Welcome to CS425/ECE428!

Distributed Systems

Instructor: Radhika Mittal

Today's agenda

- Course overview
- Logistics
- Distributed System Model (if time)
    - Chapter 2.4 (except 2.4.3), parts of Chapter 2.3 from your textbook.

Today's agenda

- Course overview
- Logistics
- Distributed System Model (if time)
  - Chapter 2.4 (except 2.4.3), parts of Chapter 2.3

What is a distributed system?

Hardware or software **components** located at **networked** computers that communicate or **coordinate** their actions only by **passing messages**.

- *Your textbook*

(Coulouris, Dollimore, Kindberg, Blair)

What is a distributed system?

A collection of **autonomous computing elements**, connected by a **network**, which appear to its users as a **single coherent system**.

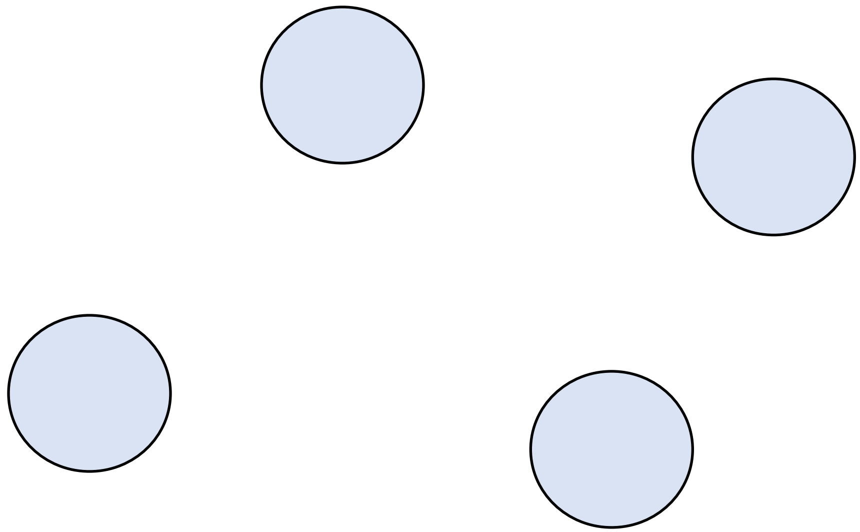
- *Steen and Tanenbaum*

What is a distributed system?

A system in which **components** located on **networked** computers communicate and **coordinate** their actions by **passing messages**. The components interact with each other in order to achieve a **common goal**.

- *Wikipedia*

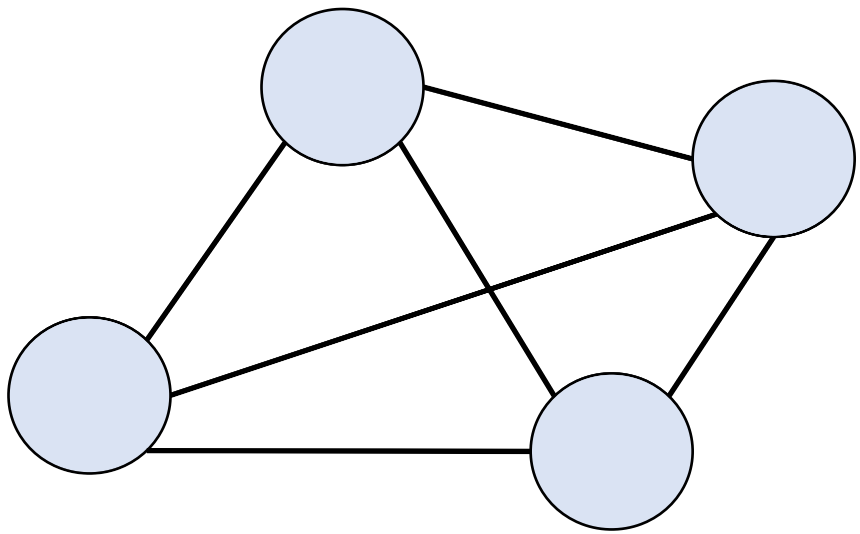
What is a distributed system?



