CS 455: Introduction To Distributed Systems [Introduction]

Shrideep Pallickara
Computer Science
Colorado State University

Topics covered in this lecture

- Introduction
- Course overview and expectations
- Communications

Distributed Systems Quick Overview

What is a distributed system?

- *A distributed system is one in which hardware and software components located at networked computers communicate and coordinate their actions only by passing messages.*

  Coulouris, Dollimore, Kindberg and Blair

- *A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.*

  Leslie Lamport

Why Distributed Systems?

- Your hard-drive's primacy has been eroding
- Data and programs are delivered over the network
  - No single hard drive can hold all the data you need
- Services themselves are distributed
  - Google search is backed by a massive distributed cloud

Distributed systems builds on a diverse set of areas

- Networking
- Concurrency
- Algorithms and Graph Theory
- Cryptography
- Failure recovery and consistency models
- Probability theory
- Machine learning
- Information Retrieval
- Transactional Systems

Distributed Systems: CHALLENGES (1/2)

- Scale with increases in data and users
- Responsiveness
  - Regardless of data size, responses must be prompt
- Intelligent
  - Correlate all sorts of information

Distributed Systems: CHALLENGES (2/2)

- Dealing with system conditions
  - Murphy's Law
  - Malicious Users
  - Byzantine failures
- Security
  - Detection
  - Privacy and Accountability
  - Authorizations

ABOUT ME

About me

- I do research in this area
- Areas that I have worked or actively work in include:
  - Cloud computing and analytics
  - Internet-of-Things (IoT)
  - Big Data
  - Content dissemination and streaming systems
  - Grid computing
  - P2P systems
  - Object Request Brokers

My research has been deployed in

- Commercial internet conferencing systems
- Defense applications
- Earthquake sciences
- Epidemic modeling
- Environmental monitoring
- Healthcare informatics & Brain Computer Interfaces
- High energy physics
- Visualizations

COURSE LOGISTICS, EXPECTATIONS, AND SUCH

Course webpage

- All course materials will be on the course webpage
  http://www.cs.colostate.edu/~cs455
  - Schedule
  - Lectures
  - Assignments
  - Syllabus and Announcements
- Grades will be posted on **Canvas**
- There is also a link to the CS455 Pizza forum on the course webpage
  - FAQs and discussions for assignments
- The course website, Piazza, Checkin, and Canvas are all live now

Office Hours

- Instructor: **Shrideep Pallickara**
  Computer Science (CSB 364)
  Office Hours:
  4:00 – 5:00 PM Tuesday and 9:00 – 10:00 AM Friday
- GTA: Office hours will be in CSB 120
  Sitakanta Mishra 12:00-1:00 && 3:00-4:00 pm Mondays and 1:00-3:00 pm Fridays
  Waruna Ranasinghe 8:00-10:00 am Tuesdays and Fridays
- Lab Sessions:
  - CSB 130 from 4:00 – 5:00 pm on Friday
- Please send all e-mails to: **cs455@cs.colostate.edu**

Course textbook

- This class has two **optional** textbooks

  Distributed Systems: Principles and Paradigms. *Andrew S. Tanenbaum and Maarten Van der Steen.* 3rd Edition. Createspace, ISBN 9781530281756

  Distributed Systems: Concepts and Design. George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair. 5th Edition. Addison Wesley. ISBN: 978-0132143011

When I make slides …

- I usually refer to several texts
  - And technical papers and articles (with URLs)
- I always list my references at the end of every slide set

Textbooks that I will refer to during the course include … (1/2)

