Lecture 1: Course overview

David Hovemeyer

January 27, 2020

601.229 Computer Systems Fundamentals

Welcome!

- Welcome to CSF!
- Today:
  - Administrative stuff
  - Course overview
  - Binary data representation

Administrative stuff

About the course

- Instructors
  - Xin Jin, xinjin@cs.jhu.edu, Malone 223
  - David Hovemeyer, daveho@cs.jhu.edu, Malone 337
- CAs
  - Coming soon, see course web page for details

Where to find stuff

- Course website: https://jhucsf.github.io/spring2020
  - Syllabus, schedule, lecture notes, assignments, etc.
  - All public course information will be here
- Piazza https://piazza.com/jhu/spring2020/601229
  - Non-public course information such as homework/exam solutions
  - Discussion forum, Q/A: please post questions here!

Syllabus highlights

- Please read the syllabus carefully:
  `https://jhucsf.github.io/spring2020/syllabus.html`
- Highlights:
  - Grades: 55% homework, 40% exams, 5% clicker quizzes
  - 6 or 7 assignments, mostly programming based, expect them to be challenging!
  - Late policy: you have 5 late days to use as needed (assignment submissions which exceed the late day limit receive no credit)
  - One midterm exam, one final exam
    - Final exam will cover material covered after midterm

Academic integrity

- Please read the academic integrity policy in the syllabus carefully
- Highlights:
  - Follow the CS Academic Integrity Code: https://www.cs.jhu.edu/academic-integrity-code/
  - Homework assignments are done individually, code sharing is not allowed
  - Exams are (obviously) individual effort
  - Violations of academic integrity will be reported to the Student Conduct office
- Be careful about using web as a resource
  - Do *not* copy code
  - *Always* cite sources used

Class meetings

- Typical class meeting: lecture/discussion interspersed with peer instruction questions and occasional group activities
- *Do the reading in advance!*
- Come prepared to actively engage with the material!
  - Learning is not passive
  - More productive class time $\rightarrow$ better outcomes
  - Ask questions!

Peer instruction

- How peer instruction works:
  - Slide with a multiple choice question
  - Answer individually, discuss with peers, then answer again
  - Shown to improve outcomes!
  - Questions may be challenging
  - Graded for participation only
- You may have done this in other courses
- You will need an iClicker 2
  - Use the google form linked from the Piazza resources page to register your iClicker remote ID

Peer instruction etiquette

- During discussion phase, form a group of 2–4 with people sitting near you
- Be inclusive! ("Would you like to join our group?")
- Be social! ("May I join your group?")
- Be respectful:
  - Let everyone participate
  - Don't put down anyone else's ideas
- Work together and think carefully about the question!
- *No electronics use*

First clicker quiz!

Clicker quiz omitted from public slides

Computing requirements

- All assignments will be done using Linux
- Autograder will use Ubuntu 18.04
- Doing your development and testing on ugrad machines is generally fine (but note that compiler and other tools will likely be different than Gradescope)
- Development on Windows or MacOS is not recommended
  - Although Windows Subsystem for Linux is *probably* ok
- Running Linux in a VirtualBox VM highly recommended: see Resources page on course website
- *Get your development environment set up NOW*

Course overview

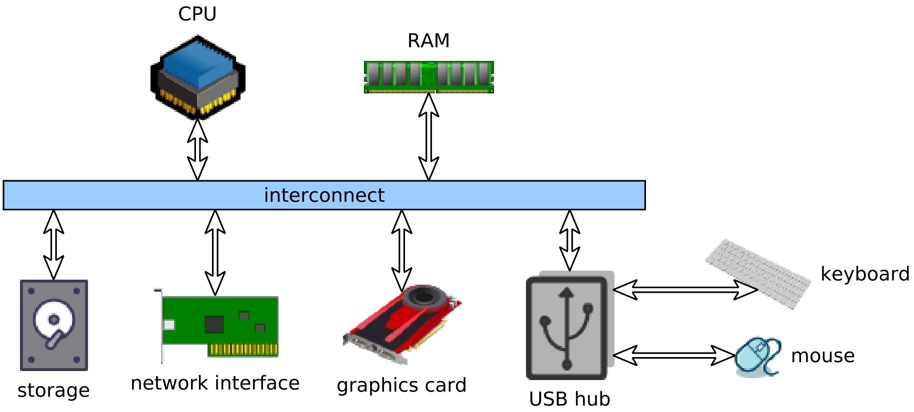
What the course is about

- Course is about *computer systems* from the *programmer's perspective*
- Computer system = hardware + software
  - Much of our concern is the interaction between hardware and software — how they work together

