Distributed Systems and Algorithms

CSCI 4510/6510    Fall 2019
General Information

- Lectures: TF 12pm - 1:50pm in Low 3051
- Instructor: Stacy Patterson (me) sep@cs.rpi.edu
- Office Hours: T F 2pm - 3pm in Lally 301 (or by appointment)

- TA: Tim Castiglia castit@rpi.edu
- TA Office Hours: W 1pm - 3pm in AE 127
Course Objectives

● This is a theory course.
● The goal is for you to learn important theory and algorithms for distributed computing systems
  ○ Through theory and practice
● These algorithms are the foundation of data center-computing and cloud computing
  ○ Used in practice today at Google, Twitter, Facebook, Hadoop, LinkedIn, ......
Pre-Requisites

● CSCI 2300: Intro to Algorithms
  ○ Analysis of algorithm correctness and performance
  ○ Writing correct and coherent proofs of algorithm properties

● CSCI 4210: Operating Systems
  ○ Familiarity with fundamental problems and features of concurrent programming
    ■ Mutual Exclusion, Deadlock, ..
  ○ Multi-threading programming
  ○ Socket programming
Course Materials

● Course content will be presented in lectures.
  ○ My aim is for these lectures to be self-contained.
  ○ I will not post lecture notes on the course web site.
  ○ Do not take pictures or record lectures without my permission before-hand.
  ○ If you miss a lecture, it is your responsibility to get the lecture notes from a classmate.

● The course content is based on seminal research papers.
  ○ Posted on the course web site.
  ○ The vast majority of these papers should be accessible to all of you.
Course Materials - Continued

- Optional supplementary textbook: *Distributed Systems and Concepts* by Coulouris et al.
  - Available in the bookstore.
  - May present different variants of algorithms and results than we cover in class.
  - You are responsible for learning the versions taught in lectures.
Grading

- In-Class Quizzes: 55%
- Take-Home Quizzes: 10%
- Programming Projects: 35%
  - Project 1 15%
  - Project 2 20%
- Quizzes will be graded in Gradescope.
- Project submissions will be handled in Submitty.
### Course Letter Grades (rounded to nearest integer)

<table>
<thead>
<tr>
<th></th>
<th>B+: 87 – 89</th>
<th>C+: 77 – 79</th>
<th>D+: 67 – 69</th>
<th>F: 0 – 59</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-: 90 – 92</td>
<td>B-: 80 – 82</td>
<td>C-: 70 – 72</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The average course grade is usually around a B+.
- I may lower the grade cutoff points. I will not raise them.
- There will be different curves for 4510 and 6510.
In-Class Quizzes

- There will be 6 in-class quizzes.
  - Quiz dates are posted on the website
- Quizzes will be:
  - Closed book
  - About 45 minutes each
  - Done independently
- Quizzes are meant to evaluate your understanding of the algorithms, not test your memorization skills.
  - Pseudocode for long algorithms will be given with the quiz.
- No makeup quizzes will be given without an official excused absence from RPI.
- Regrade requests must be made within 7 days of quiz return.
Take-Home Quizzes

● There will be 3 take-home quizzes
  ○ 2 to 3 questions
  ○ Comprehensive
  ○ Open notes, book, internet
  ○ But no collaboration
● Questions will be similar in style to the in-class quiz questions.
● Will be assigned in class, and due before the following lecture
  ○ Questions will be posted on the website
  ○ Submissions done via Gradescope
Programming Projects

● There will be 2 programming projects.
● Projects will be done in groups of 2.
  ○ CSCI 4510 students must pair up with other CSCI 4510 students.
  ○ CSCI 6510 students must pair up with other CSCI 6510 students.
● Projects will give you the chance to implement distributed algorithms in a distributed computing system.
  ○ We will use Docker containers
  ○ Projects can be done in Java or Python.
● You will have about 4 weeks for each project.
Programming Projects: Project 0

- There is a Project 0 posted on the course web site.
  - This is a self-test.
  - It will not be graded.
- This project covers the basic programming skills you will need for this course. If you do not know how to complete this project, be prepared to put in lots of time on the course projects.
Special Accommodations