- Distributed Systems: Principles and Paradigms. *Andrew S. Tanenbaum and Maarten Van der Steen. 2nd Edition. Prentice Hall. ISBN: 0132392275/978-0132392273.*
- Distributed Systems: Concepts and Design. George Coulouris, Jean Dollimore, Tim Kindberg, Gordon Blair. 5th Edition. Addison Wesley. ISBN: 978-0132143011
- Distributed Computing: Principles, Algorithms, and Systems. *Ajay Kshemkalyani and Mukesh Singhal. 1st edition. Cambridge University Press. ISBN: 0521876346/ 978-0521876346*
- Computer Networks: A Systems Approach. *Larry Peterson and Bruce Davie. 4th edition. Morgan Kaufmann. ISBN: 978-0-12-370548-8.*
- *Java Concurrency in Practice.* Brian Goetz, Tim Peierls, Joshua Bloch, Joseph Bowbeer, David Holmes, and Doug Lea. Addison-Wesley Professional. ISBN: 0321349601/978-0321349606.
- *Java Threads.* Scott Oaks and Henry Wong. . 3rd Edition. O'Reilly Press. ISBN: 0-596-00782-5/978-0-596-00782-9

Textbooks that I will refer to during the course include … (2/2)

- *Hadoop: The Definitive Guide.* Tom White. 3rd Edition. Early Access Release. O'Reilly Press. ISBN: 978-1-449-31152-0
- Concurrent Programming in Java(TM): Design Principles and Pattern. *Doug Lea. 2nd Edition. Prentice Hall. ISBN: 0201310090/978-0201310092.*
- Cloud Application Architectures: Building Applications and Infrastructure in the Cloud. George *Reese.1st edition. O'Reilly. ISBN: 0596156367/978-0596156367.*
- Practical Cryptography. *Niels Ferguson and Bruce Schneier. 1st edition. Wiley Publishing. ISBN: 0-471-22894-X/0-471-22357-3.*
- *Unix Systems Programming. Kay Robbins & Steve Robbins, 2nd edition. Prentice Hall. ISBN: 978-0-13-042411-2.*
- *Operating Systems Concepts. Avi Silberschatz, Peter Galvin, Greg Gagne. 8th edition. John Wiley & Sons, Inc. ISBN-13: 978-0-470-12872-5.*

On the Course schedule page

http://www.cs.colostate.edu/~cs455/schedule.html

- You will see the **topics** that will be covered and the **order** in which I will cover them
- The readings section will list the books (and the chapters therein) that form the basis for the materials
- You will also see the complete schedule for when the **assignments** are posted and when they are due

GRADING

Grading breakdown

- Assignments: 55%
  - HW1: 15%; HW2: 20%; HW3: 20%
- Term project and paper: 10%
- Term project presentation: 5%
- Quizzes (10 best) : 10%
- Mid Term: 10%
- Final exam: 10%

Grading Policy (1/4)

- Letter grades will be based on the following standard breakpoints:
  - >= 90 is an A, >= 88 is an A-,
  - >= 86 is a B+, >=80 is a B, >=78 is a B-,
  - >= 76 is a C+, >=70 is a C,
  - >= 60 is a D, and <60 is an F.
- I will not cut higher than this, but I *may* cut lower.

Grading Policy (2/4)

- There is **no extra credit**
  - Any credit you earn, you must do so on a level-playing field with your peers
- There will be **no make-up exams**

Grading Policy (3/4)

- Every assignment will be posted at least 4 weeks before the due date.
- Every assignment will include information about:
  - How much it will count towards the course grade
  - How it will be graded
- Late submission penalty: 7.5% per-day for the first 2 days
  - Submissions after the late submission period will have an automatic ZERO
  - If you submit the wrong files? 30% deduction
  - Detailed submission instructions posted on the course website.
  - Assignments will be graded within 2 weeks of submission

Grading Policy (4/4)

- If you have problems with the grading
  - Talk to the GTA first
- The GTA will strive to ensure that the grading is consistent across the board

Quizzes, mid term, and final

- Each account for **10%** of the course grade
- Final is comprehensive
- There will be 13 quizzes
  - 3 quizzes where you had your lowest scores will be dropped
    - We will take your 10 highest scores
  - If you have an interview or need to miss class for some reason, you do not need to let me know and there will be NO makeup for quizzes (please don't ask to do this!)