Independent components or elements

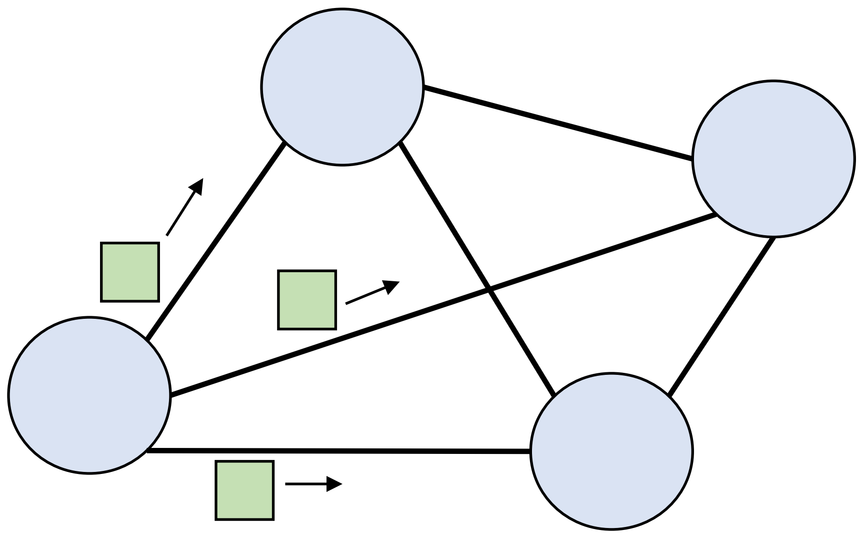
(software processes or any piece of hardware used to run a process, store data, etc)

What is a distributed system?



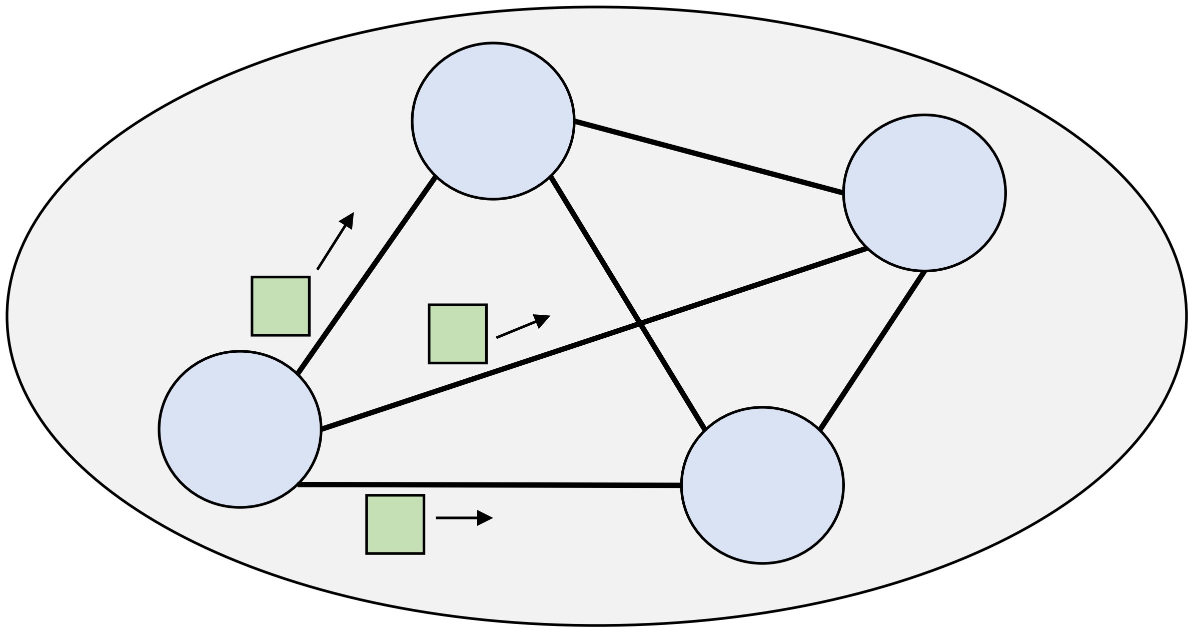
Independent components or elements that are **connected by a network**.

