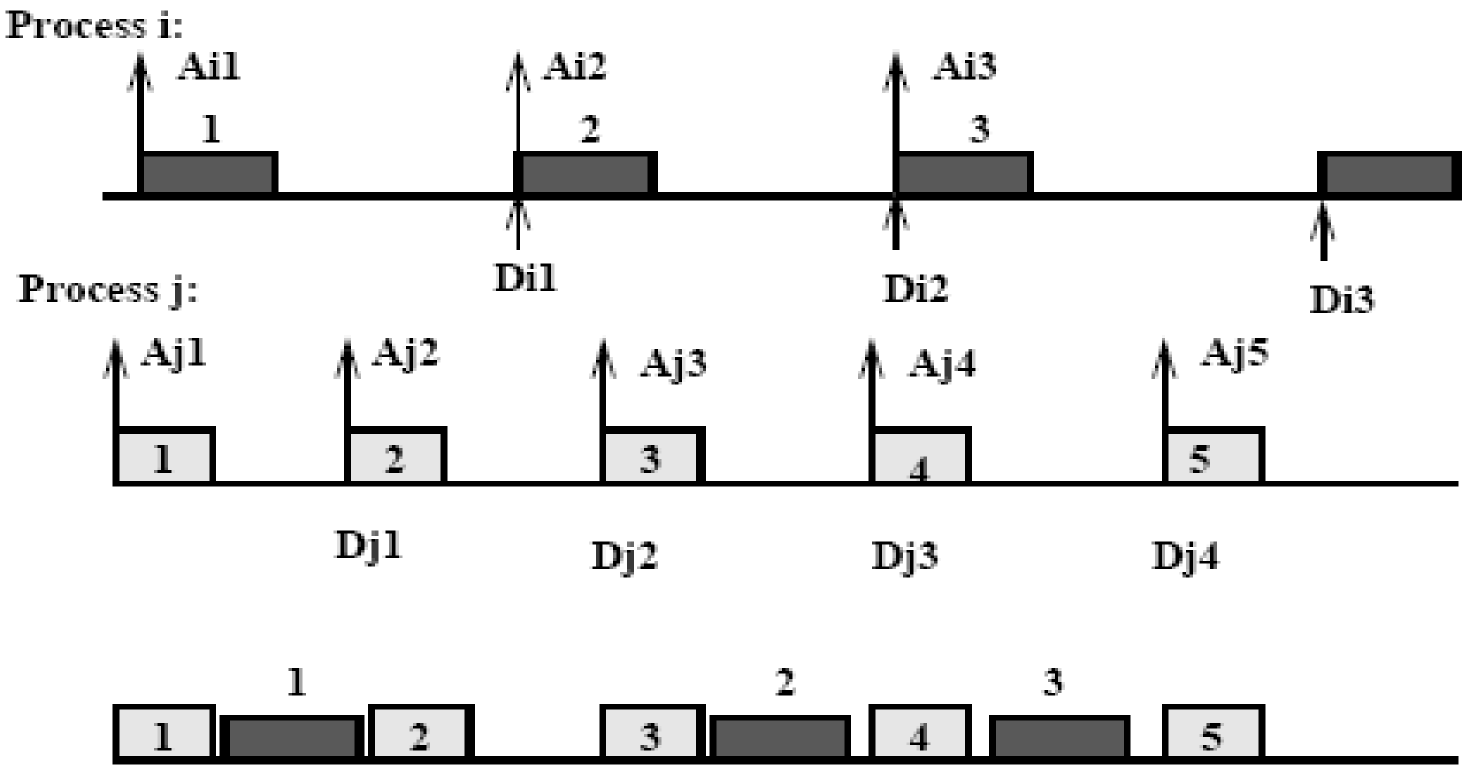


=> Relative Stream priority is fixed

Scheduling Policies for RT Tasks

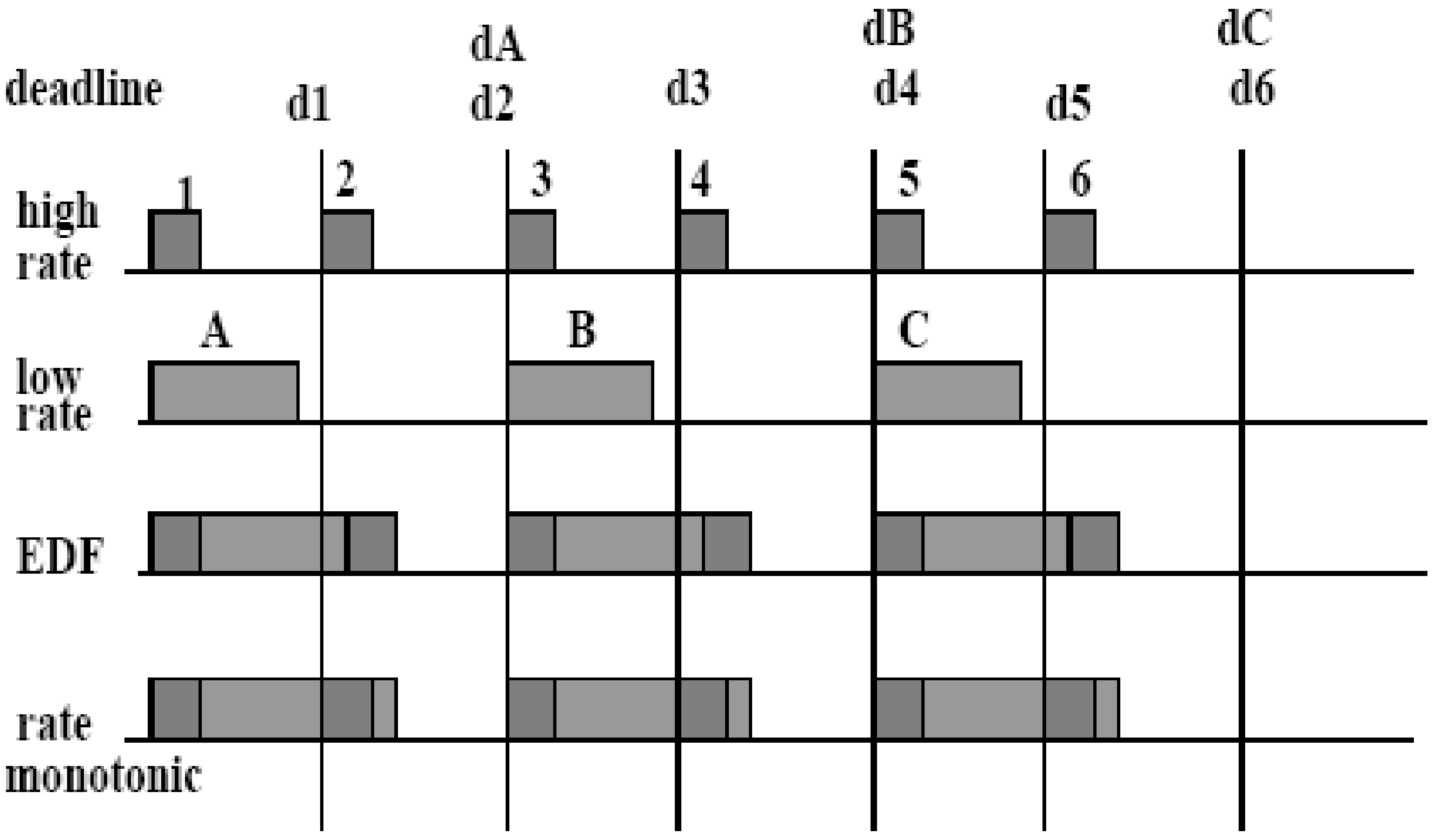
- Earliest Deadline First (EDF) Policy
  - Optimal dynamic algorithm
  - Produces valid schedule if one exists
  - Complexity $O(n^2)$
  - Upper bound of process utilization 100%
  - Policy: task with earliest deadline has highest priority

Example of EDF



Both streams scheduled according to their deadlines

Comparison between RMS and EDF



Admission Control (for preemptive tasks)

- Schedulability test for RMS

$$U \leq \ln 2 \qquad \sum_{i=1}^{n} \frac{e_i}{p_i} \leq \ln 2$$

- Schedulability test for EDF

$$U \leq 1 \qquad \sum_{i=1}^{n} \frac{e_i}{p_i} \leq 1$$

Example

- Consider the following preemptive RT tasks and their characteristics
  - T1: p1 = 50ms, e1=10ms
  - T2: p2 = 100ms, e2=20ms
  - T3: p3 = 200ms, e3=50ms
  - T4: p4 = 100ms, e4=20ms
- Are these tasks schedulable via RMS?
  - If yes, what is the feasible schedule?
- Are these tasks schedulable via EDF?
  - If yes, what is the feasible schedule?

Conclusion

- RMS and EDF are basic policies for real-time scheduling systems
- For multimedia systems, soft-real-time scheduling (SRT) concepts needed, connecting reservation-based and adaption-based SRT
- Next lecture – we look at DSRT system that shows implementation of hybrid SRT systems