Quizzes, mid term, and final

- I will only ask questions about what I teach
  - If I didn't teach it, I won't ask from that portion
- If the concepts were covered in my slides
  - You should be be able to answer the questions

If you are interested in taking this course with the honors option

- Honors courses are expected to be **tougher** courses
- You will be given 1 extra assignment
  - Reliable, ordered communications using UDP (with a simulated packet drop rate of 5%)
    - Demonstrate this with HW1-PC where TCP is replaced with your implementation
  - The best you can do on this assignment is get a **Ø**
- You might have gotten an **A** in the regular course
  - But deductions in the extra assignment may result in you getting a lower grade

CS455 is a capstone course
So there are writing components

ASSIGNMENTS

Assignments: What to expect

- There will be no busy work
  - No GUI
- Complexity will not be through obfuscation
- You will be able to look back and feel good about them
  - Delayed gratification

Assignments have been designed so that they incrementally add …

- Networking
- Threading
- Processing

There will be 3 assignments

- Routing Packets Within a Structured Peer-to-Peer (P2P) Network Overlay
- Threading Assignment
- Analytics using MapReduce (Hadoop)

Each assignment will have TWO components

- Programming element
- Written element

Assignments: The programming component

- This will account for **80%** of the grade for the assignment
- You will have about **4 weeks** to complete each assignment
- The assignments will include **milestones** that should be achieved for each week

Assignments: The written component

- This will account for **20%** of the grade for the assignment
- You will have **48 hours** to complete this
- The questions will be *reflective*
  - Design decisions, possible extensions, optimizations, choice of data structures, etc.
- Will be posted after the programming portion is due

Term project (1/2)

- This will be based on Apache SPARK
  - Team Project
- Term project deliverables
  - Source codes [7 points]
  - Term Project Report [3 points]
  - Term Project Presentation [5 points]

Term project (2/2)

- The term project is a group effort
  - Team size = 2-3 and you can choose your teammate
  - Please **respond** to your teammate's e-mails on time!
    - Make sure he/she has the e-mail that you check regularly
  - If you have problems finding a teammate, please let us know

Assignments: Logistics

- Assignments will be due **at 5:00 pm**
  - **Programming** assignments are due on **Wednesdays**
  - **Written** assignments are due on **Fridays**
- You are allowed to submit up to 2 days late
  - There is a **7.5%** deduction for each day that you are late
- All assignments (except the term project) are **individual** assignments

Expectations

- You will attend all classes
- You will focus on the discussions, and not on …
  - Other assignments
  - Social networking updates
- Assignments have to be done **individually**
- You will be **challenged** in this course
  - Assignments are geared toward real systems

WHAT IT TAKES TO SUCCEED

What it takes to succeed (1/3)

- You are required to work at least **12 hours** per-week outside of class
  - Coding and reviewing material from class
- If you miss a lecture
  - Add about 3 hours per missed lecture

What it takes to succeed (2/3)

- Work on the assignments **every day**
  - There is no such thing as waiting for inspiration to strike!
- **Reflect** about how you could have done things differently for better performance
  - Even after you have submitted an assignment
  - It will improve the choices you make in the next assignment

What it takes to succeed (3/3)

- Work in bigger-sized chunks
  - Too many short bursts = Too many context switches
    - You will be busy doing nothing
- Document your code

How to fail this course?

- Believing that you can learn via osmosis
- **Missing lectures**
  - If you don't have the discipline to show up, you will most likely not have the discipline to catch up
- **Procrastinating**
  - The assignments cannot be done in a week
  - *Organize your schedule* so that you can succeed

Other pitfalls

- Poor management of **course loads**
  - Plan the number and type of courses you take
  - Don't spread yourself so thin that you do not give yourself the opportunity to succeed
- Not attacking the problem and working on the fringes
  - Spend your time wisely on critical paths

Use of laptops, cell phones, tablets, and other electronic devices

- If you must use a laptop or tablet you will have to
  - Turn off wireless
  - And use it only for taking notes
- Authorized laptop/tablet users
  - Pledge forms on table
  - Will sit in the back row starting at the corners
- When the class is in session, put away your cell-phones!

Why attend lectures if all the slides are posted?

