CPSC 416 Distributed Systems

Winter 2023 Term 1 (October 10, 2023)

Tony Mason (fsgeek@cs.ubc.ca), Lecturer

Logistics

Teaching Assistants

Andy Hsu (andy.hsu@alumni.ubc.ca)

Hamid Ramezanikebrya (hamid@ece.ubc.ca)

Jonas Tai (jonastai@student.ubc.ca)

Cathy Yang (kaiqiany@student.ubc.ca)

Office Hours

Remember: Use Piazza for **all** official course-related communications

- Not on Piazza? Not official.
- Canvas “comments/messages” **are not monitored**

Office Hours:

Who	When	Where
Tony	Monday 14:00-15:00 Wednesday 16:00-17:00	Discord
Andy	Thursday 19:00-20:30	Discord
Hamid	Friday 16:30-18:00	Kaiser 4075
Jonas	Thursday 13:00-14:00	X241
Cathy	Friday 09:00-10:30	X237

Upcoming

This week

- **No class Thursday (October 12, 2023)** – Monday classes instead!
- Design Project 2 Due Thursday October 12, 2023 @ 17:00

Next week

- Tuesday October 17:
  - Self-Assessment
  - Design Project 1: Peer Feedback (Implementation Report)
  - Design Project 3 Team Declaration
- Thursday October 19:
  - Self-Assessment
  - Design Project 2: Peer Feedback

Note:

- You are strongly encouraged to collaborate with others on this
- You should use tools at your disposal to answer these questions
- **Do not forget to submit it.**

Today’s Failure

Gitlab.com Failure Example

Date: January 31, 2023 17:20 UTC

Event: Copied database from production to staging

19:00 UTC

Event: Increased Database load

Suspected cause: spam

Actual Issue(s):

- Users cannot post comments on issues/merge requests
- Background delete of Gitlab employee + data (due to accidental abuse flag)

Gitlab Failure

23:00 UTC

Event: Secondary replication process “falls behind”

Problem: **Primary** has already garbage collected log segments needed by secondary

Solution: Manually resynchronize primary and backup

- **Delete** the secondary backup
- **Copy** the primary to the secondary

Things get worse:

- Copy routine fails to start
- Copy routine blocks waiting for data from primary to secondary: no feedback
- Try to clean up the database directory on secondary
  - Oh no! Accidentally deleted it on the **primary**.

Gitlab Failure

Things got worse:

- They couldn’t find their backups in S3
- Turns out their backup process was using an **old** version of the backup tool.
  - It won’t backup a newer version of the database
  - Nobody noticed
  - Automatic cleaning of old backups had deleted **everything**.
- Azure disk snapshots **were not enabled for the database volumes**

Resolution:

- Restored from LVM snapshot
- Time to restore: around 18 hours

Gitlab Failure

Takeaways:

- Restore time is the **worst** time to figure out your backups didn't work
  - "Why was the backup procedure not tested on a regular basis? - Because there was no ownership, as a result nobody was responsible for testing this procedure."
- Redundancy is your friend
- Redundancy is *not* your accounting department's friend.

There is a complete write-up, including the DOS attack that led to the increased database load: [Postmortem of database outage of January 31 | GitLab]

Paxos Made Simple

Learning Goals (Paxos Made Simple)

Understand the Paxos distributed consensus algorithm

Discuss *variations* of consensus algorithms based on Paxos

Consensus Problem

Model:

- Asynchronous communications
- Non-Byzantine
- No fixed clock
- Messages can be lost, delayed, reordered, duplicated (but *not* corrupted)

Safety:

- Only proposed values can be chosen
- Only a single value is chosen
- A process learns a value if and only if it has been chosen

Terminology

Three **roles** in the algorithm:

- Proposer
- Acceptor
- Learner

Note: a *process* (or node) can play multiple roles, but must behave “correctly”

