

CS241 System Programming

CPU Scheduling (2)

Klara Nahrstedt

Lecture 11

2/13/2006



Content

- Basic Scheduling Algorithm (FCFS)
- Round Robin
- Priority Scheduling
- Summary

Administrative

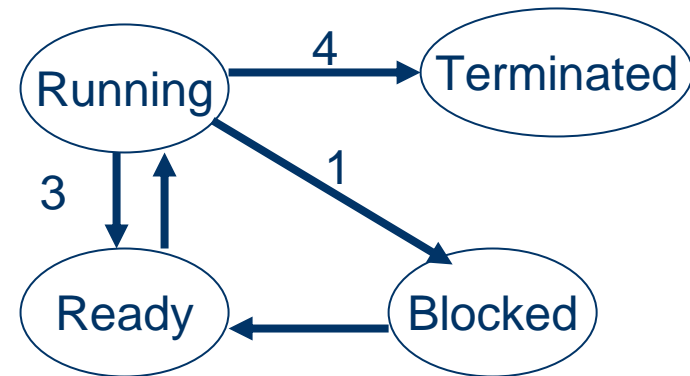
- MP2 posted today, due February 27
- MP1 due today, February 13
- Read T: 2.4
- Quiz 3 will be on Friday, February 17, 2006
- Quiz 3 will include material from Tanenbaum, 3rd edition, section 2.2, 2.3, 2.4 AND LECTURE NOTES (see lecture note files)
 - lec6-syn1
 - lec7-syn2
 - lec8-syn3
 - lec8-syn4
 - lec10-sched
 - lec11-sched

Preemptive vs. Non-preemptive

- **Non-preemptive scheduling:**

- The running process keeps the CPU until it **voluntarily** gives up the CPU

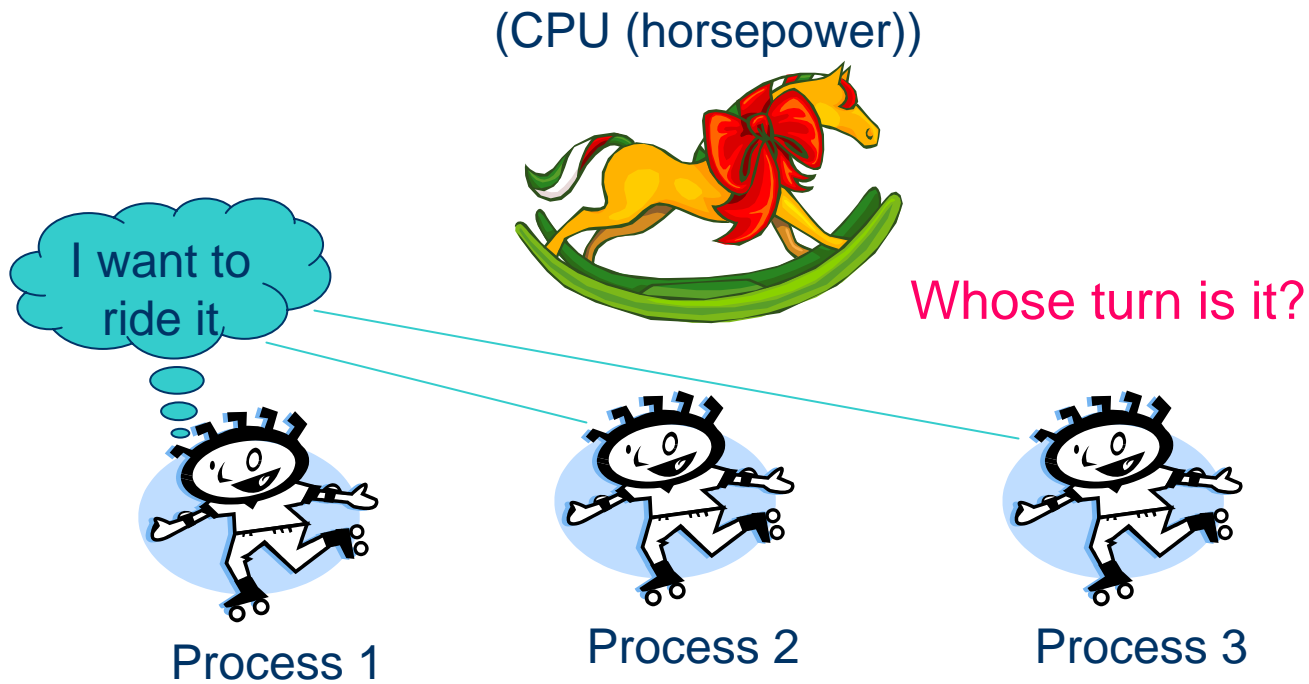
- process exits
- switches to blocked state
- 1 and 4 only (no 3)



