Announcements

- Reading 5 (5.1-5.3,5.6)
- Midterm #1 is March 10 in class
  - covers material through and including lecture 11
    - problems at the end of the chapters
    - synchronization problems
    - questions about the project
  - Suggestions for study
    - see problems on web page

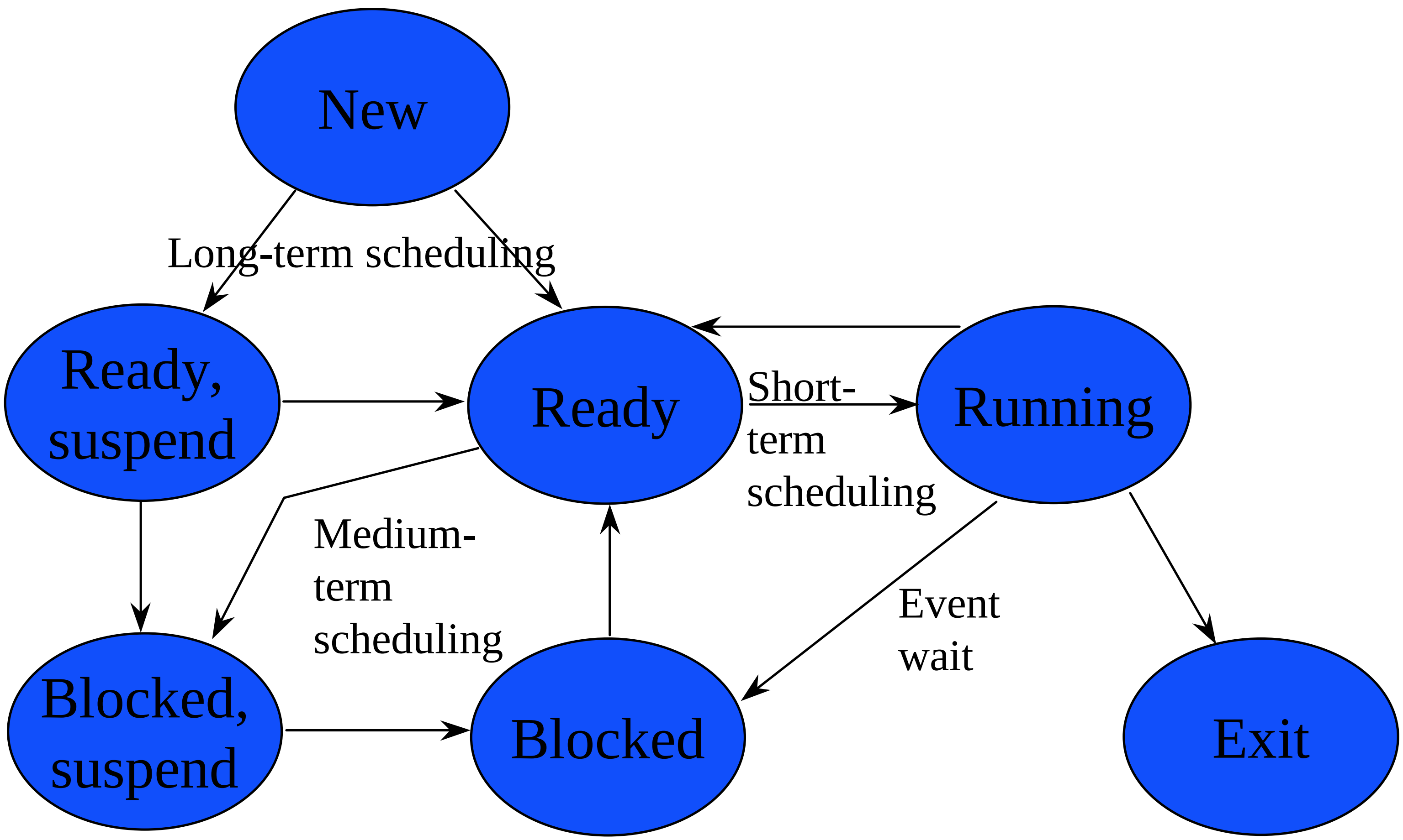
CPU Scheduling

- Manage CPU to achieve several objectives:
  - maximize CPU utilization
  - minimize response time
  - maximize throughput
  - minimize turnaround time
- Multiprogrammed OS
  - multiple processes in executable state at same time
  - scheduling picks the one that will run at any give time (on a uniprocessor)
- Processes use the CPU in bursts
  - may be short or long depending on the job

