Welcome to CSC 2227S

Topics in the Design and Implementation of Operating Systems

Spring 2019
Preparation

• Overview of CSC 2227S
  • How the course will operate
  • What I expect from you

• Goals and Topics

• Review distributed systems basics

• What’s next...
Overview of 2227S (Spring 2019)

• Check the web page for updates and news frequently
  • http://www.cs.toronto.edu/~demke/2227/S.19/

• Components
  • Written summaries of papers (before class)
  • Critical study and discussion of systems papers
    • Student-led paper presentations
    • Occasional background mini-lectures (maybe not...)
  • Term project

• Other stuff
  • No assigned books, but some on you might find useful (list on web page)
  • Prereqs
  • Grading plan
Making the grade in 2227

• Generally
  • Put in the effort and your grade will take care of itself

• Breakdown
  • 50% project
  • 20% paper summaries
  • 20% paper presentations
  • 10% class discussions

• Caveat/warning
  • This is an advanced graduate-level course, which means lots of effort on your part and less structure than undergraduate courses
  • If you dive into it, you’ll learn lots and love it!
Prereqs

• Prereq: undergraduate OS
  • You should have a solid command of this material
  • If you don’t, you will struggle
  • Worse, you will not benefit as much as you should

• Prereq: some knowledge of advanced OS topics
  • OS structure, perf. eval., synchronization, distributed system models (see CSC469/CSC2208)

• Refresher questions – self-test
  • The point is to swap in your OS knowledge
  • use your OS book(s) from undergrad
  • discuss the problems and topics with your peers
  • now is the time to refresh your memory!
Paper reading and reviews

• 2 or 3 papers will be assigned for each week
  • You should read them carefully before the class
    • be prepared to recall and discuss their contents
  • You should type up a considered review before class
  • You should submit review before class starts
    • don’t be late or skip class to do this; participation counts too

• Review contents: about 0.25-0.5 of a page
  • List the three most important things the paper says (to you)
  • Describe the paper’s most glaring deficiency
  • Describe what the paper taught you about system building
  • DO NOT just repeat abstract or provide book report

• Grading
  • Complete/Incomplete
    • A very poor summary will be considered incomplete
  • Roughly 1% per paper (you can miss a few)
Paper Presentations

• Conference-style short presentation of paper
  • Problem, approach, outcome, related work
  • Connections
    • for older papers, where can you see the influence of this paper? What was the historical context?
    • For newer papers, what are logical extensions, applications, or next steps for the work?
• Plan for roughly 25 minute presentation + Q&A
  • Not necessarily in that order
  • Q&A more like leading a discussion than conference Q&A (you are not solely responsible for defending the paper; you can pose questions for the rest of the class to answer, etc.)

• Full schedule for term available by January 10
  • Bid for papers using hotcrp conference management system
How to read a research paper

• Consider the source (don’t dismiss based on source, but do take it into consideration)
  • Who wrote it -- are they experts or unknowns?
  • Where was it published -- top journal or personal web page?
  • Other aspects: sponsor, review process, structure, tone, etc.

• Dig for the point
  • Read the abstract, intro, conclusion and related work
  • Flip through the paper, looking at headings, figures and data, and bibliography
  • Consider how much time you really want to devote to the guts
  • What is the hypothesis, how do they try to prove it, and do they succeed?
How to read a research paper (2)

• Expect to read the paper more than once
  • First reading, get overall picture of the system and results
  • Second reading, decide what details to focus on

• Take notes as you read (be an active reader)
  • Write down questions to keep track of what you don’t understand
  • Refer back and add notes if your questions are answered later in the paper
    • Or on subsequent reading, or after discussion with peers
  • Question authors’ assumptions, importance of problem, important effects not mentioned by authors, etc.

• Don’t accept ideas / design details that you don’t understand
  • Authors’ assumptions and design choices should not be assumed correct simply because the paper was published.
  • May need multiple readings and discussion with peers
How to evaluate a research paper

• Important and relevant problem? Clever idea?
  • These can be orthogonal! A paper may be important because it clearly presents a straightforward idea to an important new problem. Clever ideas may find uses beyond the problem they are applied to in the paper.

• Are the assumptions and models reasonable?

• Has the system or idea been influential? (older)
  • Everyone uses systems derived from it
  • The idea shows up in many different contexts

• Is the potential impact clear? (newer)

• Does it help to make sense of complex phenomenon or area with many competing ideas?
  • Comparison studies or surveys
2227 Projects

• Practical experience is a must for understanding systems
  • Thus, you will be required to design, construct and evaluate an interesting software system

• What software system?
  • It’s up to you
    • You are encouraged to propose your own project idea
    • various project topic ideas will be posted on web page to help
    • Projects that span traditional sub-areas of CS/CE are great
  • ... but it must relate to 2227
    • At least one of the 2227 topics should be involved
    • must be explicitly okay’d

• Working in groups of 2 is encouraged
• Talk to me early if special equipment is required for the project you want to do
2227 Project Documents

- Project proposal (Feb. 20) - 5%
  - 2 pages describing your project idea and plan
- Project literature survey (March 6) - 5%
  - ~3 pages (+ bib) describing related work (~10 papers) and how it relates to your project
- Project design document (March 27) – 7%
  - 5 pages revising and detailing your project idea and plan
- Poster session (April 27) – 8%
- Project final report (April 30) - 25%
  - 12 pages describing the completed project, including the idea, the execution, the evaluation, and the related work
Course Communications

• Website, email and hotcrp
• HotCRP allows reviewers to add comments on papers
  • Can opt to receive email on every comment or not
Where to find papers (quick tangent)

• You should not feel limited to reading the papers I give you
• Great source: web search engines and on-line paper listings
• Another great source: library (ACM digital library is great!)
  • Every serious researcher should spend time looking for related papers
• Some good computer systems conferences
  • SOSP, OSDI, NSDI, EUROSY, ASPLOS, Usenix ATC, SIGMETRICS, SIGCOMM, ISCA, ...
• Some good computer systems journals
  • ACM Transactions on Computer Systems (TOCS)
  • IEEE Computer
  • Communications of the ACM (older issues)
  • IBM Systems Journal
Goals

• To understand the key problems in designing and implementing distributed systems and their solutions
  • Recent systems papers lean heavily toward networked systems.
  • This course should provide the background to read and understand the current research.
Topics (may be revised)

- Historical distributed systems
- Consensus
- Coordination Services
- Distributed Hash Tables
- Key-Value Stores
- In-memory computing and storage
- Distributed file systems
- Programming frameworks
- Serverless computing
- Scheduling and load balancing
- Distributed performance analysis & debugging
- Very large systems
Why Distributed Systems?

- Information exchange (WAN)
- Resource sharing (LAN)
- Parallelization to increase performance
- Replication to increase reliability
- Multicore programming
Distributed systems vs. Uniprocessors

Distributed systems differ from uniprocessor systems in three aspects.

• Lack of knowledge on the global state: A process usually has no up-to-date knowledge on the local states of other processes.

• Lack of a global time frame: No total order on events by their temporal occurrence.

• Nondeterminism: Execution of processes is nondeterministic, so running a system twice can give different results.
  • Example: Race conditions.
Review:

• Have a look at the self-study quizzes
  • These are fairly old and some terminology may be unfamiliar

• Dig out your old OS notes and review
  • Arpaci-Dusseaus have great online textbook:
    • Operating Systems in Three Easy Pieces (OSTEP):
      http://pages.cs.wisc.edu/~remzi/OSTEP/