What is a distributed system?



Independent components or elements that are **connected by a network** and communicate by **passing messages**.

What is a distributed system?



Independent components or elements that are **connected by a network** and communicate by **passing messages** to achieve a **common goal**, appearing as a single coherent system.

What is a distributed system?

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*

Examples of distributed systems

- World Wide Web
- A cluster of nodes on the cloud (AWS, Azure, GCP)
- Multi-player games
- BitTorrent
- Online banking
- Bitcoin
- ….

Why distributed systems?

- Nature of the application
  - *Multiplayer games, P2P file sharing, client requesting a service.*
- Availability despite unreliable components
  - *A service shouldn't fail when one computer does.*
- Conquer geographic separation
  - *A web request in India is faster served by a server in India than by a server in US.*
- Scale up capacity
  - *More CPU cycles, more memory, more storage, etc.*
- Customize computers for specific tasks
  - *E.g. for storage, email, backup.*

Example: scaling Facebook (Meta)

- 2004: Facebook started on a single server
  - Web server front end to assemble each user's page.
  - Database to store posts, friend lists, etc.
- 2008: 100M users
- 2010: 500M users
- 2012: 1B users
- 2019: 2.5B users
- 2023: 3B users

How do we scale?

Example: scaling up Facebook (Meta)

- One server running both webserver and DB

- Two servers: one for webserver, and one for DB
  - *System is offline 2x as often!*

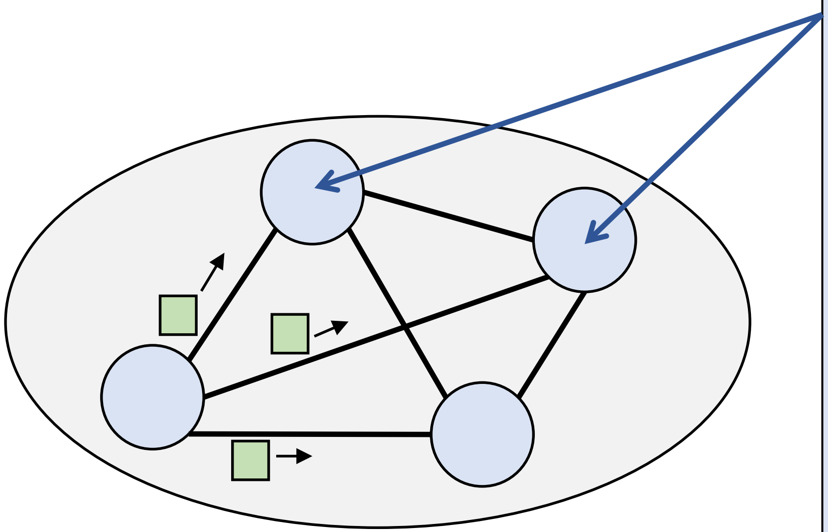
- Server pair for each social community
  - *E.g., school or college*
  - *What if server fails?*
  - *What if friends cross servers?*

Example: scaling up Facebook (Meta)

- Scalable number of front-end web servers.
  - Stateless: if crash can reconnect user to another server.
  - Use various policies to map users to front-ends.

- Scalable number of back-end database servers.
  - Run carefully designed distributed systems code.
  - If crash, system remains available.

