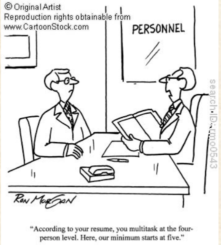
UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

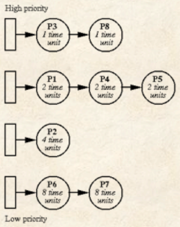
CS 202
Introduction to Computation

Professor Andrea Arpaci-Dusseau
Fall 2010

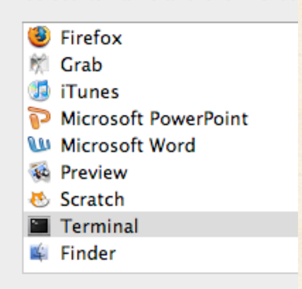
Lecture 25:
How does a computer…
run many applications simultaneously?



"According to your resume, you multitask at the four-person level. Here, our minimum starts at five."



Users Run Many Applications Simultaneously



- Firefox
- Grab
- iTunes
- Microsoft PowerPoint
- Microsoft Word
- Preview
- Scratch
- Terminal
- Finder

Expect all to be running, doing work for you…

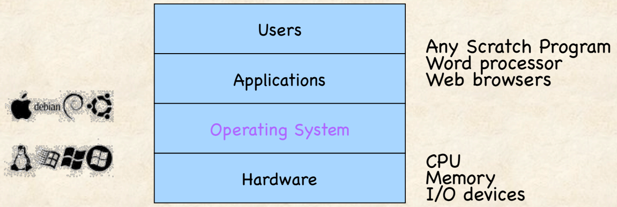
Today's Mystery

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?

What is an Operating System?



Users

Applications — Any Scratch Program, Word processor, Web browsers

Operating System

Hardware — CPU, Memory, I/O devices

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

How does OS help applications share hardware?

OS gives each application illusion that it is only one running on hardware

Manage hardware resources for applications

What do you think the OS must provide?

Roommate Scenario

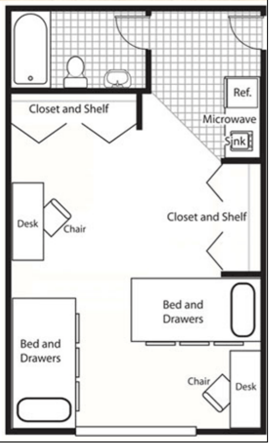
Imagine two roommates sharing double dorm room (1 bath)

What properties must hold to be *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 stuff = Protecting data that resides in Memory
- Getting to do what you want = Running on CPU when you want

Specifics

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

Must handle misbehaving apps – Before harm occurs

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

Terminology

OS executes "processes" not applications

- An application may be composed of multiple processes

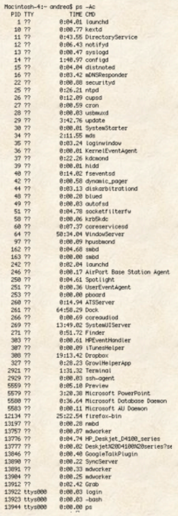
What is a process?

Execution stream (i.e., instructions) in context of process state (i.e., data)

- What you want to do, plus your stuff

Find processing by running "ps" (Unix-based)