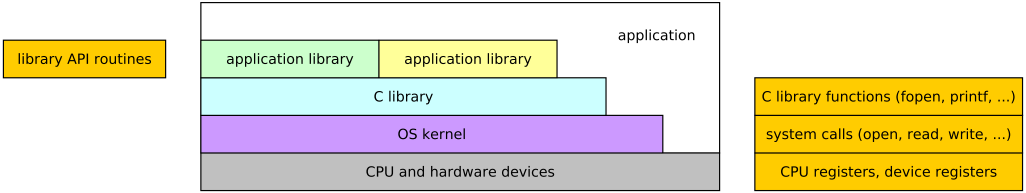
Goals of course

- "Deep" understanding of how computers work (down to hardware)
  - OS and runtime library interfaces
  - Machine-level ISA / assembly language
  - Processor features
  - Operating system features
- Apply this understanding to...
  - Optimize application performance
  - Avoid pitfalls such as security vulnerabilities
  - Take full advantage of the computer's and operating system's capabilities

A computer system (hardware)

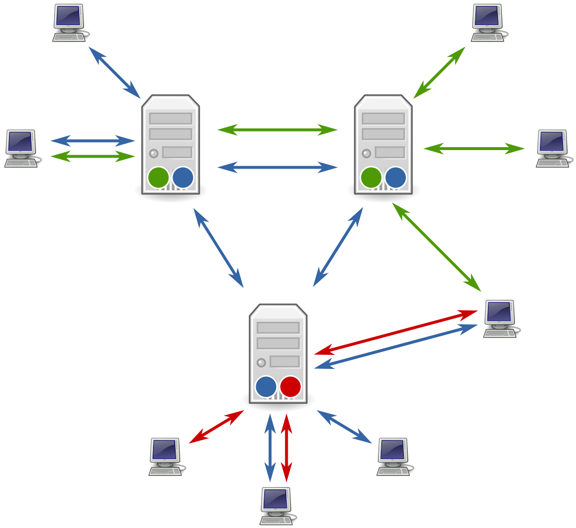


A computer system (software)



- Your application program is supported by lower layers of software and hardware
- Each layer provides an interface to the layer above

A computer network



Computer networks allow your program to communicate with peer systems.

Thanks to the global Internet, the peer systems could be anywhere on earth!

Binary data representation

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:
  - A *continuous* representation would allow the number to have *any* value
    - We think of physical phenomena (mass, velocity, etc.) as being continuous

Discrete data representation

- Digital computers use a *discrete* representation for all data
- Consider a representation of a number:
  - A *continuous* representation would allow the number to have *any* value
    - We think of physical phenomena (mass, velocity, etc.) as being continuous
  - A *discrete* representation would allow the number to have one of a set of possible values, where the set of possible values is *enumerable*
    - Often we think of discrete values as corresponding to a range of integers

Why discrete representation?

- Why do digital computers use a discrete representation for all data?

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously
- OK, let's think about what discrete data representations will look like...

Why discrete representation?

- Why do digital computers use a discrete representation for all data?
- Answer: internally, information is represented using *digital voltages*
  - High voltage (1) vs. low voltage (0)
  - Digital circuits (with discrete high vs. low voltages) have many advantages over *analog* circuits, where voltages can vary continuously
- OK, let's think about what discrete data representations will look like...
  - Starting with *integers* (if you can represent integers, you can represent anything)

Decimal numbers

- We're all familiar with decimal (base 10) numbers
- E.g.,

$$42 = 4 \cdot 10^1 + 2 \cdot 10^0$$

  - Digits are 0–9
  - Places are powers of 10

Other bases

- Base 10 is arbitrary!
- Representing decimal 42 using base 5:

$$42_{10} = 132_5 = 1 \cdot 5^2 + 3 \cdot 5^1 + 2 \cdot 5^0$$

  - "Digits" are 0–4
  - Places are powers of 5

Try it!

How to express decimal 42 using base 6?

$$\_\_\_ \cdot 6^2 + \_\_\_ \cdot 6^1 + \_\_\_ \cdot 6^0$$

How to express decimal 79 using base 6?

$$\_\_\_ \cdot 6^2 + \_\_\_ \cdot 6^1 + \_\_\_ \cdot 6^0$$

Reference:

$$6^2 = 36$$
$$6^1 = 6$$
$$6^0 = 1$$

Binary

- Binary = base 2
- Representing decimal 42 using base 5:

$$\begin{aligned} 42_{10} &= 101010_2 \\ &= 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 \end{aligned}$$

  - "Digits" are 0 and 1
  - Places are powers of 2
- Computers use binary representations for all data, because
  - *Digital circuits* use two voltage levels, high and low
  - By convention, 1=high voltage, 0=low voltage
  - So, computer hardware fundamentally operates on binary data

Try it!

How to express decimal 29 using base 2?

$$\_\_\_\_ \cdot 2^5 + \_\_\_\_ \cdot 2^4 + \_\_\_\_ \cdot 2^3 \ \_\_\_\_ \cdot 2^2 + \_\_\_\_ \cdot 2^1 + \_\_\_\_ \cdot 2^0$$

Reference:

$$2^5 = 32$$
$$2^4 = 16$$
$$2^3 = 8$$
$$2^2 = 4$$
$$2^1 = 2$$
$$2^0 = 1$$