CS4410/11: Operating Systems

CPU Scheduling

Rachit Agarwal
Anne Bracy
You have multiple CPU tasks (processes) to execute!

- Which one to execute next?

- Answer obvious in this example
- But, usually not!
CPU Scheduling — Example 1 (Idealized)

Arrival Time

Job Length (e.g., #CPU cycles)

4410 HW

Cooking

Calling Dad

t=0

Each job may have a different length
Each job may arrive at different time
### CPU Scheduling — Example 1 (Idealized)

**Arrival Time**

- 4410 HW
- Cooking
- Calling Dad

**Job Length** (e.g., #CPU cycles)

- 4410 HW
- Cooking
- Calling Dad

**Completion Time**

- 4410 HW
- Calling Dad
- Cooking

**Waiting Time of Job 2**

- t=0

**Completion Time of Job 1**

**Completion Time of Job 2**
CPU Scheduling — Why is it important?

Problem encountered in many setting. Similar principles!

• Street
  • Which car should move next?

• Supermarkets
  • Which customer to help next?

• Airports
  • Which plane should land (or fly) next?

• Hospitals
  • Which patient to attend next?
Problem encountered in many setting. Similar principles!

• Within an OS
  • CPU scheduler
  • Disk or I/O Scheduler
  • Network Scheduler
  • ....

CPU Scheduling — Why is it important? [Cont.]
Problem encountered in many setting. Similar principles!

• Networks (Internet, Google, Facebook, ....)

Which user to schedule next? (you or your roommate)

Which packet to schedule next? (Movies or Skype)
CPU Scheduling — Why is it important? [Cont.]

Problem encountered in many setting. Similar principles!

• Cloud (Amazon, Google, Facebook, ....)
  • Thousands of machines
  • Different kind of customers can rent machines
  • Also, internal employees want to use machines
  • Who should use which machine and when?

A very active area of research!!
CPU Scheduling — Lecture Plan

• Learn **about** various scheduling policies

• Learn how to **compare** different policies

• Learn how to **choose** between different policies

• **Strategy:**
  • A lot of examples
  • Active problem solving
Six principles we discussed in first lecture —

• Reliability
• Availability
• Security
• Privacy
• Portability
• Fairness

• Taking it to an extreme: Starvation
• What's even more extreme?
Operating Systems Design Principles

Today, we will focus on two —

- Reliability
- Availability
- Security
- Privacy
- Portability
- Fairness
  - Taking it to an extreme: Starvation
Operating Systems Performance

- Latency
  - How long does a task take to complete?

- Throughput
  - #Tasks per unit time

- Utilization
  - Fraction of resources used over time

- Scalability
  - How does the performance change with size?

- Predictability
  - Consistency (over time) for an objective
Operating Systems Performance

• Deadlines
  • How many of the tasks meet their deadlines?
CPU Scheduling — Latency

• “Tail” Completion Time
  • When does the last task complete?

• Average Completion Time
  • How long does it take to complete a task on an average?

• High Percentile Completion Time
  • How long does it take to complete 90% of the tasks?

• Completion Time of “Small” Tasks

• Waiting time of Tasks

• .....

Scheduling for “Tail” Latency — FIFO, FCFS

- First-In First-Out

Schedule?

- Schedule?
Scheduling for “Tail” Latency — FIFO, FCFS

• First-In First-Out
• When does it matter?
  • Goods: Simple
  • Not-so-goods: High ACT

• Schedule?
Scheduling for “Tail” Latency — LIFO

- Last-In First-Out
- Goods: ?
- Not-so-goods: High ACT
- Not-so-goods: Starvation
- Why?

• Schedule?
Undo

Bugs that took weeks now take minutes
Scheduling for “Average” Latency — SJF

- Shortest-Job First
- **Goods**: Minimizes ACT
- **Not-so-goods**: Starvation
- **Why?**

- Schedule?
Scheduling for “Average” Latency — SJF

- Shortest-Job First
- **Goods**: Minimizes ACT
- **Not-so-goods**: Starvation
- Optimal? Why, or why not?

• Schedule?

012
Scheduling for “Average” Latency — (P)-SJF

- Shortest Job First + Preemption
- When is Preemption useful?

- Schedule?
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption

CPU bursts

0
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption

Assumption?
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption

The diagram illustrates the concept of improved resource utilization through the use of preemption. The green, red, and purple bars represent different resources or processes, and the numbers within the bars indicate the utilization or duration of each resource. The graph shows the allocation and reallocation of resources over time, demonstrating how preemption can lead to better utilization compared to non-preemptive scheduling.
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Improved Resource Utilization — Preemption
Scheduling for “Average” Latency — SRTF

• Shortest-Remaining Time First
• **Goods:** Minimizes Avg. ACT
• **Not-so-goods:** Starvation
• Optimal? Why, or why not?

• Schedule?
Scheduling for Fairness — RR

- Round Robin

$t=0$

10
4
8
Scheduling for Deadlines — EDF

- 4410 HW (11 hrs)
- Cooking (23 hrs)
- Calling Mom (18 hrs)

• Schedule?

012

10 8 4
Scheduling for Happiness

• Priority Scheduling

Your chosen schedule:

- 10
- 4
- 8

- 4410 HW
- Cooking
- Calling Dad
**“Universal” Scheduling**

Do we need to implement each and every policy?

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>FIFO</strong></td>
<td>Arrival time</td>
</tr>
<tr>
<td><strong>LIFO</strong></td>
<td>Current time - Arrival time</td>
</tr>
<tr>
<td><strong>SJF</strong></td>
<td>Job length</td>
</tr>
<tr>
<td><strong>SRTF</strong></td>
<td>Remaining job length</td>
</tr>
<tr>
<td><strong>RR</strong></td>
<td>?</td>
</tr>
<tr>
<td><strong>Priority</strong></td>
<td>Priority</td>
</tr>
</tbody>
</table>
“Mix-and-match” Scheduling

Different kind of jobs sharing the same OS ....

<table>
<thead>
<tr>
<th>Category</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactive</td>
<td>Facebook, Skype, ...</td>
</tr>
<tr>
<td>Batch</td>
<td>Data Analytics</td>
</tr>
<tr>
<td>Network bound</td>
<td>Downloading movies</td>
</tr>
<tr>
<td>CPU bound</td>
<td>Siri, Image processing, ...</td>
</tr>
<tr>
<td>Low priority</td>
<td>Life</td>
</tr>
</tbody>
</table>
“Mix-and-match” Scheduling

Different kind of jobs sharing the same OS ....

• Multi-level Queue Scheduling

• Each queue may implement a different policy
CPU Scheduling — Topics we did not cover

Many other scheduling problems within OS!

• Multi-processor scheduling
  • How to schedule tasks across multiple processors?

• Threads vs Processes
  • How to schedule threads?

• Jobs with dependencies
  • Job 2 can run only after Job 1 has finished...

• ....