- **Preemptive scheduling:**

- The running process can be interrupted and must release the CPU (can be **forced** to give up CPU)

What are the scheduling objectives?



Scheduling Objectives

- **Fair** (nobody cries)
- **Priority** (lady first)
- **Efficiency** (make best use of equipment)
- **Encourage good behavior** (good boy/girl)
- **Support heavy loads** (degrade gracefully)
- **Adapt to different environments** (interactive, real-time, multi-media)

Performance Criteria

- **Fairness**
- **Efficiency**: keep resources as busy as possible
- **Throughput**: # of processes that completes in unit time
- **Turnaround Time** (also called elapse time)
 - amount of time to execute a particular process from the time its entered
- **Waiting Time**
 - amount of time process has been waiting in ready queue
- **Response Time**
 - amount of time from when a request was first submitted until first response is produced.
 - predictability and variance
- **Policy Enforcement**:
 - seeing that stated policy is carried out
- **Proportionality**:
 - meet users' expectation
- **Meeting Deadlines**: avoid losing data



Different Systems, Different Focuses

- For all
 - Fairness, policy enforcement, resource balance
- Batch Systems
 - Max throughput, min turnaround time, max CPU utilization
- Interactive Systems
 - Min Response time, best proportionality
- Real-Time Systems
 - predictability, meeting deadlines

Process Profiles

- I/O – Bound
 - Does too much I/O to keep CPU busy
- CPU – Bound
 - Does too much computation to keep I/O busy
- Process Mix
 - Scheduling should load balance between I/O bound and CPU-bound processes
 - Ideal would be to run all equipment at 100% utilization but that would not necessarily be good for response time

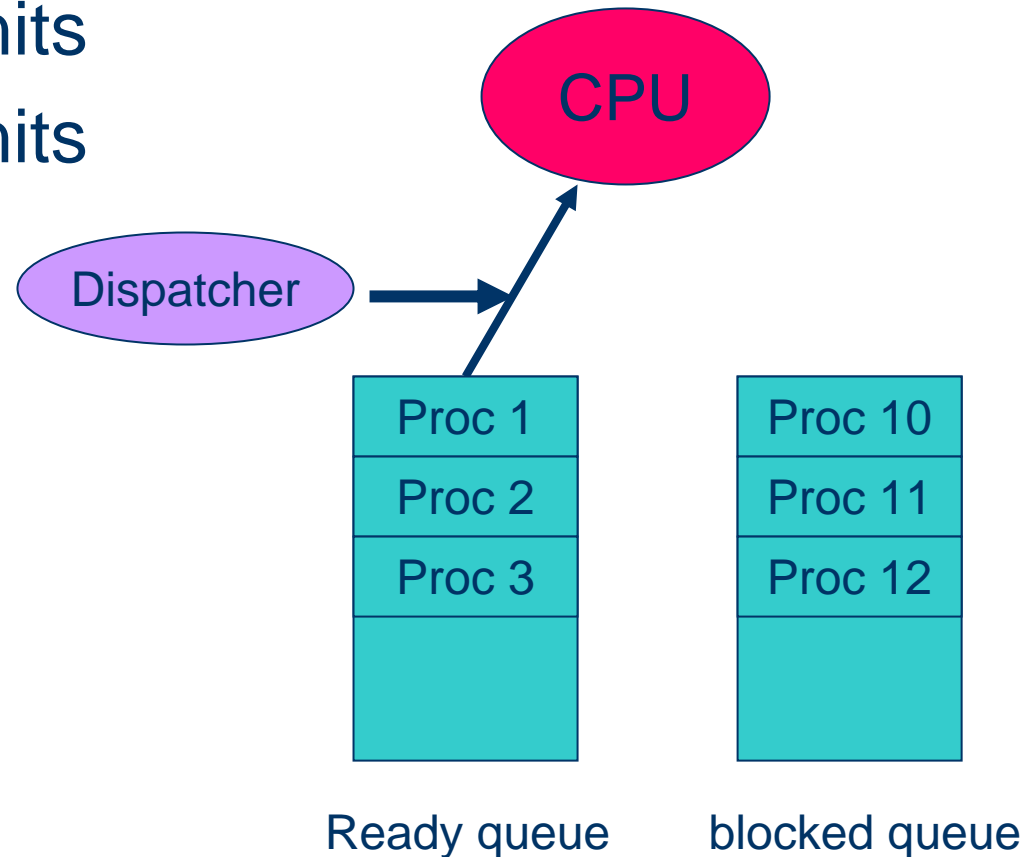
Program Behaviors Considered in Scheduling

