Lecture 25:
How does a computer...
run many applications simultaneously?

Today’s Mystery
How does computer run multiple applications on one set of hardware?

- CPU: processing unit (ALU + set of registers) and control unit (program counter)
- Random Access Memory
- Input/output devices

How are applications going to share?

Users Run Many Applications Simultaneously
Expect all to be running, doing work for you...

What is an Operating System?

Operating System (OS):
- Software that converts hardware into a useful form for many different applications
- Very complex: millions of lines of code, 1000 person-years

Any Scratch Program
Word processor
Web browsers

Users
Applications
Operating System
Hardware
How does OS help applications share hardware?

OS gives each application \textit{illusion} that it is only one running on hardware.

Manage hardware resources for applications.

What do you think the OS must provide?

Roommate Scenario

Imagine two roommates sharing double dorm room (1 bath).

What properties must hold to be \textit{as if} they have own room?

- Other person doesn’t vandalize their stuff.
- Other person doesn’t look through their personal stuff.
- Can use bathroom whenever you need.

Some properties about

- Protecting your stuff.
- Getting to do what you want.

How does this match Computer?

Multiple applications sharing same hardware.

General Requirements

- Protecting your \textit{stuff} = Protecting \textit{data} that resides in \textit{Memory}.
- Getting to do what you want = Running on \textit{CPU} when you want.

Specifics

- No vandalism of stuff = Another app can’t \textit{overwrite} your data.
- Can’t look through stuff = Another app can’t \textit{read} your data.
- Can use bathroom when needed = Run on \textit{CPU} when app has \textit{work} to do (not when sleeping).

Must handle \textit{misbehaving} apps – Before harm occurs

- What if roommate won’t leave bathroom????
- Must have way to remove them against their will!

Terminology

OS executes \textit{”processes”} not applications.

- An application may be composed of multiple processes.

What is a process?

\textit{Execution stream} (i.e., instructions) in context of \textit{process state} (i.e., data).

- What you want to do, plus your stuff.

Find processing by running \textit{”ps”} (Unix-based).

- More processes running than you might expect!
Terminology

Multi-programming: Multiple processes resident in memory at same time
• Same as multi-tasking
Opposite: Uni-programming
• Only one process resident at a time
• Examples: First systems and DOS for PCs
Not multiprocessing: Multiple processors
Advantages: Better user convenience and performance
• Why does it improve performance???

Hardware Resources

1) How to share memory?

2) How to share CPU?

How to Share Memory?

Illusion provided by Operating System?
• Each process has all of physical memory to itself
Reality: Reside in physical memory at same time
Technique: Space sharing

Challenges with Sharing Memory

1) Ensure one process cannot r/w another process’s memory
• OS and HW cooperate to implement protection
  Translate memory references from logical to physical addresses
Challenges with Sharing Memory

2) Not enough physical memory for all address spaces
   - What policy should OS use to determine what is kept in memory and what is not?

Process 1
Process 2
Logical View of Address Space
Everything process can address thru memory including data and code

Process 3
Physical View

How to Share CPU?

Process alternates between CPU and I/O
- I/O: wait for user input
- Analogy: Alternate between bathroom and bedroom
Time sharing: Switch quickly between processes

OS and HW together perform context switch
- Change contents of registers and Program Counter (PC)
  - Stored in memory when not running (only OS can read this memory)
- Change active address space in memory
  - One process should not be able to read data of another process!

Perform context switches at different points
- When one job waits for I/O, switch to new job
- When one job has been using CPU too long, switch
  - Prevents one process from hogging CPU

How to Share CPU?

If only one wants CPU, no problem

What do you do if multiple processes want CPU at same time?

OS must also implement policy
- Many processes want to run, but which should run when??

Example:
Service at a Deli

Many customers waiting for service at deli...
In what order should customers be handled?
Easiest Policy?
First-come-first-served (FCFS)

How to implement this policy?
• Customers take ticket when arrive, serve next number
• Customers add to end of line, serve next customer in line

Why do you think this a good policy?
• Easy to implement
• Intuitively Fair: Earlier you arrive, sooner you get service

Why could it be bad?
• Treats all customers identically but may have different requirements

Different Requirements?
1) Some customers have a deadline
2) Some customers are more important than others
3) Some customers have short orders, others very long orders

1) Scheduler for Handling... Deadlines?
Earliest Deadline First
• Ask everyone when need to be done by
• Serve costumer with next deadline (search for min!)

Examples in Real World?
• Sometimes in long lines for airline check-in

What is good?
• Everyone finishes by when they need to

What is bad about this approach?
• Not fair: Works best when everyone works together
• Needs knowledge and trust: When is your real deadline?
• Impossibility: Might not be able to meet all deadlines

Earliest Deadline First

In Computer Systems?
• Used for “real time” and “embedded” applications

Control system must periodically perform different tasks
• Check different sensors (temp, speed, location, battery life)
• Adjust different controls (rotation, power)

How to determine deadlines?
• Every task needs to run periodically at same interval
• Next deadline = last time ran + interval
• Some sensors and controls more important than others
  – Check/control them more frequently...
2) Scheduler for Handling Important Customers?

Priority-based scheduling
- Allow important customers to move to front of line

Examples in real life?
- First-class in airlines, Fast Pass at Amusement Parks

Advantages?
- Give fastest service to most important customers (make them happiest)

Disadvantages?
- Less important customers can starve
  - Might never receive service if many important customers keep arriving
  - Extreme of “unfair”
- Determining who is “important” can be difficult
  - Spend the most money? Influence the most other people? Angriest?

3) Scheduler for Handling Short Jobs?

“Shortest Job First” (SJF)
- Figure out which customer has shortest order
- Let shortest orders go to front of line (search)

Examples in Real World?
- Decide to interrupt counter person with question…
- Separate lines for “10 items or less”

Advantages
- Creates optimal schedule for average waiting time
  - Minimizes average waiting time over all customers
  - Moving short job before long job:
    - Improvement in wait time of short job > Penalty to long job

Disadvantages
- Costumers with many items can starve; unfair
- How can you tell how length of job will take? Incentive to lie!
  - Must solve for this to work in computer systems!
  - How???

Priority Scheduling

In Computer Systems…
Which processes should be given high priority?

Give higher priority to system processes
- Responsible for keeping machine running

Give higher priority to “interactive” processes
- Processes user is currently “interacting” with
  - Give priority to which process they are typing to
  - Give priority to which process is creating output

Big Idea:
Use Past to Predict Future

Processes behave in future similarly to past (just like people?)
- Did this process use CPU for small time in the past?
- Use info to schedule short CPU bursts

Remember: Process alternates btwn CPU and I/O
(e.g., wait for user input)
**Details: Implementation in OS**

**Multi-level feedback queue**
- Multi-level queue: N lists of different priorities
- Feedback: Move to queue based on length of last CPU burst

**OS runs process at highest (priority) queue**
- Processes in same queue scheduled **Round Robin**
  - Switch out jobs hogging CPU
- Can also fix starvation problem by moving up processes not scheduled for awhile...

**Today’s Summary**

Operating System: Software that manages hardware
- Provides illusion to each process that it’s only one running
  - Context switches CPU across processes (Time share)
  - Protects memory across processes (Space share)
- Scheduling policies for CPU:
  - First-come-first-served (FCFS), Earliest-deadline-first, Priority-based, Shortest-Job-First (SJF)
  - Use past behavior to approx SJF: Multi-level Feedback Queue

**Reading**
- Section 6.4 of Invitation to CS

**Announcements**
- HW 6 due before class Friday: Lists in Scratch