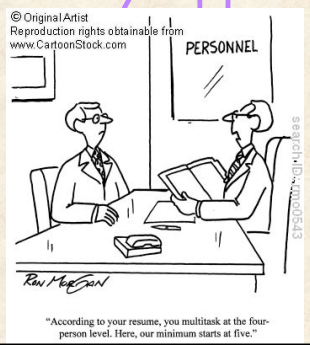
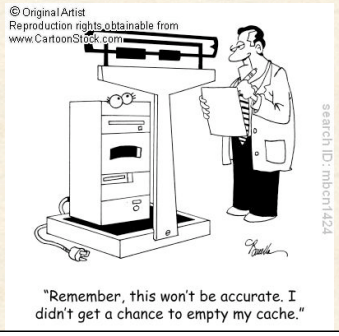
UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

CS 202
Introduction to Computation

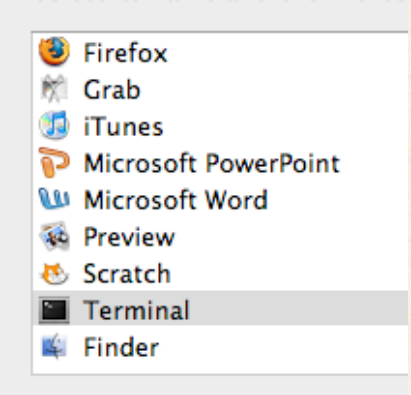
Professor Andrea Arpaci-Dusseau
Spring 2010

Lecture 26: How does a computer… run many applications simultaneously?



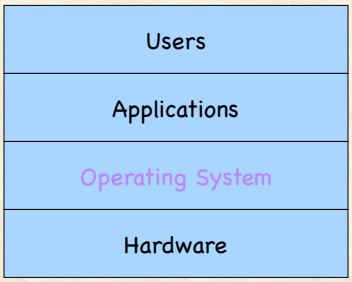


Users Run Many Applications Simultaneously



Expect all to be running, doing work for you…
How does computer do this with only 1 (or 2) CPUs?

What is an Operating System?



compilers
databases
word processors
web browsers

CPU
memory
I/O devices

Operating System (OS):
Software that converts hardware into a useful form for applications

Very complex: millions of lines of code, 1000 person-years

What does OS do?

Manage hardware resources for applications

What do you think this entails?

Role #1: Provide standard library for accessing resources
- Allow applications to reuse common facilities
- Make different devices look the same
- Provide higher-level abstractions

Role #2: Coordinate usage of resources (i.e., manager)
- Virtualize resources so multiple users or applications can share
- Protect applications from one another
- Provide efficient and fair access to resources

What are different HW Resources?

CPU: Process scheduler
- Determines when and for long each process executes

Memory: Memory manager
- Determines when and how memory is allocated to processes
- Decides what to do when main memory is full

Disk: File system
- Organizes named collections of data in persistent storage

Network: Networking
- Enables processes to communicate with one another

How to run multiple applications?

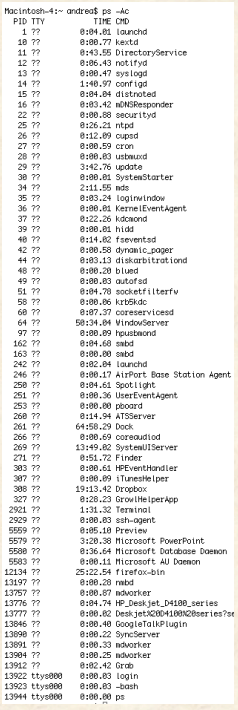
OS executes "processes" not applications
- Related to application, but not identical

What is a process?
Execution stream in context of process state

Informally:
Everything you need to run an application

More processes running than you might expect!

Find by running variants of "ps" (Unix-based)



```
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 ??         0:00.88 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 ??        64:50.29 Dock
  266 ??         0:00.69 coreaudiod
  269 ??        13:49.82 SystemUIServer
  271 ??         0:51.72 Finder
  303 ??         0:00.61 HPEventHandler
  307 ??         0:00.09 iTunesHelper
  308 ??        19:13.42 Dropbox
  327 ??         0:28.23 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 ??         0:00.28 nmbd
13757 ??         0:00.87 mdworker
13776 ??         0:04.74 HP_Deskjet_D4100_series
13777 ??         0:00.02 DeskjetX300D4100N2Bseries?se
13846 ??         0:00.40 GoogleTalkPlugin
13898 ??         0:00.22 SyncServer
13899 ??         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
```

How to run multiple processes?