- Is it I/O bound? Example? 
- Is it CPU bound? Example? 
- Batch or interactive environment
- Urgency
- Priority
- Frequency of page faults
- Frequency of preemption
- How much execution time it has already received
- How much execution time it needs to complete

CPU Scheduler

- Proc 1: 14 time units
- Proc2: 8 time units
- Proc3: 8 time units

- Dispatcher
- Preemptive vs. non-preemptive



Dispatcher

- Gives the control of the CPU to the process, scheduled by the short-term scheduler.
- Functions:
 - switching context
 - switching to user mode
 - jumping to the proper location in the user program.
- ***Dispatch Latency***: time to stop process and start another one.
 - Pure overhead
 - Needs to be fast

Single Processor Scheduling Algorithms

- Batch systems
 - First Come First Serve (FCFS)
 - Short Job First
- Interactive Systems
 - Round Robin
 - Priority Scheduling
 - Multi Queue & Multi-level Feedback
 - Shortest process time
 - Guaranteed Scheduling
 - Lottery Scheduling
 - Fair Sharing Scheduling

First Come First Serve (FCFS) (Batch Scheduling Policy)

- Process that requests the CPU FIRST is allocated the CPU FIRST.
- Also called FIFO
- Non-preemptive
- Used in Batch Systems
- Real life analogy: **Fast food restaurant**
- Implementation: FIFO queues
 - A new process enters the tail of the queue
 - The schedule selects from the head of the queue.
- Performance Metric: **Average Waiting Time.**
- Given Parameters:
 - Burst Time (in ms), Arrival Time and Order

FCFS Example

Process	Duration	Order	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

The final schedule:



P1 waiting time: 0
P2 waiting time: 24
P3 waiting time: 27

The average waiting time:
 $(0+24+27)/3 = 17$

Problems with FCFS

- Non-preemptive
- Not optimal AWT
- Cannot utilize resources in parallel:
 - Assume 1 process CPU bounded and many I/O bounded processes
 - result: **Convoy effect**, low CPU and I/O Device utilization
 - Why?

FCFS Performance

- What is the impact of convey effect?
 - a) Slows down CPU bound processes?
 - b) Slows down I/O bound processes?
 - c) Low CPU utilization?
 - d) Low I/O device utilization?
 - e) All of the above?