Types of Scheduling

- At least 4 types:
  - long-term - add to pool of processes to be executed
  - medium-term - add to number of processes partially or fully in main memory
  - short-term - which available process will be executed by the processor
  - I/O - which process's pending I/O request will be handled by an available I/O device
- Scheduling changes the ***state*** of a process

Process State Transitions



Long-term scheduling

- Determine which programs admitted to system for processing - controls degree of multiprogramming
- Once admitted, program becomes a process, either:
  - added to queue for short-term scheduler
  - swapped out (to disk), so added to queue for medium-term scheduler
- Batch Jobs
  - Can system take a new process?
    - more processes implies less time for each existing one
    - add job(s) when a process terminates, or if percentage of processor idle time is greater than some threshold
  - Which job to turn into a process
    - first-come, first-serve (FCFS), or to manage overall system performance (e.g. based on priority, expected execution time, I/O requirements, etc.)

Medium vs. Short Term Scheduling

- Medium-term scheduling
  - Part of swapping function between main memory and disk
    - based on how many processes the OS wants available at any one time
    - must consider memory management if no virtual memory (VM), so look at memory requirements of swapped out processes
- Short-term scheduling (dispatcher)
  - Executes most frequently, to decide which process to execute next
  - Invoked whenever event occurs that interrupts current process or provides an opportunity to preempt current one in favor of another
  - Events: clock interrupt, I/O interrupt, OS call, signal

Scheduling criteria

- Per processor, or system oriented
  - CPU utilization
    - maximize, to keep as busy as possible
  - throughput
    - maximize, number of processes completed per time unit
- Per process, or user oriented
  - turnaround time
    - minimize, time of submission to time of completion.
  - waiting time
    - minimize, time spent in ready queue - affected solely by scheduling policy
  - response time
    - minimize, time to produce first output
    - most important for interactive OS

Scheduling criteria
non-performance related

- Per process
  - predictability
    - job should run in about the same amount of time, regardless of total system load
- Per processor
  - fairness
    - don’t starve any processes, treat them all the same
  - enforce priorities
    - favor higher priority processes
  - balance resources
    - keep all resources busy

Short-term scheduling algorithms

- First-Come, First-Served (FCFS, or FIFO)
  - as process becomes ready, join Ready queue, scheduler always selects process that has been in queue longest
  - better for long processes than short ones
  - favors CPU-bound over I/O-bound processes
  - need priorities, on uniprocessor, to make it effective

Algorithms (cont.)

- Round-Robin (RR)
  - use preemption, based on clock - time slicing
    - generate interrupt at periodic intervals
  - when interrupt occurs, place running process in Ready queue, select next process to run using FCFS
  - what's the length of a time slice
    - short means short processes move through quickly, but high overhead to deal with clock interrupts and scheduling
    - guideline is time slice should be slightly greater than time of "typical job" CPU burst
  - problem dealing with CPU and I/O bound processes

Algorithms (cont.)