Proposer

Responsible for *proposing* values

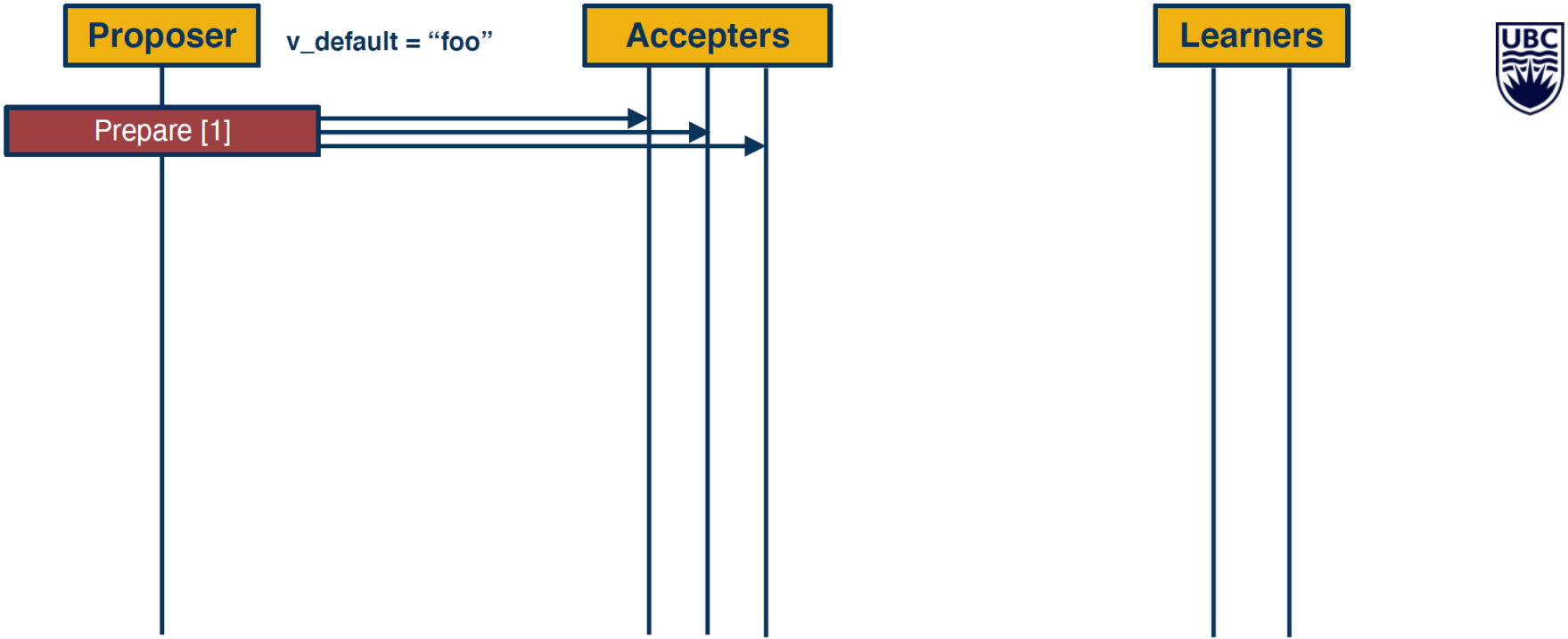
Submits *proposed* values to acceptors

Proposals are identified by a *number*

- The term “number” is quite general
- Proposal numbers are *ordered* in some fashion.
- See also: Lamport bakery algorithm.

Proposer *goal* is to get promises from a quorum of *acceptors*.

Proposal



Acceptor

Acceptors vote on the proposals received from proposers

Acceptor accepts a new proposal if:

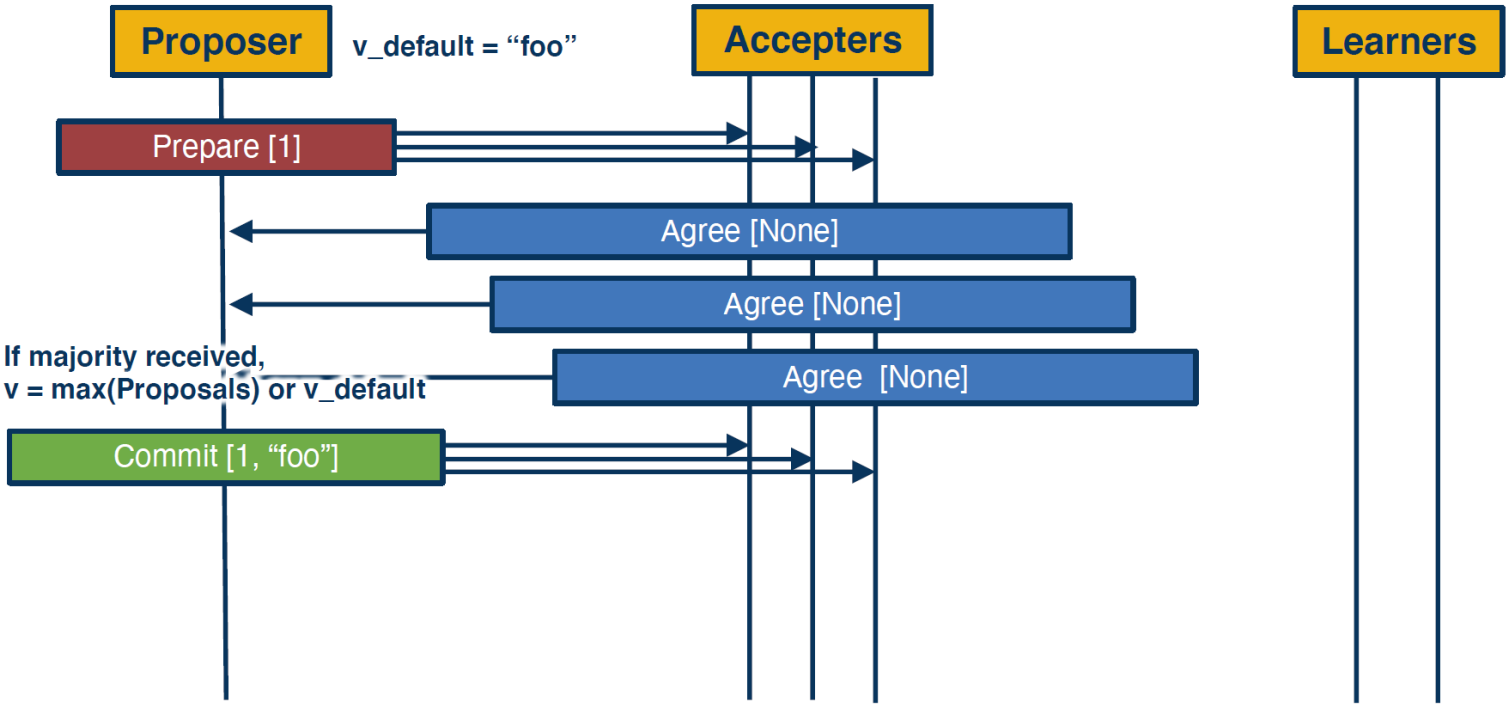
- This is the *first* proposal it has seen; or
- The new proposal contains a number *greater than* the previous accepted proposal

Note: “accepts” here means “sends a message back to the proposer”

Once a *quorum* of acceptors accepts a value, no other value is accepted.