Convoy Effects

- Consider $n-1$ jobs in system that are I/O bound and 1 job that is CPU bound.
- I/O bound jobs pass quickly through the ready queue and suspend themselves waiting for I/O.
- CPU bound job arrives at head of queue and executes until complete.
- I/O bound jobs rejoin ready queue and wait for CPU bound job to complete.
- **I/O devices idle until CPU bound job completes.**
- When CPU bound job completes, other processes rush to wait on I/O again.
- **CPU becomes idle.**

Priority Scheduling (Interactive Scheduling Policy)

- Each job is assigned a priority.
- FCFS within each priority level.
- Select highest priority job over lower ones.
- Rational: higher priority jobs are more mission-critical
 - Example: DVD movie player vs. send email
- Problems:
 - May not give the best AWT
 - indefinite blocking or starvation a process

Set Priority

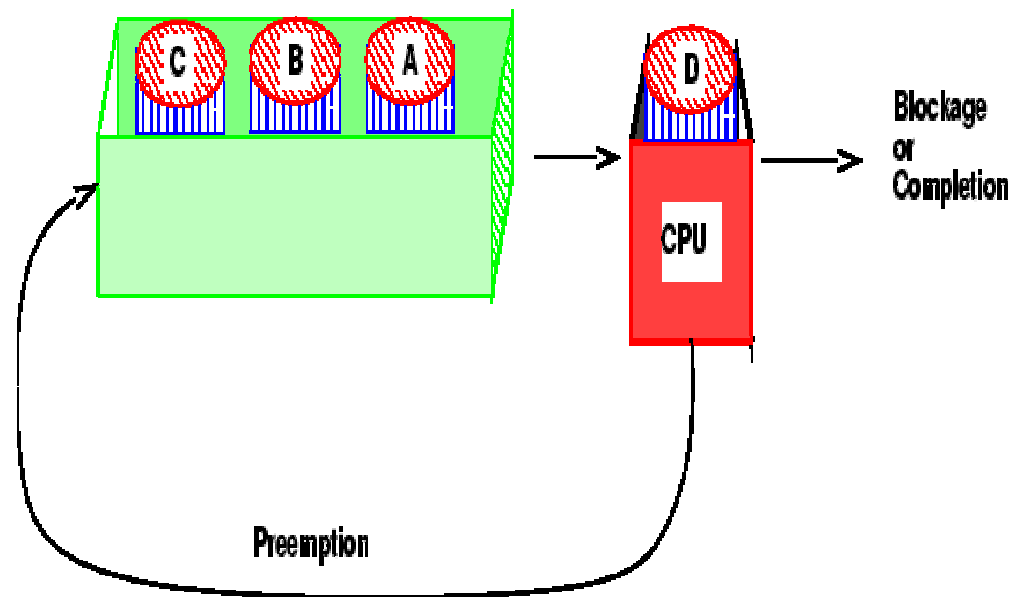
- Two approaches
 - Static (for system with well known and regular application behaviors)
 - Dynamic (otherwise)
- Priority may be based on:
 - Cost to user.
 - Importance of user.
 - Aging
 - Percentage of CPU time used in last X hours.

Round-robin (Interactive Scheduling Policy)

- One of the oldest, simplest, most commonly used scheduling algorithm
- Select process/thread from ready queue in a round-robin fashion (take turns)

Problem:

- Do not consider priority
- Context switch overhead



Round-robin: Example

Process	Duration	Order	Arrival Time
P1	3	1	0
P2	4	2	0
P3	3	3	0

Suppose time quantum is: 1 unit, P1, P2 & P3 never block

P1 P2 P3 P1 P2 P3 P1 P2 P3 P2



0 10

P1 waiting time: 4

P2 waiting time: 6

P3 waiting time: 6

The average waiting time (AWT):
 $(4+6+6)/3 = 5.33$

Time Quantum

- Time slice too large
 - FIFO behavior
 - Poor response time
- Time slice too small
 - Too many context switches (overheads)
 - Inefficient CPU utilization
- Heuristic: 70-80% of jobs block within time-slice
- Typical time-slice 10 to 100 ms
- Time spent in system depends on size of job.

Summary

- What is scheduling
- Scheduling objectives
- CPU Scheduling
- FCFS
- Priority
- Round Robin
- Next lecture: other scheduling algorithms
 - Shortest Job First
 - Multi-Queue
 - Scheduling in Real-Time Systems
 - Thread Scheduling