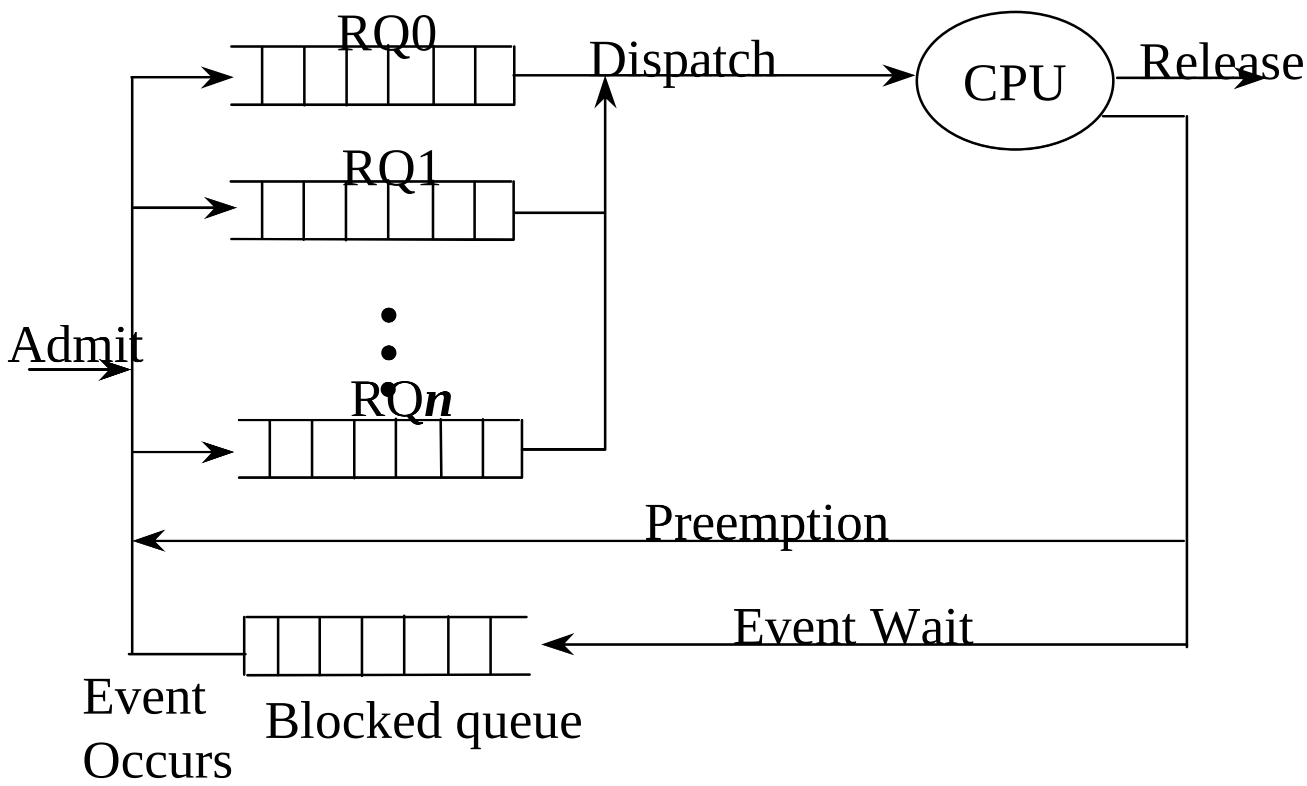
- Shortest Process Next (SPN)
  - non-preemptive
  - select process with shortest expected processing time
  - improves response time, but increases its variability, reducing predictability - provably decreases average waiting time
  - problem is estimating required processing time
  - risk of starving longer processes, as long as there are shorter processes around
  - not good for time sharing - non-preemptive

Algorithms (cont.)

- Shortest Remaining Time (SRT)
  - preemptive version of SPN
  - scheduler chooses process with shortest expected remaining process time
  - still need estimate of processing time, and can starve longer processes
    - no bias in favor of longer processes, as in FCFS
    - no extra interrupts as in RR, so reduced overhead
  - must record elapsed service times
  - should give better turnaround time than SPN

Priority Based Scheduling

- Priorities
  - assign each process a priority, and scheduler always chooses process of higher priority over one of lower priority
- More than one ready queue, ordered by priorities



Priority Algorithms

- Fixed Queues
  - processes are statically assigned to a queue
  - sample queues: system, foreground, background
- Multilevel Feedback
  - processes are dynamically assigned to queues
  - penalize jobs that have been running longer
  - preemptive, with dynamic priority
  - have ***N*** ready queues (RQ0-RQ***N***),
    - start process in RQ0
    - if quantum expires, moved to i + 1 queue