UBC CS 538B
Distributed Systems Abstractions

with Ivan Beschastnikh

COVID Year 2020, Fall term

Agenda for our first class

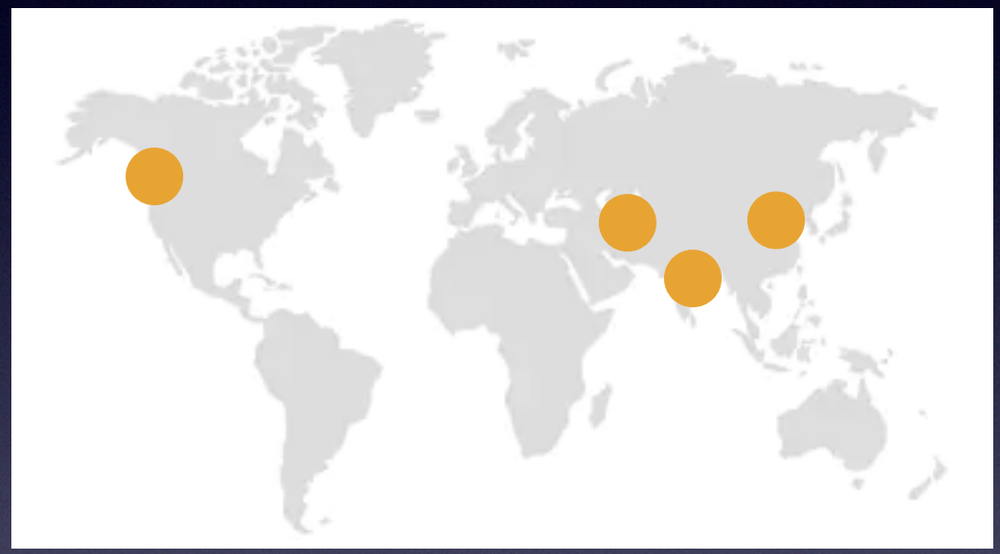
- Who am I, and who are you?
- Online format
- Survey results
- Course overview; reading research papers
- *Break*
- Distributed systems discussion

Who am I?

- Call me Ivan
- I'm in Vancouver (actually live on UBC campus)
- Have been at UBC for 7 years
- Before that worked on my PhD at U.Washington
- Broad research interests, usually intersecting with distributed systems in some way
- I teach UBC grad + ugrad distributed system courses

Who are you?

- We are distributed across *4* countries



Let's go around!

1. What is your **name**? CS MSc/PhD/other **program**?
2. **Where** are you connecting from?
3. What is one **interesting fact about yourself**?

New this year: online format

- The atmosphere in the classroom is key
- Generally, there won't be slides

Discussion-focused course

- Note: my first time teaching online
- Looking forward to figuring this out together with you, would like to hear your suggestions!

Format guidelines

- Respect for each other
- Active participation
    - My camera will always be on
        - Your camera use is optional, but recommended. Try to at least turn camera on at start and end of class. Consider putting up a photo if you switch off camera.
        - Ask questions in chat, or interrupt by voice/raise hand
- Let’s make an effort to be clear: speak slowly and clearly

Other suggestions/ideas?

Other survey results

- Reached consensus quickly

➡ No final

➡ Open-ended project

➡ Recorded zoom calls (only available to class)

Show survey results

Course resources overview

- **Canvas**: place to find zoom links
- **Zoom**: this magic place we are all in
- **Piazza**
  - Use this for all course-related communication
- **Slack** (*optional*)
  - Place to hang out, good for project work

Structure of the course

- Schedule with papers
- Advocate/skeptic roles
- Project details
- Marking details
- How to do well
- Honesty
- *Taking care of yourself*

Review homepage

Reading research papers

(Today's readings)

- The three pass approach by Keshav

1. Bird eye's view (note order of sections)

2. Careful read

3. Virtually re-implement the paper

Reviewing papers
(Today's readings)

- Roscoe's suggestions
- *"A review is a chance to get your own thoughts on the paper straight by writing them down. It's surprising how your opinion of a paper can change by being forced to explain it."*

Responses advice on homepage
https://www.cs.ubc.ca/~bestchai/teaching/cs538b_2020w1/responses.html

Volunteer advocate/skeptics for next week

- **Tuesday**: *Fundamentals of Distributed Computing: A Practical Tour of Vector Clock Systems*
  - *Advocate: Shiqi*
  - *Skeptic: Mayank*
- **Thursday**: *Distributed Snapshots: Determining the Global States of a Distributed System.*
  - *Advocate: Lucca*
  - *Skeptic: Fangyu*

Distributed Systems

- What are they?
  - A system where components may **fail**
  - A ***network*** with inter-dependent computers
  - Is RAID a distributed system? (no network)
  - Anything that requires **more than one machine**?
    - Concurrency (versus parallelism)
    - Concurrency = executing **at same time (independent processes)**
    - Parallelism = **group work** of multiple tasks
  - Multiple cores in a CPU? (2 say yes!)

Distributed Systems

- Why build them / what are their advantages?
  - **Security**: distribute information across parties
  - **Fault tolerance**: one failure doesn't bring down the system (redundant copies of data)
  - **Scalability**: support more users/workload
- Why **not** build them / what are the disadvantages?
  - SE challenge: splitting logic into components
  - Complexity: SE and design challenge
  - Security: data in more places; risk goes up because there are more places where you can be vulnerable

D. Systems *Abstractions*

- *What do you think would be some distributed systems abstractions that might be useful?*

1. Name a (potential) abstraction
2. How is it useful?

- **Reliable communication**: ensure message delivery
- **Sharding**: split data into chunks and assemble chunks from different machines (data is a collection of chunks)
- **Map reduce:** distribute compute abstraction

Next class

- **Distributed time** reading
- Slightly theoretical
- Fundamental abstraction!

Post-class short survey

- I'll post on piazza a short survey about what worked well and what you wish could be done better.

- Please answer it to give me some feedback :-)