- Slides are only part of the story
  - They anchor the discussion
- Any field has a *language* associated with it
- People who have worked in an area for a long time speak the language
  - Sitting in classes helps you learn how to frame questions and responses
- Often there are surprising questions
  - Some of these may be asked by interviewers

Recorded lectures are ONLY for distance students

- The on-campus course is the on-campus course
  - Who wants to lecture in an empty room!
- Students in the on-campus section will not be given the recording links
  - So please don't ask even if you have an interview
- Distance students should not post (or share) the EchoCenter links for the recorded lectures on the Piazza forum or any other forum.

Help me help you

- We will have **surveys** at the end of every class
- You will provide a list of
  - 3 concepts you followed clearly
  - 3 concepts you had problems keeping up with
- Problem areas for the majority of the class will be addressed in the next class

Interactions

- You can have discussions with me, the GTAs, and your peers
- There are **two constraints** to these discussions
  1. No code can be exchanged under *any* circumstances
  2. No one takes over someone else's keyboard
- Bumps are to be expected along the way
  - But you should get over this yourself
  - It will help you with the next problem you encounter

Topics covered in CS455

Topics covered in CS455 [1/2]

- Communications
- Threads: Safety and Concurrency
- Building scalable servers
- MapReduce
- Spark

Topics covered in CS455 [2/2]

- Distributed mutual exclusion
- Election algorithms
- File systems and network storage
- Distributed server topologies
- Distributed storage systems

Communications & Networking

Example: Setting up connections to a server

- Programs open a **socket** to a server that's *listening* for connections
- To create a `Socket` you need to know the Internet host you want to connect to
- Servers don't know *who* will contact them
  - If it did, difficult to synchronize *when* this would happen

An analogy

- Server is like a person sitting by the phone
  - Doesn't know *who* will call and *when*
  - When the phone rings?
    - Talk to whoever is on the other line

Java provides a `ServerSocket` to enable writing servers

- `ServerSocket` runs on the server
  - **Listens** for *incoming* network connections on a particular **port** on the host that it runs on
- When a client socket on a remote host attempts to connect to that server port
  1. Server **wakes** up
  2. **Negotiates** a connection between the client and server
  3. **Opens** a regular `Socket` between the two hosts

Some more about the two types of sockets

- `ServerSockets` **wait** for connections
- Client `Sockets` **initiate** connections
- Once the `ServerSocket` has set up the connection?
  - **Data always travels over the regular `Socket`**

Using the `ServerSocket`

- Created on a particular **port** using the `ServerSocket(port)` constructor
- Listen for communications on that port using `accept()`
  - **Blocks** *until* a client attempts to make connection
  - Returns a `Socket` object that **connects** the client to the server
- Use the `Socket`'s `getInputStream()` and `getOutputStream()` to communicate

Creating the `ServerSocket`

- ```
  ServerSocket serverSocket =
        new ServerSocket(5000);
  ```
  - Tries to create a server socket on port 5000
- ```
  ServerSocket serverSocket =
        new ServerSocket(5000, 100);
  ```
  - Can hold up to 100 incoming connections
- ```
  ServerSocket serverSocket =
    new ServerSocket(5000, 100,
      InetAddress.getHostByName
            ("address2.cs.colostate.edu"));
  ```
  - On a **multi-homed** host, specify the network-address over which connections should be accepted
  ```

Accepting network connections