- Additional acceptors will *only* accept the chosen value

Agreement

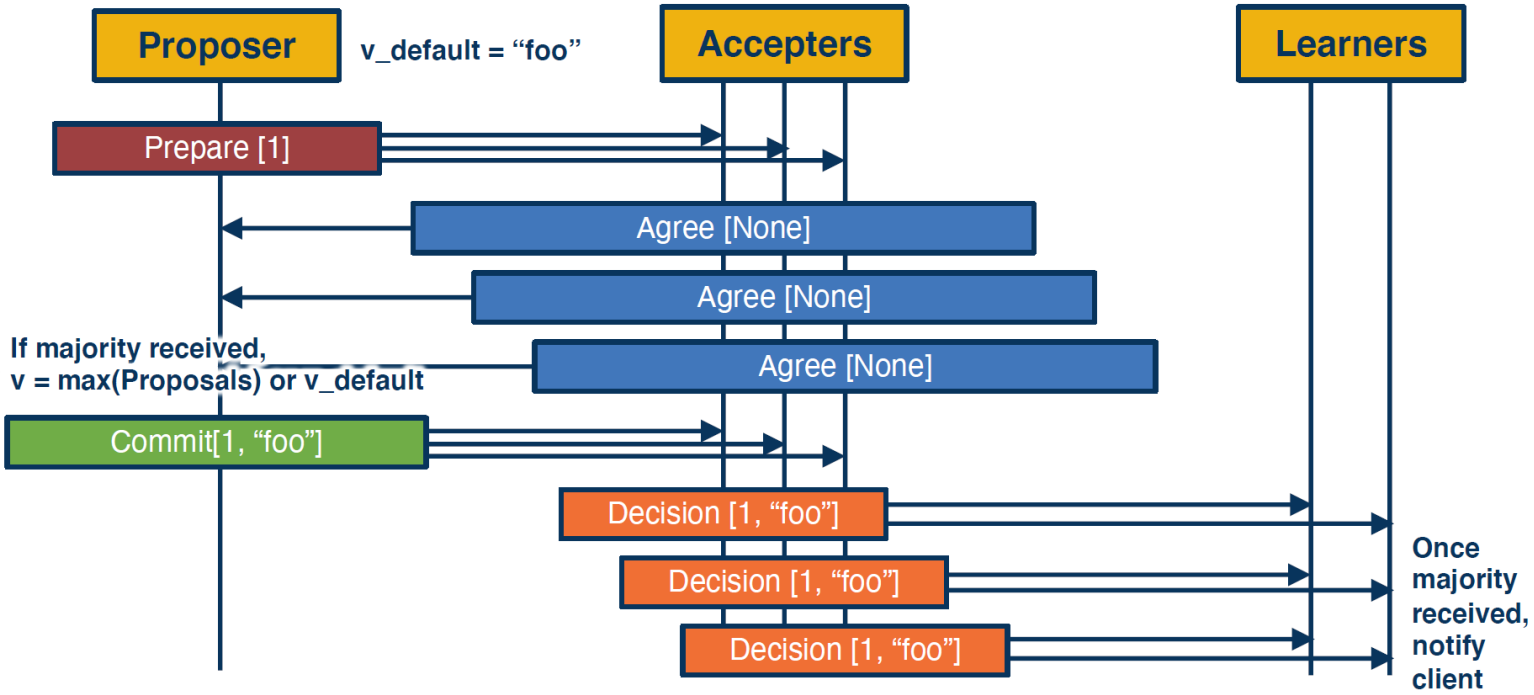

Proposer
v_default = “foo”
Accepters
Learners
Prepare [1]
Agree [None]
Agree [None]
If majority received,
v = max(Proposals) or v_default
Agree [None]
Commit [1, “foo”]

Learner

A *learner* is a node that sees accepted proposals

- Specific *mechanism* is not defined
- Acceptor(s) can push to their learners
- Learners can query acceptors
- Learners can *learn* from each other
- Learner can ask proposer to issue a proposal

