CSE 550: Systems for All

Ratul Mahajan ratul@cs

Emmanuel Azuh Mensah emazuh@cs
Who we are
Emmanuel
Ghana -> MIT -> Amazon -> UW ICTD

Research: Intersection of Systems ML & International Development w/ Kurtis Heimerl & Luis Vega (SAMPA Lab)

Hobbies: Biking, Kayaking, Cooking, Dancing! Dabbling in philosophy & nature photography

Fun fact: Once met Bill Gates while working on ML application in agriculture at MSR
Course structure
CSE 550: Systems for All

- Quals course that covers foundational systems topics from:
  - Operating Systems, Networks, Distributed Systems, Databases
- No “real” prerequisites
  - Designed to allow first-year undergrads from other areas to engage
  - Functional knowledge of real systems will be helpful
- Gateway course to CSE 551, 552, and 561 or a terminal course for students desiring breadth
- Goal: A thorough understanding and appreciation of the work your systems colleagues are doing!
Course organization

1. Read papers
   - Deeply read 1-2 per class
   - Shallow reads (optional)

Check out: https://derekchia.com/how-to-read-a-research-paper-3-pass-approach/
   - Deep read = All three passes
   - Shallow read = 1st pass
Course organization

2. Discuss papers
   • For each lecture, we will post a small set of questions on the assigned papers.
     • We'll create one thread per discussion question set.
     • You're required to add a comment to the discussion for each of the threads by 9am on
       the day of the class (to give time for everyone to read the responses before class.)
     • For each thread, pick one of the questions in the question set to answer.

   • Free-form discussion
Course organization

3. Lead paper’s discussion in a group of 2
   • Sign up sheet on the course Web page

Ratul will do a short intro to the topic prior to that
Course organization

4. Projects

- Two programming/measurement assignments
  - Done in groups of 2 or 3
  - Meant to both clarify the nature of systems contributions as well as engage you in tooling issues (a defining feature of Systems)
  - Assignment #1 is already posted
- Project 3
  - Choose: Independent research project or in-depth assignment from us
    - Implement Paxos!
  - Same groups of 2 or 3
  - Both conclude with writeup of results.
Course organization

No exams!
Tools

• Canvas for projects/assignments
• Ed for discussions
• Slides posted on the Web page after the lecture
  • Adapted from Arvind and Kurtis
• Feedback: email/chat, mid-quarter eval, feedback.cs.washington.edu
Office hours

• Opportunity to have more personal interactions with us
  • Ratul: On-demand by Zoom; around after class
  • Emmanuel: 4-5 pm Wed
Grading

• Online discussions: 10%
• In-class discussion leading and participation: 10%
• First two projects: 25% + 25%
• Project 3: 30%
Late policy, getting off-track

Each person gets three late days for reading responses.

If you projects/assignments will be late, reach out to us beforehand and we’ll figure it out.

Special circumstances: Come talk to us.
Questions?
Course content
system

/ˈsɪstəm/

See definitions in:

All  Physiology  Computing  Science  Gambling  Music

noun
plural noun: systems

1. a set of things working together as parts of a mechanism or an interconnecting network.  
   "the state railroad system"
   Similar: structure  organization  order  arrangement

2. a set of principles or procedures according to which something is done; an organized scheme or method.  
   "a multiparty system of government"
   Similar: method  methodology  technique  process  procedure

Definitions from Oxford Languages  Feedback

Translations and more definitions
Our focus

*Software* systems
• Though the overlap with hardware is large!

Understand how software systems achieve a specific external behavior
• e.g., deliver videos, online social network, email, ML execution

Comprises of many components
• Components interact and cooperate to provide overall behavior
• They typically have (well) specified interfaces

Large-scale, running across thousands or millions of hardware devices
Let’s build a social network together

Assume that you have computer hardware
Common themes across systems

Correctness – does it work as advertised?
Reliability – stay functional as components fail
Performance
Scalability
Efficiency
Security