```
ServerSocket serverSocket =
     new ServerSocket(portNum);
while(true) {
   Socket socket = serverSocket.accept();
   ...
}
```


Closing the client and server sockets

- Closing a `ServerSocket` **frees** a port on the host that it runs on
- Closing a `Socket` **breaks** the connection between the local and remote hosts

We exchange byte streams over the socket

- The `java.io` package contains the `DataInputStream` and `DataOutputStream` that lets you do this elegantly
- `DataInputStream din = new DataInputStream(socket.getInputStream());`
- `DataOutputStream dout = new DataOutputStream(socket.getOutputStream());`

Elements that play a role in communications

- **Transmission media**
  - Wire, cable, fiber, and wireless channels
- **Hardware devices**
  - Routers, switches, bridges, hubs, repeaters, and network interfaces
- **Software components**
  - Protocol stacks, communication handlers, and drivers

Communications & Networking: Topics that we will cover

- Data transmission
- Switched Networks
- Bandwidth and Latency
- Multiplexing
- Internet Architecture
- IP routing
- The TCP and UDP protocols

COMMUNICATIONS & NETWORKING {HOW DATA IS SENT}

How is the data sent?

- Are we sending **1**'s and **0**'s?
- Whatever the physical medium, we use **signals**
  - Electromagnetic waves traveling at the speed of light
    - Speed of light is different in different mediums

Components of encoding binary data in a signal

- Modulation
- Duplexity

Encoding binary data: Modulation

- Objective is to send a ***pair*** of **distinguishable** signals
- Vary frequency, amplitude, or phase of the signal to transmit information
  - E.g. vary the power (amplitude) of signal
  - $x(t) = A\sin(2\pi ft + \theta)$

Encoding binary data: Duplexity

- *How many* bit streams can be encoded on a link at a time?
  - If it is one: nodes must share access to link
- Can data flow in both **directions** at the same time?
  - Yes ➔ full-duplex
  - No ➔ half-duplex

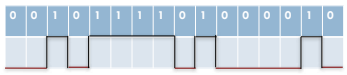
For our purposes, let's ignore details of modulation

- Assume we are working with two signals
  - High and low
- In practice:
  - Different voltages on a copper-based link
  - Different power-levels on an optical link

Let's do the obvious thing

- Map **1** to a high signal
- Map **0** to a low signal

Non-return to zero (NRZ)



Problems with NRZ because of consecutive 1's and 0's: **Baseline wander**

- Receiver keeps **average** of the signal seen so far
- Average is used to *distinguish* between low and high
- Lots of consecutive **1**/**0**'s will make it difficult to detect a significant change

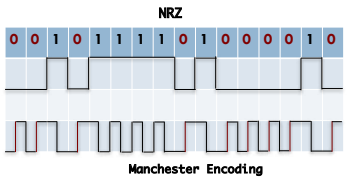
Problems with NRZ because of consecutive 1's and 0's: **Clock Recovery**

- Every clock cycle, sender transmits and the receiver receives
- Sender and receiver's clocks must be perfectly **synchronized**
  - Otherwise, it is not possible to decode the signal

Manchester encoding

- **0** is a low-to-high transition
- **1** is a high-to-low transition

Manchester encoding and NRZ



Some more about Manchester encoding

- Doubles the rate at which signal **transitions** are made on the link
  - Receiver has ½ the time to *detect* each pulse
- Rate of signal changes: baud rate
- Bit rate is ½ the baud rate
  - Encoding is considered 50% efficient

NRZI (Non return to zero inverted)

- Make a transition from current signal to encode a 1
  - **Stay** at current signal to encode a 0
- Solves the problem of consecutive 1's
  - But does nothing for consecutive 0's

4B/5B encoding

- Attempts to address inefficiencies in Manchester encoding
  - Without suffering from problems due to extended high/low signals
- The crux here is to insert **extra** bits into bitstream
  - Breakup long sequences of 1s or 0s
  - 4 bits of actual data encoded in a 5-bit code
  - 5-bit codes are carefully selected
    - No more than 1 leading 0 & no more than 2 trailing 0s

4B/5B encoding

4B	5B
0000	11110
0001	01001
0010	10100
0011	10101
0100	01010
0110	01110
0111	01111
1000	10010
1001	10011
1010	10110
1011	10111
1100	11010
1101	11011
1110	11100
1111	11101

4B/5B: Rules for the conversion of 4-bit codes to 5-bit codes

- Objective is to ensure that in each translation there is:
  - No more than one leading 0
  - No more than two trailing 0's
  - When sent back-to-back
    - No pair of 5-bit codes results in more than 3 consecutive 0's being transmitted
- 5-bit codes are transmitted using NRZI
  - This is why they are so concerned with consecutive 0's

The contents of this slide-set are based on the following references

- *Computer Networks: A Systems Approach. Larry Peterson and Bruce Davie. 4th edition. Morgan Kaufmann. ISBN: 978-0-12-370548-8. [Chapter 1, 2]*
- *Java Network Programming, Third Edition. Elliotte Rusty Harold. O'Reilly. ISBN-10: 0596007213 / 978-0596007218.* [Chapter 7]