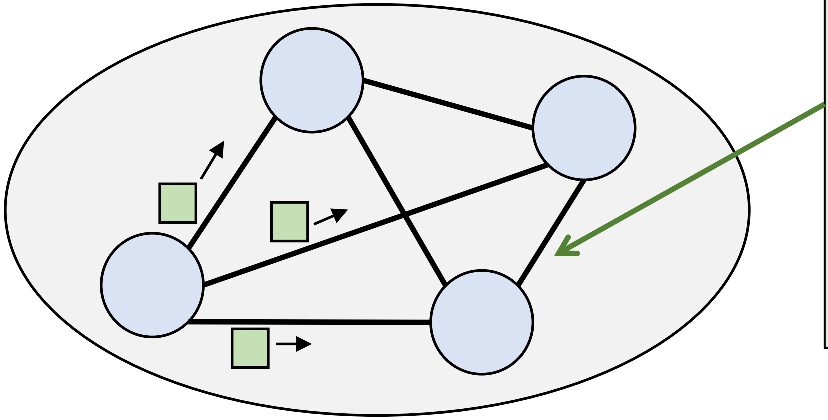
Challenging properties



Multiple computers

- Concurrent execution.
- Independent failure.
- Autonomous administration.
- Heterogeneous.
- Large numbers.

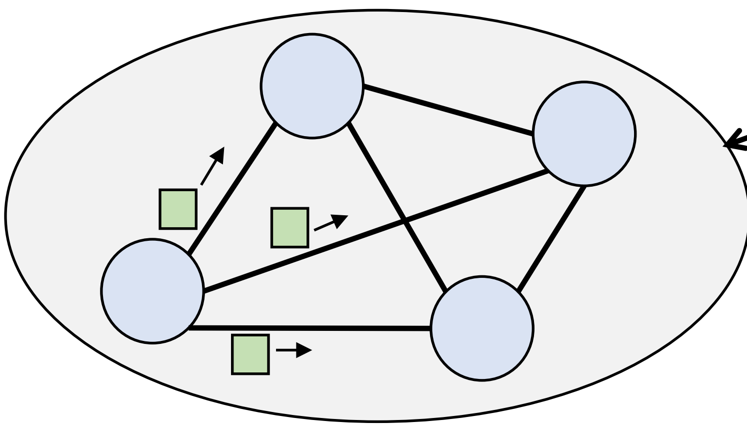
Challenging properties



Networked communication

- Asynchronous
- Unreliable
- Insecure

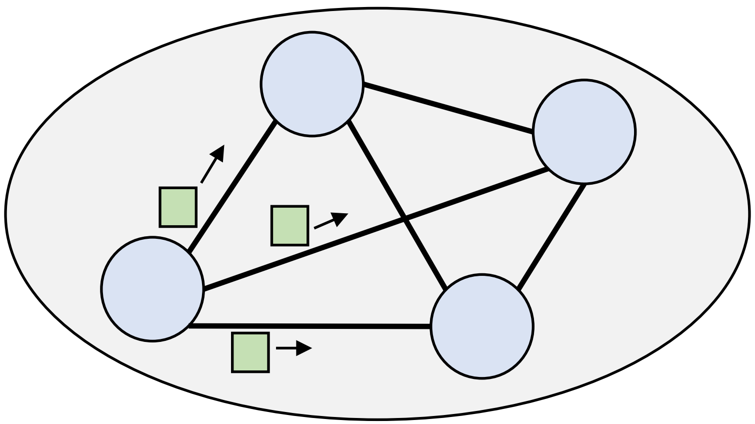
Challenging properties



Common goal

- Consistency
- Transparency

Challenging properties



Multiple computers

- Concurrent execution.
- Independent failure.
- Autonomous administration.
- Heterogeneous.
- Large numbers.