- More processes running than you might expect!



```
Macintosh-4:~ andrea$ ps -Ac
  PID TTY           TIME CMD
    1 ??         0:04.01 launchd
   10 ??         0:00.77 kextd
   11 ??         0:43.55 DirectoryService
   12 ??         0:06.43 notifyd
   13 ??         0:00.47 syslogd
   14 ??         1:40.97 configd
   15 ??         0:04.04 distnoted
   16 ??         0:03.42 mDNSResponder
   22 ??         0:00.08 securityd
   25 ??         0:26.21 ntpd
   26 ??         0:12.09 cupsd
   27 ??         0:00.59 cron
   28 ??         0:00.03 usbmuxd
   29 ??         3:42.76 update
   30 ??         0:00.01 SystemStarter
   34 ??         2:11.55 mds
   35 ??         0:03.24 loginwindow
   36 ??         0:00.01 KernelEventAgent
   37 ??         0:22.26 kdcmond
   39 ??         0:00.01 hidd
   40 ??         0:14.02 fseventsd
   42 ??         0:00.58 dynamic_pager
   44 ??         0:03.13 diskarbitrationd
   48 ??         0:00.20 blued
   49 ??         0:00.03 autofsd
   51 ??         0:04.78 socketfilterfw
   58 ??         0:00.06 krb5kdc
   60 ??         0:07.37 coreservicesd
   64 ??        50:34.04 WindowServer
   97 ??         0:00.09 hpusbmond
  162 ??         0:04.68 smbd
  163 ??         0:00.00 smbd
  242 ??         0:02.04 launchd
  246 ??         0:00.17 AirPort Base Station Agent
  250 ??         0:04.61 Spotlight
  251 ??         0:00.36 UserEventAgent
  253 ??         0:00.00 pboard
  260 ??         0:14.94 ATSServer
  261 ??        64:58.29 Dock
  266 ??         0:00.69 coreaudiod
  269 ??        13:49.02 SystemUIServer
  271 ??         0:51.72 Finder
  303 ??         0:00.61 HPEventHandler
  307 ??         0:00.09 iTunesHelper
  308 ??        19:13.42 Dropbox
  327 ??         0:20.23 GrowlHelperApp
 2921 ??         1:31.32 Terminal
 2929 ??         0:00.03 ssh-agent
 5559 ??         0:05.10 Preview
 5579 ??         3:20.38 Microsoft PowerPoint
 5580 ??         0:36.64 Microsoft Database Daemon
 5583 ??         0:00.11 Microsoft AU Daemon
12134 ??        25:22.54 firefox-bin
13197 ??         0:00.20 nmbd
13757 ??         0:00.87 mdworker
13776 ??         0:04.74 HP_Deskjet_D4100_series
13777 ??         0:00.02 Deskjet%20D4100%20series?se
13846 ??         0:00.40 GoogleTalkPlugin
13890 ??         0:00.22 SyncServer
13891 ??         0:00.33 mdworker
13904 ??         0:00.25 mdworker
13912 ??         0:02.42 Grab
13922 ttys000    0:00.03 login
13923 ttys000    0:00.03 -bash
13944 ttys000    0:00.00 ps
```

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?

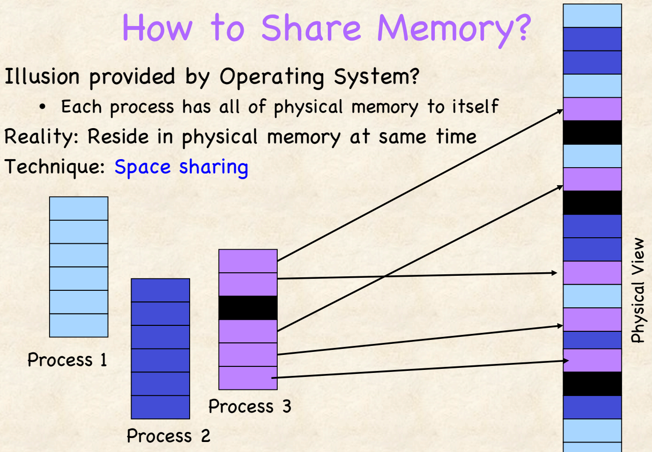
How to Share Memory?

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

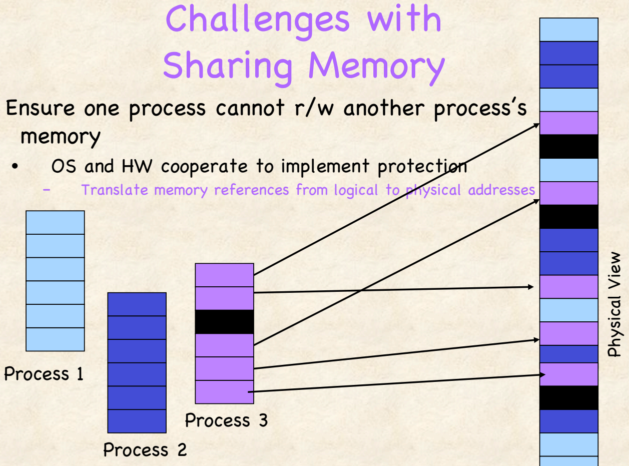


Logical View of Address Space
Everything process can address thru memory including data and code