Multi-programming: Multiple processes resident at a time

- Same as multi-tasking
- Opposite: Uni-programming
  - Only one process resident at a time
  - Examples: First systems and DOS for PCs
- Different than multiprocessing
  - Multiprocessing: Systems with multiple processors
- Advantages: Better user convenience and system performance
  - Why does it improve system performance???



How to Support Multiprogramming?

OS provides illusion that each has HW to itself

Must be able to handle misbehaving applications (that don't want to share!)



How to Share CPU?

Illusion?

- Switch quickly between active processes: Time sharing
- Processes must not be able to hold onto CPU forever!

OS and HW work together to perform context switch

- Change contents of registers and Program Counter (PC)
- Change active address space in memory

Context switch is mechanism

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)

- 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 othe

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 next by deadline that must be met

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?
- Might not be able to meet all deadlines

Use in computer systems…

- Used for "real time" applications (control systems and video)

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?

Computer systems

- Give higher priority to system processes
- Higher priority to "interactive" processes

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

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

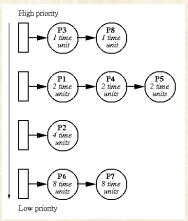
Processes behave in future similarly to behavior in past (just like people?)

- Did this process use CPU for small time in the past?
- Use info to schedule short bursts
- Past doesn't mean yesterday – refers to this execution of process

Job: Different than "application" or "process"

- Process alternates btwn CPU and I/O (e.g., wait for user input)
- Job is the CPU burst
- Measure past CPU bursts of one process

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 FCFS
- Can also fix starvation problem by moving up processes not scheduled for awhile…

Motivation for Sharing Memory: Cutting-Edge Libraries

Why is this relevant to sharing memory?

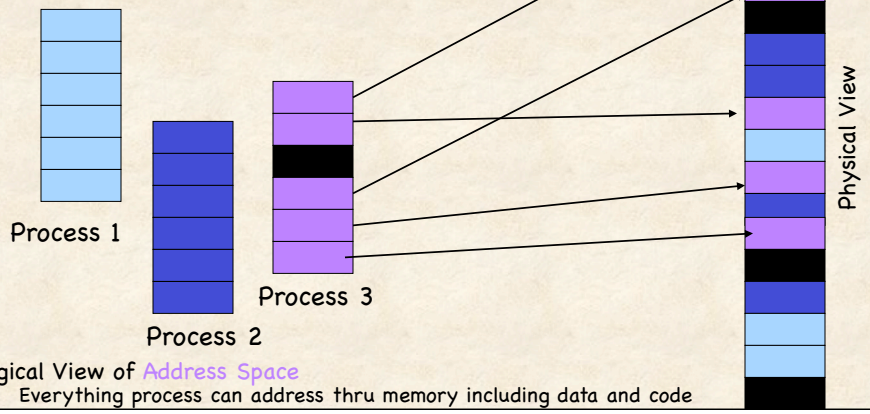
How to Share Memory?

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



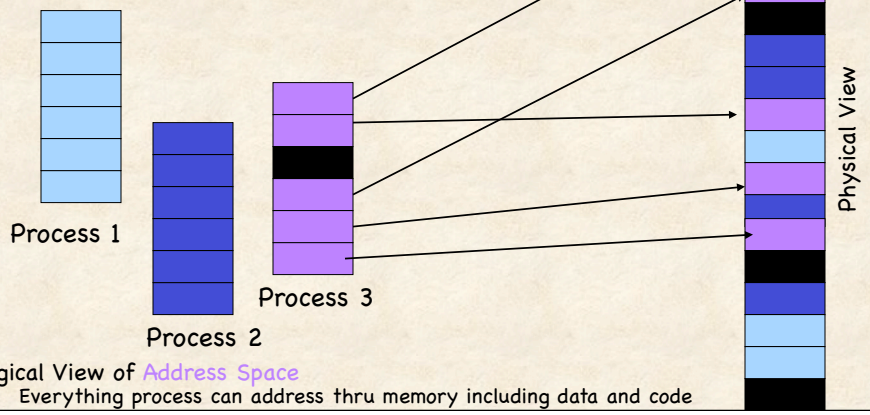
Logical View of Address Space

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 mechanism
  - Translate memory references from logical to physical addresses
- Not focus of today's lecture