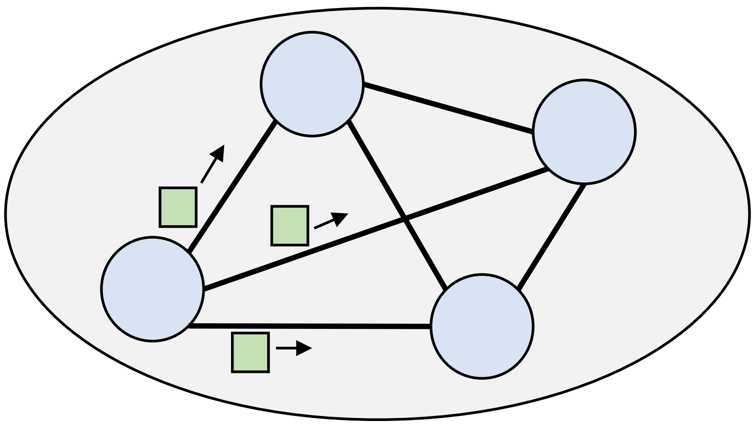
Common goal

- Consistency
- Transparency

Networked communication

- Asynchronous
- Unreliable
- Insecure

Challenging properties



Multiple computers

- Concurrent execution.
- Independent failure.
- Autonomous administration.
- Heterogeneous.
- Large numbers.

Networked communication

- Asynchronous
- Unreliable
- Insecure

Common goal

- Consistency
- Transparency

What you will learn in this course

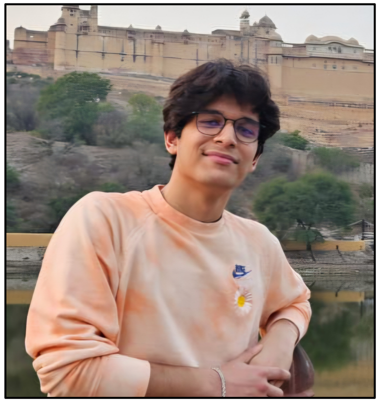
- **Distributed system concepts and algorithms**
  - How can failures be detected?
  - How do we reason about timing and event ordering?
  - How do concurrent processes share a common resource?
  - How do they elect a "leader" process to do a special task?
  - How do they agree on a value? Can we always get them to agree?
  - How to handle distributed concurrent transactions?
  - ….
- **Real-world case studies**
  - Distributed key-value stores
  - Distributed file servers
  - Blockchains
  - …

Today's agenda

- Course overview
- Logistics
- Distributed System Model (if time)
  - Chapter 2.4 (except 2.4.3), parts of Chapter 2.3

Course Staff

Radhika Mittal
Asst. Prof.
ECE and CS



Naman Raina
(MS, ECE)

Lyuwei Su
(MS, ECE)

Talha Waheed
(PhD, CS)

Yu Li
(MS, ECE)

Sources of information

- **Course website**
  - https://courses.grainger.illinois.edu/ece428/sp2026/
    - https://courses.grainger.illinois.edu/cs425/sp2026/ also works.
  - Time slots and locations for office hours (to be updated)
  - Homeworks, MPs
  - Lecture schedule, readings, and slides

- **Campuswire**
  - Announcements, questions, clarifications

Books

- *Distributed Systems: Concepts and Design*, Coulouris et al., 5th edition.
  - Earlier editions may be acceptable.
  - Your responsibility to find correct reading sections.

- Other texts
  - *Distributed Systems: An Algorithmic Approach*, Ghosh
  - *Distributed Systems: Principles and Paradigms,* Tanenbaum & Steen
  - *Distributed Algorithms*, Lynch

Class Timings and my OHs

- Wednesdays and Fridays, 12:30-1:45pm.
- My office hours: right outside class, right after class on WF (1:45-2:15pm).

Lecture Videos

- Lecture videos will be uploaded to MediaSpace.
- Plan on attending classes for a better learning experience.
  - Use lecture videos only to fill in gaps in understanding.
- Students with conflicts during class timings:
  - Please make sure you view the lectures timely and regularly.
  - Ask clarifying questions on Campuswire or during office hours.

Relevant Online Platforms

- Campuswire
  - Link with access code has been shared over email.
  - Reach out to Yu Li (yuli9@illinois.edu) if you need access to CampusWire.

- Gradescope
  - We will add students soon……stay tuned.

- PrairieLearn and CBTF for exams
  - More instructions to follow.

Those waiting to register

- A few more seats might open this week.
- Please start attending classes.
- More sure you have Campuswire access.

Grade components

- **Homeworks**
    - 5 homeworks in total.
    - Approx every 2-3 weeks.
    - Will be submitted using Gradescope.
    - Must be **typed** (hand-written diagrams are fine).
    - Must be done **individually**.