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

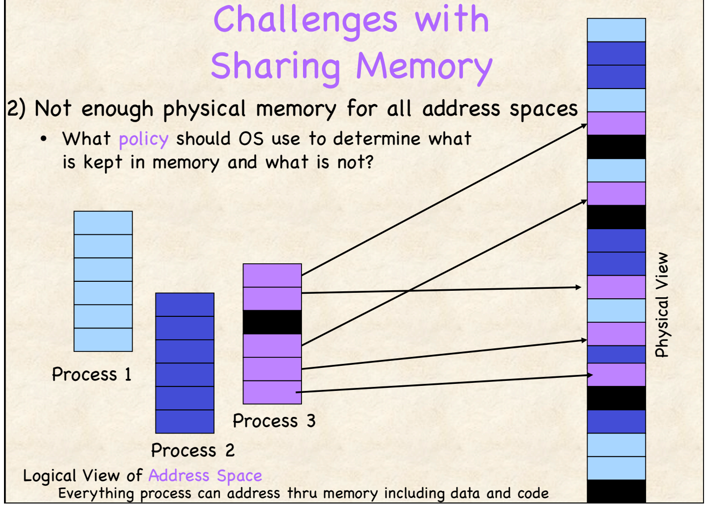


Logical View of Address Space
Everything process can address thru memory including data and code

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?



Logical View of Address Space

Everything process can address thru memory including data and code

How to Share CPU?

Process alternates btwn CPU and I/O

- I/O: wait for user input
- Analogy: Alternate btwn 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

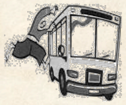
Different Requirements?

1) Some costumers have a deadline

2) Some customers are more important than others

3) Some costumers 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?

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

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

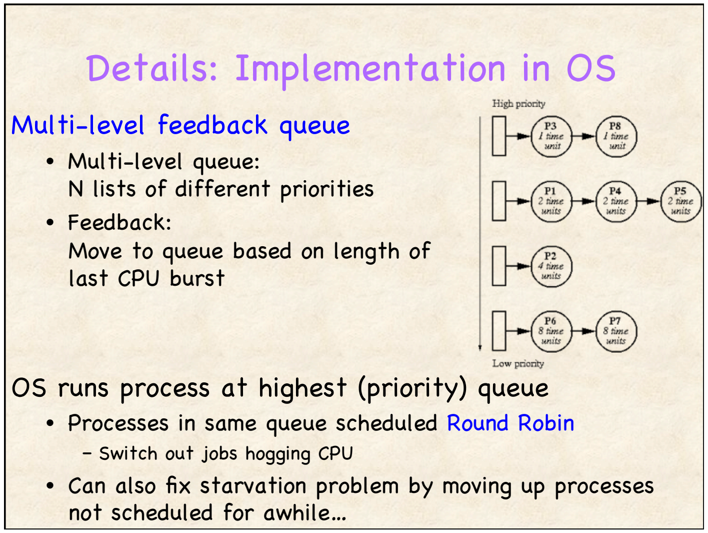
- Costumers with many items can starve; unfair
- How can you tell how length of job will take? Incentive to lie!
  - Must solve for this to work in computer systems!
  - How???

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)



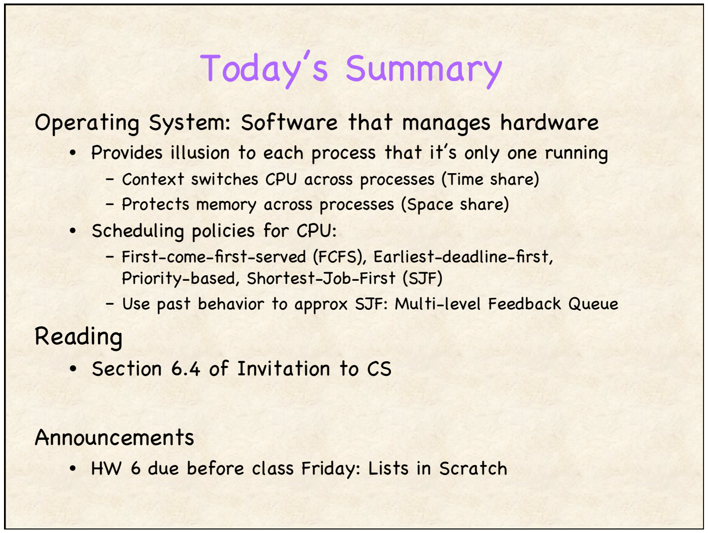
Details: Implementation in OS

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