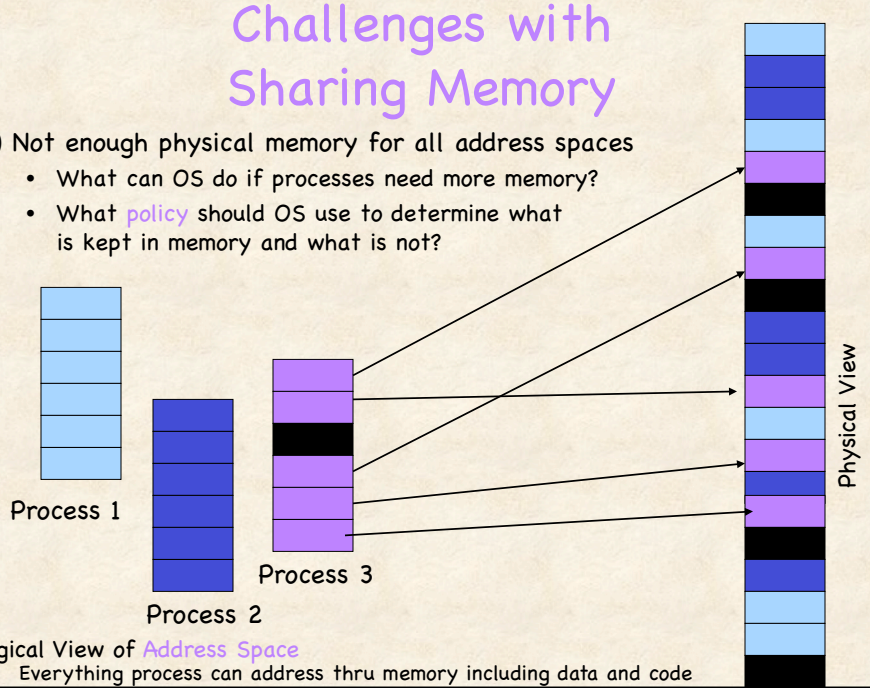
Logical View of Address Space

Everything process can address thru memory including data and code

Challenges with Sharing Memory

2) Not enough physical memory for all address spaces

- What can OS do if processes need more memory?
- What policy should OS use to determine what is kept in memory and what is not?



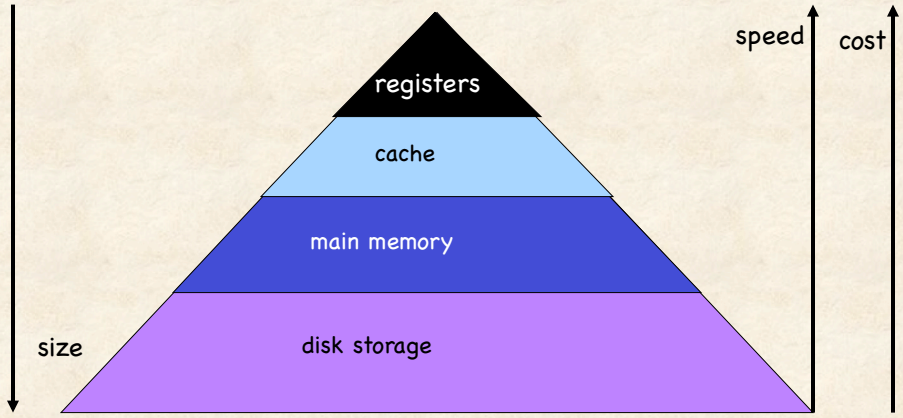
Logical View of Address Space

Everything process can address thru memory including data and code

Memory Hierarchy

Leverage memory hierarchy of machine architecture

- If doesn't fit in one level, move to next level down



Sizes of Memory Hierarchy

Challenge

What parts of each address space should OS keep in main physical memory vs. disk storage?

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
- Caching policies for Memory:
  - Speed of fastest memory; Capacity largest memory
  - Optimal Replacement Algorithm requires knowledge of future
  - Use past to predict future (Least-Recently-Used)

Announcements

- HW 7 due before class Friday: Design of Project 2
- Working on Project 2