Grade components

- **Homeworks**
- **MPs (only for 4 credit version)**
    - 4 mini projects.
    - First (warm-up) MP0 will be released next Wednesday!
    - Groups of up to 2
        - Need to fill up a form to activate VM clusters (will be shared on Friday)
    - MP0, MP1, and MP3 can be in any language
        - Supported languages: Python, Go, C/C++,
        - You can also use other languages (e.g. Java, Rust), but might get limited help from course staff.
    - *MP2 must be implemented in Go.*

Late Policy

- For homeworks:
  - Can use a total of 48 late hours across the entire semester.

- For MPs
  - Can use a total of 168 late hours (1 week) across the entire semester.
  - Counted individually for each student, so keep your late hours in mind if you end up changing groups over the course of the semester.

Grade components

- Homeworks
- MPs (only for 4 credit version)
- Exams via CBTF
  - Two midterm
    - Midterm 1: Mar 4 – Mar 6
    - Midterm 2: Apr 12 – Apr 14
    - More details to follow.
  - Comprehensive final: May 7 – May 15

Grade components

- Homeworks
- MPs (only for 4 credit version)
- Exams
- CampusWire / Class participation

Grade distribution

	3-credit	4-credit
Homework	33%	16% (drop 2 worst HWs)
Midterms	33%	25%
Final	33%	25%
MPs	N/A	33%
Participation	1%	1%

Switching between credits

- If you'd like to switch between 3 and 4 credits, you should be able to do so using self-service.
- If you are unable to make the switch, reach out to CS advising office for help.

Grading

- Homeworks will not be curved.
  - For 3-credit students:
    - (sum of all 5 homework scores) * 100 * 0.33 / 200
  - For 4-credit students:
    - (sum of best 3 homework scores) * 100 * 0.16 / 120

- MPs will not be curved.
  - (sum of all four MP scores) * 100 * 0.33 / 330
  - MPs have different weightage: MP0:30, MP1: 110, MP2: 110, MP3: 80

- Participation score: directly taken from Campuswire
  - if reported score > 100, you get full 1%
    - Else you get (reported score /100)%
  - Upto 100% bonus for active participation in class.

Grading

- Exam curving formula:
  - Average score across undergraduate students = 80%
  - relative: 80 + 10*(your score – avg_UG_score) / standard_dev
  - We will use max(absolute, relative) to get final score out of 100.

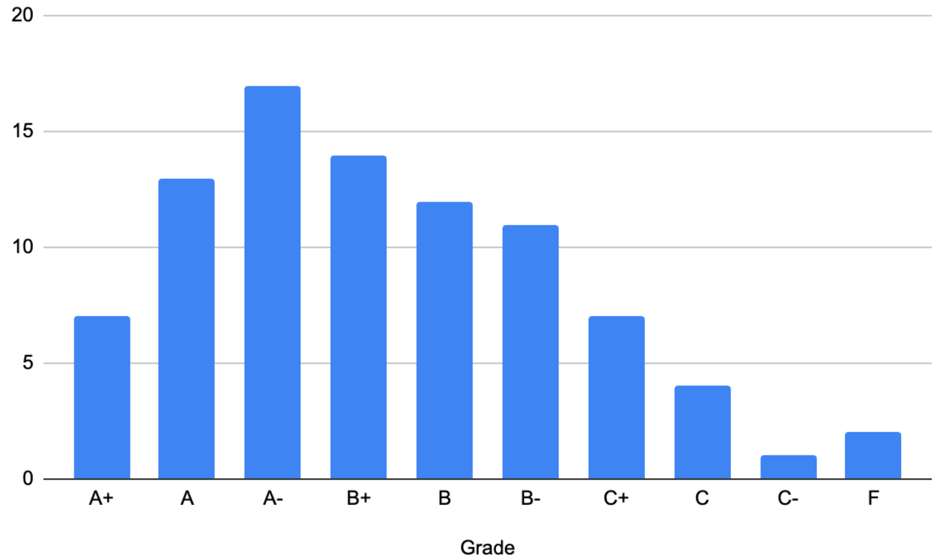
- Multiply the resulting score (out of 100) for each midterm by:
  - 0.165 for 3-credit students
  - 0.125 for 4-credit students

