Processes 2

Jonathan Windle

University of East Anglia

J.Windle@uea.ac.uk

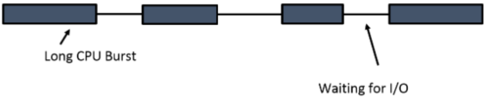
June 9, 2017

Overview I

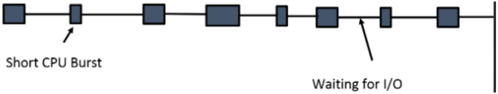
1. Types of Process
2. When to schedule
3. Scheduling Goals
4. Computing Environments
5. Scheduling Algorithms
    - First-Come-First-Serve
    - Shortest Job First
    - Shortest time Remaining
    - Round Robin
    - Priority Scheduling
    - Multiple Queue Schedular
    - Lottery Scheduling

Types of Process

- Compute bound:



- I/O bound:



- Types of Scheduling Algorithm:
  - Nonpreemptive
    - Picks a process to run and lets it run until it blocks or completes.
  - Preemptive:
    - Picks a process to run and lets it run for a fixed period of time.

When to Schedule

1. When a process switches from the running state to the waiting state.
   - e.g. When it blocks waiting for I/O
2. When a process switches from the running state to the ready state.
   - e.g. When an interrupt occurs
3. When a process switches from the waiting state to the ready state.
   - e.g. When an I/O operation completes.
4. When a process terminates

Nonpreemptive scheduling takes place under circumstances 1 and 4 only. Others are preemptive.

Scheduling Goals

- All scheduling algorithms should be fair.
- Scheduling policies must be enforced.
  - Safety controll processes should get priority (even if the payroll is delayed).
- All parts of the system should be kept busy)as this gets more work done per second than if some are idle).
  - In a batch system, where the scheduler controls which jobs are brought into memory, its better to have a mix of CPU bound and I/O bound processes.

Computing Environments

- Batch:
  - No user interaction, processes can be switched infrequently.
  - Aim to maximize throughput and CPU utilization.
  - Nonpreemptive scheduling is used.
- Interactive
  - Processes must reflect user actions e.g. Time-shared multiprogramming
  - Aim to minimize responsee time
  - Preemptive scheduling is needed
- Real-Time
  - System must meet real-time constraints
  - Well designed systems to not need preemptive scheduling.

Scheduling Algorithms

- First-Come-First-Served
- Shortest Job First
- Shortest Remaining Time
- Round-Robin
- Priority Scheduling (and derivatives)

Batch Systems

Interactive Systems

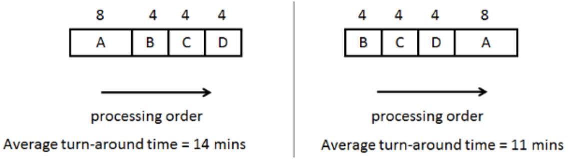
Real-Time Systems

First-Come-First-Serve

- The process that has waited longest, goes next.
- Process then run to completion or blocked (nonpreemptive)
- Advantages:
  - Queue does not need to be ordered
  - Simple
  - Fair
- Disadvantages:
  - No consideration on throughput
  - Potentially long turnaround (if the queue is large).
  - Short or I/O bound processes are penalised.

Shortest Job First

- Ordered by time to complete
- Allowed to run until complete (nonpreemptive)
- Reordered at context switch
- Allows high throughput but penalises long processes.
- Open to abuse - hard to accurately estimate required time.
- Processes could underestimate the required time to jump ahead in the queue.



Shortest time Remaining

- Order processed by closest time to completion.
- As new jobs are submitted, choose the job closest to terminating.
- Provides good throughput and response time
- Still penalises long processes.
- Difficult to predict remaining time.
- Advantages:
  - Favours new "short" processes.
  - Optimises throughput
- Disadvantages:
  - Penalises long processes.

Round Robin

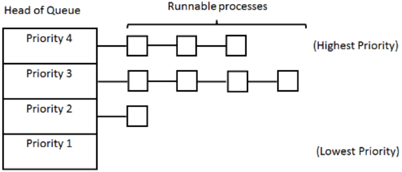
- Defines unit of time (typically 10-50ms) called the quantum
- If it's too short, the system ends up context switching a lot, if it's too long, the system is sluggish and unresponsive.
- Selective process runs for the quantum amount of time.
- After the quantum expires, the CPU is relinquished and process moved to the back of the queue.
- Process at the front is then allocated the CPU for a quantum.
- Advantages:
  - Fair
  - Easy to implement
- Disadvantages:
  - I/O bound processes are penalised.
  - Length of quantum must be carefully chosen to achieve maximum throughput.
  - No account of priority.

Priority Scheduling

- Each process is assigned a priority, so after the quantum expires, the process with the next highest priority is selected.
- Static priority: Priority is fixed for the lifetime of the process.
- Dynamic priority:
  - Priority varies on the CPU usage pattern.
  - Multiply priority by 100/x for the next schedule, where x is % of quantum used.
  - Fairer for I/O bound processes that do not utilise the CPU much before blocking.
- Advantages:
  - Simple to implement, fair
- Disadvantages:
  - Can suffer priority inversion if static.

Multiple Queue Schedular

- Groups processes into priority classes:
  - Uses priority scheduling among the classes.
  - Uses round-robin scheduling withing each class.



Lottery Scheduling

- Give each process a lottery ticket for resources (e.g. CPU time).
- At next context switch, the schedular draws a ticket at random, process holding ticket gets the resource.
- Higher priority processes are given more tickets and therefore higher chance of being chosen.
- New processes have a chance of winning immediately so the system appears responsive.
- Tickets can be exchanged between processes to temporarily increase the priority.
- Advantages:
  - Allocation of tickets reflects fraction of resources allocated.
  - Chance of winning is always determined by number of tickets held.
  - Cooperating processes can exchange tickets if they wish.

Summary

- To hide the effects of interrupts (difficult and dangerous for users to manage) operating systems provide conceptual model consisting of sequentil processes.
- Two classes of process:
  - Compute bound
  - I/O bound
- Two classes of schedular:
  - Nonpreemptive
  - preemptive.
- Different scheduling algorithms have different properties and a choice of particular algorithm may favour one class of process over another.

The End