CPSC 457
Operating Systems

Lecture 5

Scheduling

Today

Updates

- Assignment 1
- Assignment 2
- Concept Review

Scheduling Processes

- Concepts
- Algorithms

Assignment 1

Assignment 2

Concept Overview

Last Time

Processes

- Creation
- Termination
- Information
- Context Switching

Threads

- Split Execution from Resources
- Life Span
- Implementation

Scheduling Processes

Scheduling Concepts

- Scheduling Timing
- Preemptive vs Nonpreemptive
- Types of Scheduling Algorithms
- Scheduling Algorithm Goals and Metrics

Scheduling Algorithms

- First Come, First Served
- Shortest Job First
  - Exponential Average
- Priority Scheduling
- Round-Robin
- Multi-Queue Scheduling
- And many more!

Scheduling Concepts

Deciding which process should be running

Defining process behaviour

CPU Burst Length

Deciding when to schedule

Do we interrupt a process once it's running

Preemptive vs Nonpreemptive

Scheduling Goals and Metrics

- Fairness
- Policy Enforcement
- Balance
- Throughput
- Turnaround Time
- CPU Utilization
- Response Time
- Wait Time
- Proportionality
- Meeting Deadlines
- Predictability

Real Time

Types of Scheduling Algorithm

- Batch
- Interactive
- Real Time

Scheduling Algorithms

First Come, First Served
Shortest Job First
Priority Scheduling
Round-Robin
Multi-Queue Scheduling

Guaranteed Scheduling
Lottery Scheduling
Fair-Share scheduling

Break time

First Come, First Served

Algorithm

- As processes arrive to be scheduled, they are added to the end of the queue.
- When it's time to schedule a new process, the scheduler picks the head of the queue.

First Come, First Served

Comments

- Nonpreemptive
- Bad for an interactive system
- Turnaround and Wait Times based on the sequence of arrival
- Easy to program
- Tends to bunch up I/O bound processes

Shortest Job First

Algorithm

- As processes arrive to be scheduled, they are added to the end of the queue.
- When it's time to schedule a new process, the scheduler picks the process with the shortest **CPU burst** next.

Shortest Job First

Problem

- We don't know how long each CPU burst is going to be.
- But we're good at estimating things:

Exponential Average for CPU Burst Time

$$\clubsuit_{n+1} = \heartsuit t_n + (1 - \heartsuit)\ \clubsuit_n$$

Powerpoint on mac...

$\clubsuit_n$ - the predicted burst time for the n^{th} burst of the process

$\clubsuit t_n$ - the actual burst time for the n^{th} burst of the process

$\heartsuit$ - weight to bias the average more to the last burst or the history

Shortest Job First

Comments

- Preemptive or Nonpreemptive
- Can lead to **starvation**
- Provably optimal (if all jobs arrive at the same time)
- Slightly harder to program
  - Need to implement burst estimation and track for processes

Priority Scheduling

Algorithm

- Processes are assigned different priorities (somehow)
- A process is added to the queue for it's priority
- The scheduler selects (fcfs, sjf or someotherway) from all of the processes in the highest priority queue

Priority Scheduling

Comments

- Preemptive or Nonpreemptive
- Can lead to **starvation**
- We can use **aging** to stave off starvation
- Adheres to system policy
- Still fairly easy to implement

Round Robin

Algorithm

- Start with FCFS
- Preempt processes after a fixed amount of CPU time **(called the Time Quantum or the Time Slice)**
- If you stopped a process for using all of it's time, put it on the back of the queue.

Round Robin

Comments

- Preemptive
- Have to factor in context switch time
- We can adjust the quantum to alter the behaviour of the system
- Generally want the quantum to long enough that the majority of processes (I/O bound) finish within it

Where Next?

More Scheduling

Real World Implementations

Memory Management