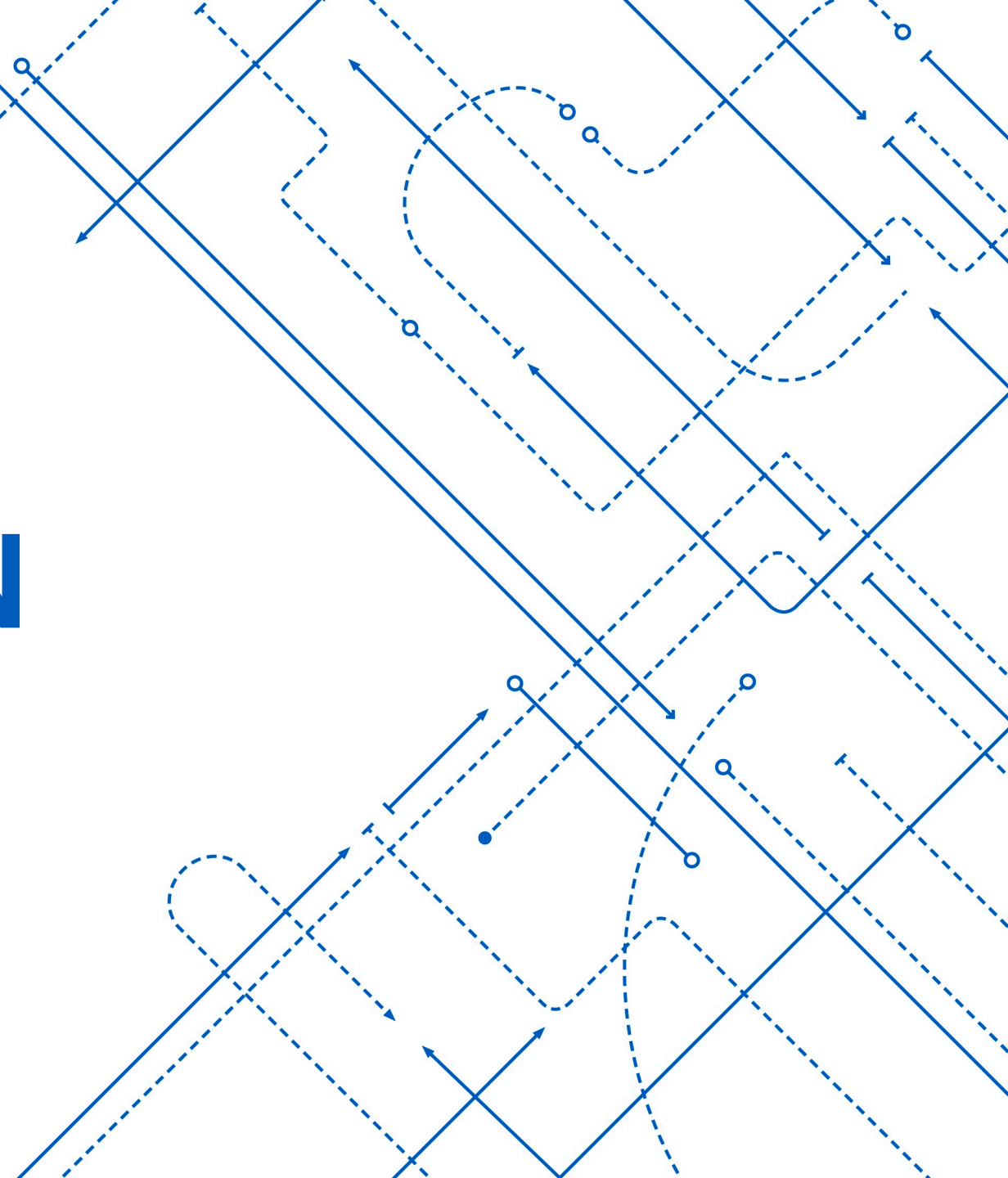
University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences



COURSE INTRODUCTION

CSE 486(586): Distributed Systems

Lecture 1

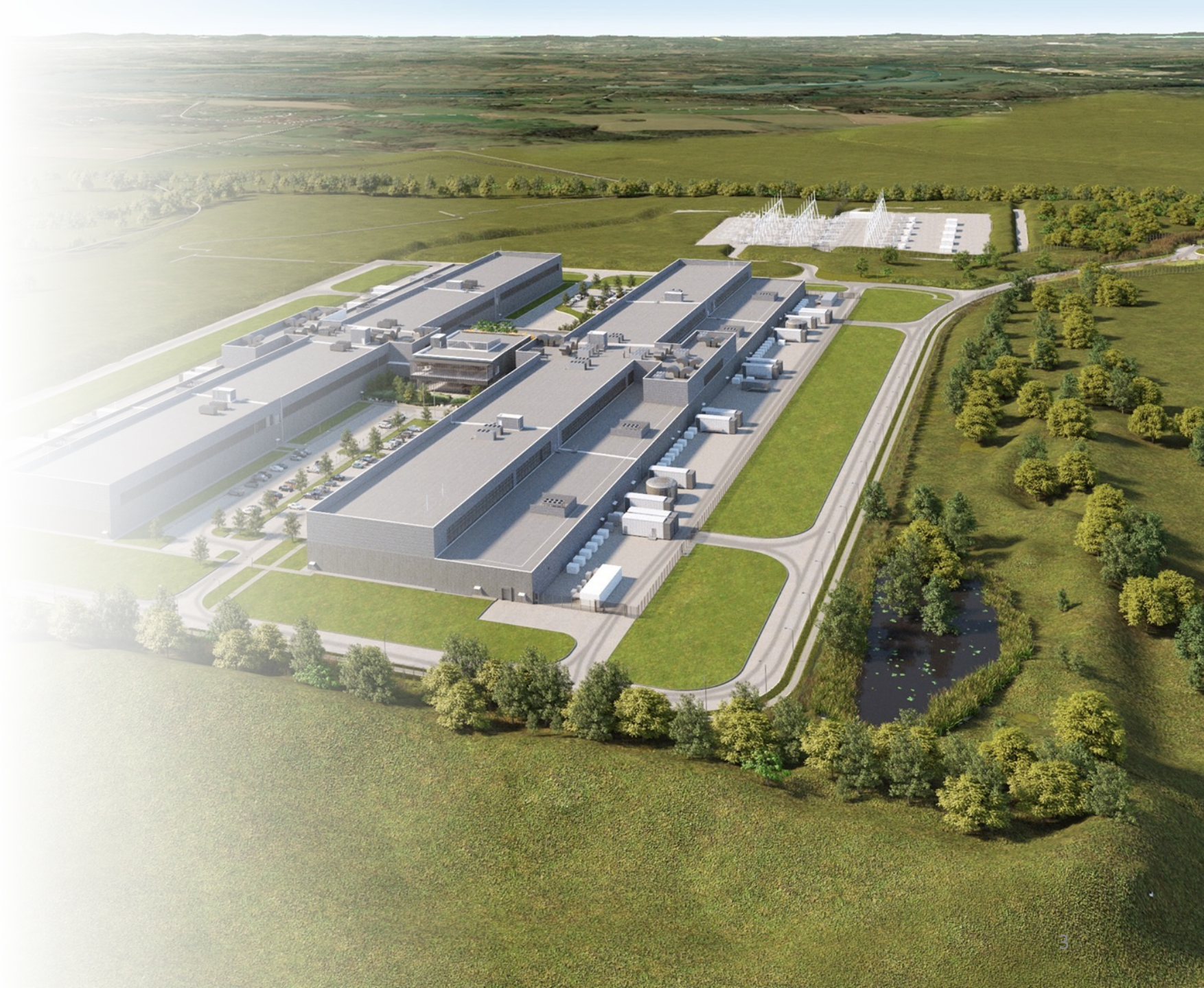
Haonan Lu

What Are Distributed Systems?

- Many physical machines
- Connected by a network
- Work together to provide services