- If you need special accommodations for this course, please let me know at least 2 weeks before the affected assignment (quiz).
  - This means now.
Academic Integrity Policy

- No collaboration is allowed on quizzes.
- For programming assignments, you may discuss the project with other students, but you (your team) must write your own code.
  - Use of external libraries must be approved by me in advance.
  - We will use plagiarism checking software on your project code.
- We will discuss the academic integrity policy for exams closer to the final exam date.
- Any student who violates these policies will be subject to penalties outlined in the Rensselaer Student Handbook.
Introduction to Distributed Systems
What is a distributed system?

- “A distributed system is one in which components located at networked computers communicate and coordinate their actions only by passing messages.”
  - Coulouris et al., Distributed Systems

- Significant characteristics
  - Concurrency: Different operations execute on different computers at the same time.
  - No global clock: Difficult to synchronize (coordinate) actions on different computers.
  - Independent failures: computers can crash, the network may fail or slow down, network partitions may arise.
  - The rest of the system keeps running, may not be aware of failures.
I want the application to behave like:

- It is running on a single computer with infinite resources that never fails
- And I am the only one using the application
The application is actually

- Running on hundreds (thousands?) of computers
- Spread across multiple data centers, on multiple continents
- With thousands (or more) simultaneous users.
Google’s Data Center Locations
Data Center Architecture

Server

Rack = 40 to 80 servers

cluster
Inside a Data Center
The Horrible Truth...

Typical first year for a new cluster:

~1 network rewiring (rolling ~5% of machines down over 2-day span)
~20 rack failures (40-80 machines instantly disappear, 1-6 hours to get back)
~5 racks go wonky (40-80 machines see 50% packetloss)
~8 network maintenances (4 might cause ~30-minute random connectivity losses)
~12 router reloads (takes out DNS and external vips for a couple minutes)
~3 router failures (have to immediately pull traffic for an hour)
~dozens of minor 30-second blips for dns
~1000 individual machine failures
~thousands of hard drive failures
slow disks, bad memory, misconfigured machines, flaky machines, etc.

Long distance links: wild dogs, sharks, dead horses, drunken hunters, etc.

• Reliability/availability must come from software!

Slide by Jeff Dean, Google Senior Fellow
What is a distributed system?

“A distributed system is a system in which I can’t do my work because some computer that I’ve never even heard of has failed.”

- Leslie Lamport
Model of a Distributed System

● What are the entities that are communicating in the distributed system?
  ○ An entity is a single process.
  ○ Other options: objects, services, ...

● What communication paradigm do they use?
  ○ Entities communicate by sending messages.
  ○ Other options: shared memory, RPC, publish/subscribe, ...

● How are they mapped onto the physical distributed infrastructure?
  ○ A process runs on a single physical machine.
  ○ Other options: mobile code, mobile agents, ...
Some Model Components

● Interaction characteristics
  ○ Can messages be lost?
  ○ Do they arrive in the order in which they were sent?
  ○ What about message delay?

● Failures
  ○ Can processes crash?
  ○ Can they recover?
  ○ Does data survive the crash?

● Security
  ○ Do all processes follow the specified algorithm?
  ○ If not, what kind of “attacks” are allowed?
Two Important Model Variants

- **Synchronous System:** Known bounds on times for message transmission and processing
  - Can use *timeouts*.

- **Asynchronous System:** No known bounds on times for message transmission and processing
  - More realistic and practical
  - More challenging to develop algorithms for this model?
What is a distributed algorithm?

- An algorithm consists of the steps taken by each process.
  - Sending and receiving messages.
  - Changing local state.

- We will analyze problems and algorithms in the context of models.
  - An algorithm may work under one model but not another.
  - Some problems may be solvable under one model but not another.
Course Topics

● Clocks and the ordering of events in distributed systems
● Distributed logs
● Distributed mutual exclusion
● Global snapshots
● Broadcast algorithms
● Distributed agreement
● Leader election
● Distributed commit protocols
● Consistency models and replication
● Consistent hashing and distributed hash tables