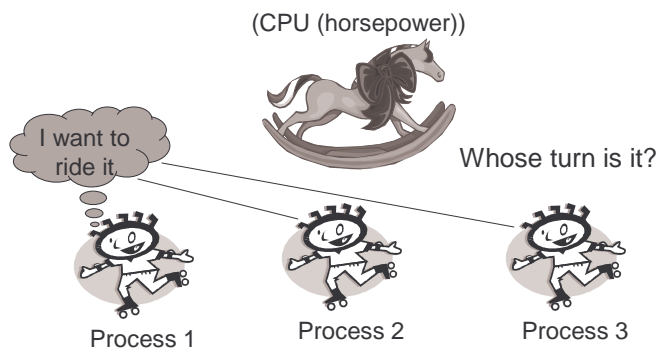


CPU Scheduling

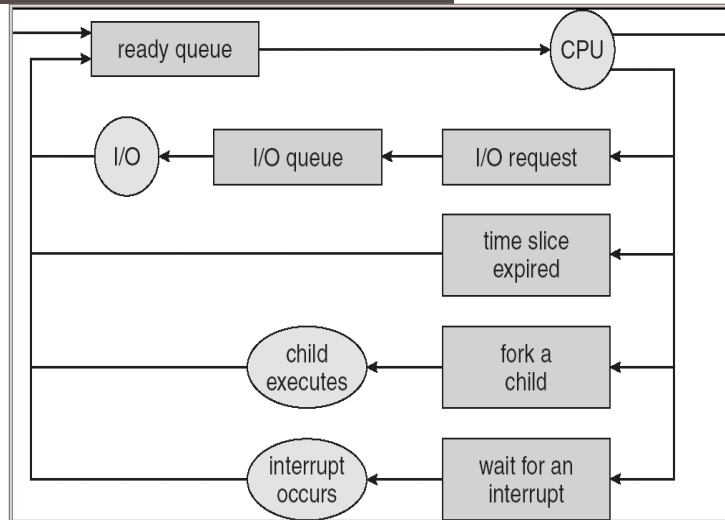
- Why Scheduling?
- Basic Concepts of Scheduling
- Scheduling Criteria
- Basic Scheduling Algorithm (FCFS)
- Summary

Why Scheduling?

- Deciding which process/thread should occupy the resource (CPU, disk, etc)



When to Schedule?



CSE660: Introduction to Operating Systems

3

Scheduling Objectives

- Fairness
- Priority
- Efficiency Encourage good behavior
- Support heavy loads
- Adapt to different environments
(interactive, real-time, multi-media)

CSE660: Introduction to Operating Systems

4

Performance Criteria

- Fairness : no starvation
- 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 complete a particular process from its beginning
- 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.
- Policy Enforcement
 - Enforcing that stated policy is carried out
- Proportionality
 - meet users' expectation
- Meeting Deadlines: avoid losing data

Different Systems, Different Focuses

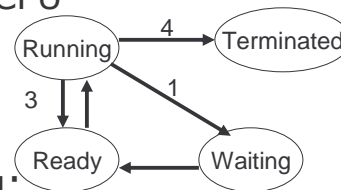
- For all
 - Fairness, policy enforcement, resource balance
- Batch Systems
 - Max throughput, min turnaround time, max CPU utilization
- Interactive Systems
 - Min Response time, best proportionality
- Real-Time Systems
 - predictability, meeting deadlines

Preemptive vs. Non-preemptive

□ Non-preemptive scheduling:

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

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



□ Preemptive scheduling:

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

Process Behavior

□ I/O – Bound

- Does too much I/O to keep CPU busy


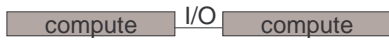
□ CPU – Bound

- 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

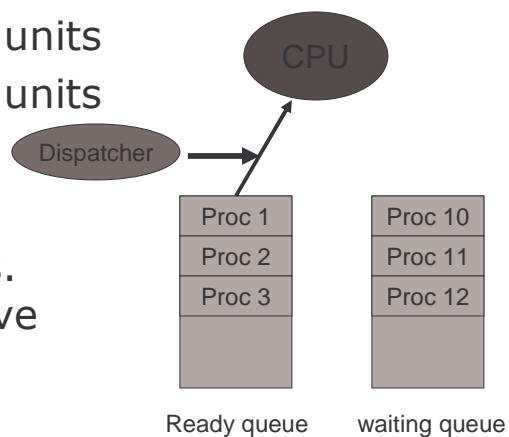
Program Characteristics Considered in Scheduling

- Is it I/O bound? 
- Is it CPU bound? 
- 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 a process and start another one.
 - Pure overhead
 - Needs to be fast

Single Processor Scheduling Algorithms

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

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?

Why Convoy Effects?

- Consider 100 I/O-bound processes and 1 CPU-bound job in the system.
- I/O-bound processes pass quickly through the ready queue and suspend themselves waiting for I/O.
- The CPU-bound process arrives at head of queue and executes the program until completion.
- I/O bound processes rejoin the ready queue and wait for the CPU-bound process releasing the CPU.
- I/O devices idle until the CPU-bound process completes.
- In general, a convoy effect happens when a set of processes need to use a resource for a short time, and one process holds the resource for a long time, blocking all of the other processes. Essentially, it causes poor utilization of the other resources in the system.