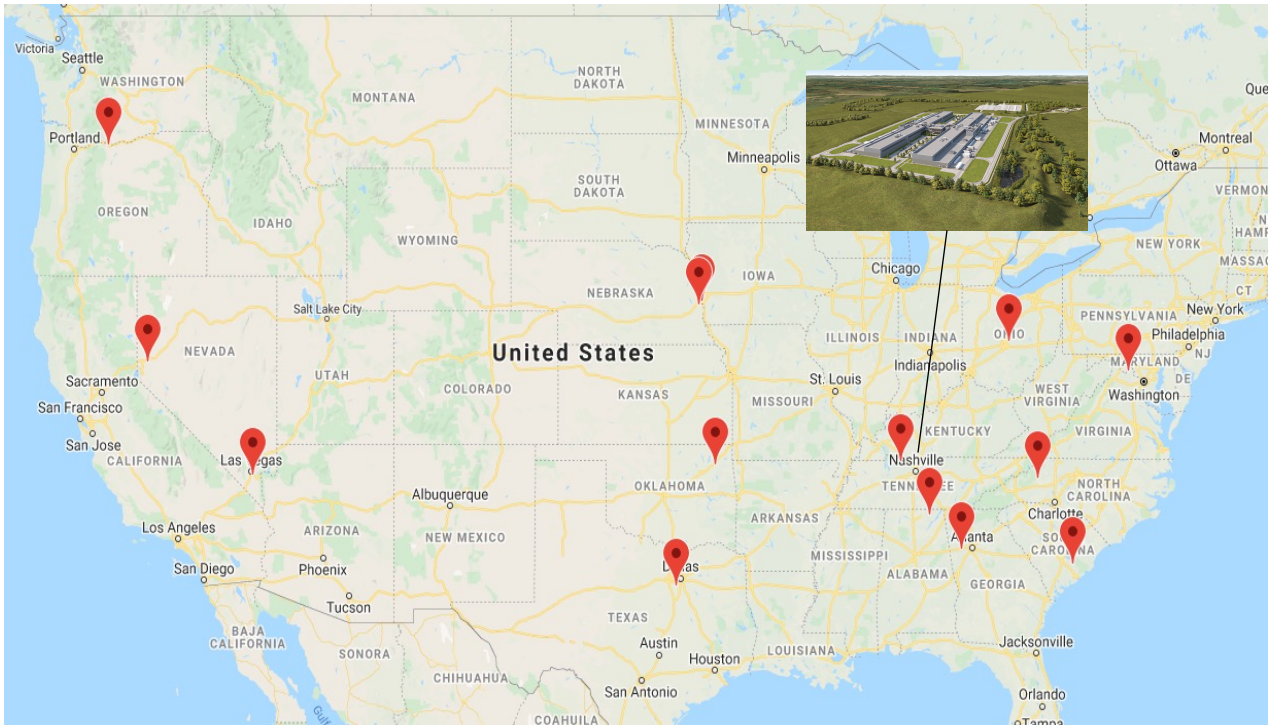
A row of racks of servers inside a typical datacenter

A typical datacenter



Geographically

dispersed datacenters



Why Distributed and System?

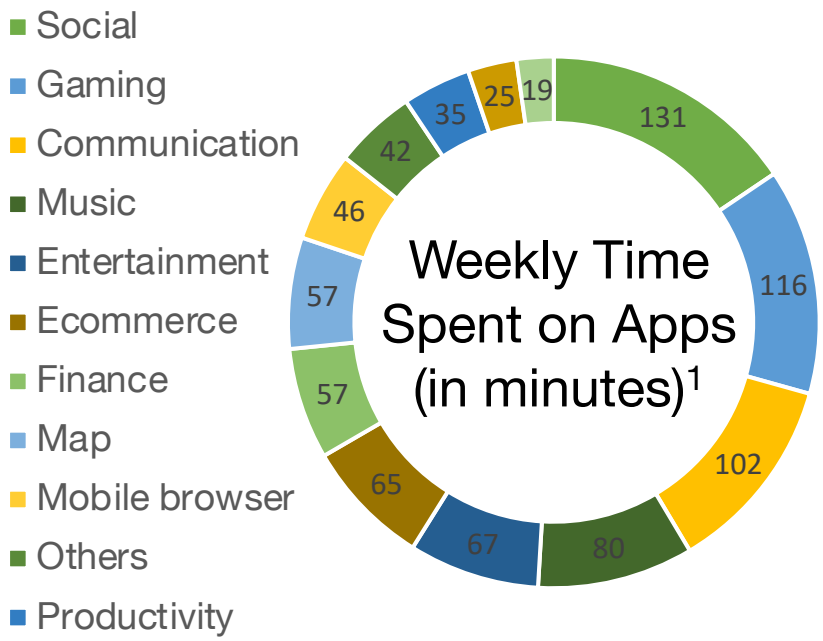
- Distributed vs. a single machine
  - [Storage] stores a huge amount of user data
  - [Throughput] processes a large number of user requests
  - [Latency] brings services closer to users
  - [Availability] tolerates failures, enables “always-on” services

Why Distributed and System?

- A system serves developers and end users
  - ➢ Provides a high-level interface by abstracting the physical machines underneath, simplifying application development
  - ➢ Defines/controls how user requests are handled, affecting user experience

Why Study Distributed Systems?

- Fundamental building block of today's applications
  - ➢ Facebook [TAO, ATC '13]
  - ➢ Google [Spanner, OSDI '12]
  - ➢ Amazon [Dynamo, SOSP '07]



[1] Simform. App Usage Statistics 2021 that'll Surprise You. https://www.simform.com/blog/the-state-of-mobile-app-usage/.