- Multiply the resulting score (out of 100) for the final by:
  - 0.33 for 3 credit students
  - 0.25 for 4-credit students

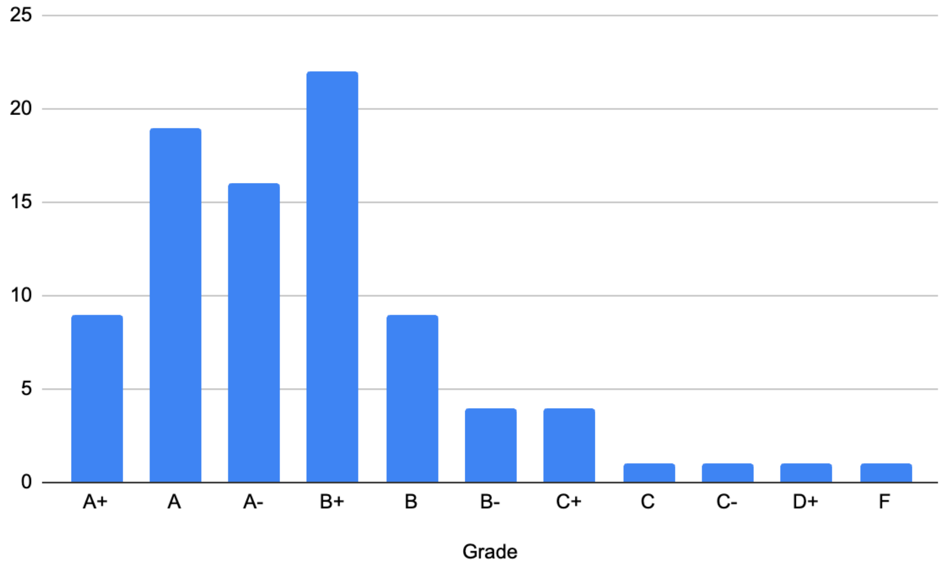
Tentative Grades Cutoff

- Tentative mapping from score to grade (*rough estimate*):
  - Cutoff for B: 80%
  - Bump up a grade for each 4% leap above 80%.
    - B+ 84%, A- 88%, A 92%, A+ 96%
  - Bump down a grade for each 4% leap below 80%
    - B- 76%, C+ 72%, …..
- This is subject to change!
- Technically possible for all of you can get an A (if you all score above ~92%), but……

Expectations



3 credit grade distribution from 2025



4 credit grade distribution from 2025

Expectations

- My teaching goal:
    - ~~high score on your exams for this course~~
    - learn concepts that will be useful to you for years to come!

Expectations

- Doing well in assignments and exams will require thorough understanding of concepts taught in class.

- No practice exams (set as template of actual exams) will be provided!

- Abridged version of class slides will be made available during the exams.
  - Given limited time and the nature of the questions, you cannot rely on simply scrolling through it to answer questions.

- You need to study hard!

Expectations for 4 credits

- MPs are difficult and require significant amount of work.
- MP0, that will be released next week, is only a light warm-up to familiarize you with the cluster environment.
- Subsequent MPs will be much much harder, requiring you to build complete systems using concepts taught in class.
- Dealing with the risks associated with your partner not doing their fair share of work or dropping/switching midway through the course is your responsibility.

Integrity

- Academic integrity violations have serious consequences.
  - Min: 0% on assignment
  - Max: expulsion
  - All cases are reported via the FAIR system.
- **As students, it is your responsibility to uphold academic integrity.**
- Example of violations:
  - Sharing of code outside group.
  - Copying homework and MP solutions (from colleagues, from previous years', from the web).
  - Using AI assistance for auto-completing MPs.
  - Collaborating in exams.
  - ……

Questions?