Why Study Distributed Systems?

- Key enabler of future applications
  - ➢ Data keeps growing
  - ➢ Increasing demand on performance
  - ➢ Applications are increasingly complicated

Course Overview

Learning Objectives

- Understand fundamental concepts/techniques
  - RPCs, notion of time, etc.
- Reason about challenges and solutions
  - Concurrency, scalability, failures, performance, etc.
- Build systems like a system designer/developer
  - Correctly handle concurrency and failures
  - Quickly pick up a “new” language
  - Develop and submit your code using version control

Lectures

- 2:00 – 3:20 PM, T/R, Knox 109
- Slides posted before class meetings
- No textbook
- Research papers for optional reading
- Piazza for discussion and announcements
- Prerequisites: CSE 305/505 (programming maturity)

Course Staff

- Instructor: Haonan Lu
  - haonanlu@buffalo.edu, 338M Davis Hall
- TAs
  - Elvis David Rodrigues (elvisdav@buffalo.edu)
  - Adithya Raman (araman5@buffalo.edu)
  - Ramesh Pavan Pothamsetty (rpothams@buffalo.edu)

Office Hours

- Haonan: 3:30 PM--5:00 PM Monday and Thursday
- Elvis: 10 AM--12 PM Tuesday and Thursday
- Adithya: 2 PM--4 PM Wednesday and Friday
- Ramesh: 8 AM--10 AM Monday and Wednesday

Grade Components

- Exams (60%)
  - Midterm: 30%
  - Final: 30%
- Programming assignments (40%)

Midterm and Final Exams

- In-class, closed-book
- Questions are from lectures and assignments only
- Midterm: 2:00 PM--3:20 PM, Thursday, October 6, 2022, Knox 109
- Final: 3:30 PM--6:30 PM, Tuesday, December 13, 2022, Knox 109

Course Project

- Four programming assignments, all in Go
  1. “MapReduce” (Go warmup)
  2. Distributed snapshots
  3. Raft leader election
  4. Raft log consensus

Course Project

- All are individual assignments
- Submission via Git, graded on Autolab
- You are provided with tests
  - They are exactly the tests we use to grade your assignments
- Late policy
  - 1 free late day for a1-1 --- a1-3, 2 free late days for a2, 3 free late days for a3 and a4
  - Granularity: a day, e.g., 1-second late == 1-day late
  - 10% off for every 1-day late

Warnings

- Assignments are difficult and VERY time-consuming!
  - ➢ Do NOT be fooled by Assignment 1
    - ❖ Made easy on purpose, helps learn Go
  - ➢ Later ones are “exponentially” more difficult
  - ➢ Debugging can take 10X more time than initial coding

Warnings

- Some self-reported hours spent on assignments

Assignments	Median	Min–Max
1-1	2	1 – 6
1-2	3	1 – 8
1-3	4	1 – 10
2	6	2 – 15
3	12	5 – 29
4	30	10 – 100+

Warnings

- These assignments are not just about programming
  - About understanding and designing protocols
  - A careless design makes it significantly harder to code right
  - Assignment 4 builds on Assignment 3's solution

Suggestions

- Start early early early (weeks early)
- Understand protocols and work out design first
- Work out solutions and write code progressively
- Save progress frequently, e.g., git commit and push!
- Try test-driven debugging

You Got a Deal!

- Course project (40%)
  - Assignment-1: 10%
  - Assignment-2: 10%
  - Assignment-3: 20%
  - Assignment-4: 10% extra credits

Write Your Own Code

- Do NOT discuss code details
- Do NOT copy another person's programs, comments, or any part of the assignments
- Do NOT write code for others
- Do NOT make your code publicly available
- We check code duplication

Before Next Class

- Read course description carefully
  - Available on UB Learns (course information tag)
- Join Piazza
  - Join link: https://piazza.com/buffalo/fall2022/cse486586
- Course calendar
  - http://shorturl.at/ewHPX

Before Next Class

- Get your copy of programming assignments
  - Accept https://classroom.github.com/a/xDwEA7z-, and read README of the repo
  - Play around with assignment setup, read Go-and-Assignments-Step on UB Learns
  - Fill out this google form: https://forms.gle/T8Gzz6YQroZ1P2Xs5
  - Try login to Autolab: https://autograder.cse.buffalo.edu/
- Read suggested paper
  - How to Read a Paper
- Questions? Post on Piazza under Lec-1 tag