Learning

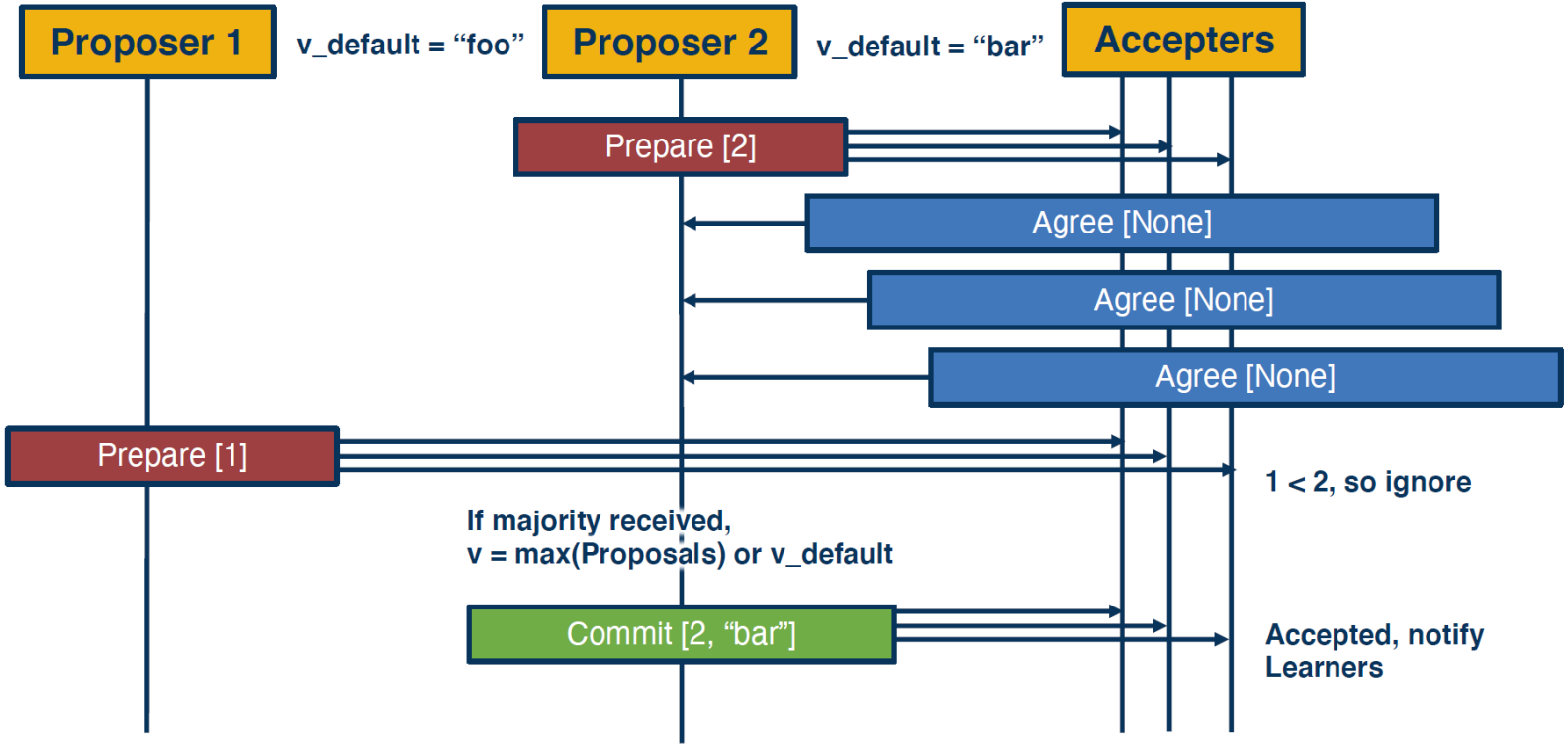

Proposer
v_default = “foo”
Accepters
Learners
Prepare [1]
Agree [None]
Agree [None]
If majority received,
v = max(Proposals) or v_default
Agree [None]
Commit[1, “foo”]
Decision [1, “foo”]
Decision [1, “foo”]
Decision [1, “foo”]
Once majority received, notify client

Edge Cases

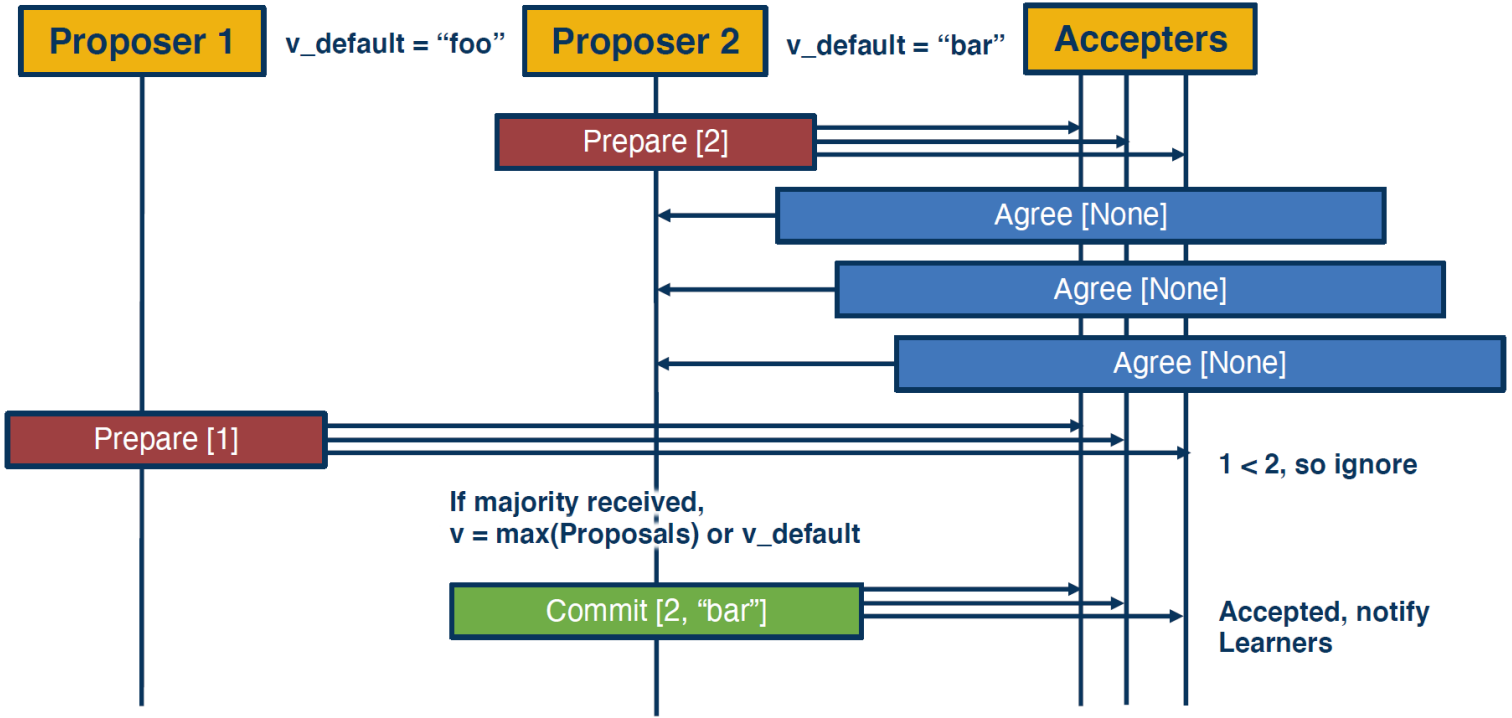
Proposers with *different* values

- Can only choose one!
- Ignore lower numbered proposal.

Challenging Cases: Different Values Proposed



Challenging Cases: Different Values Proposed



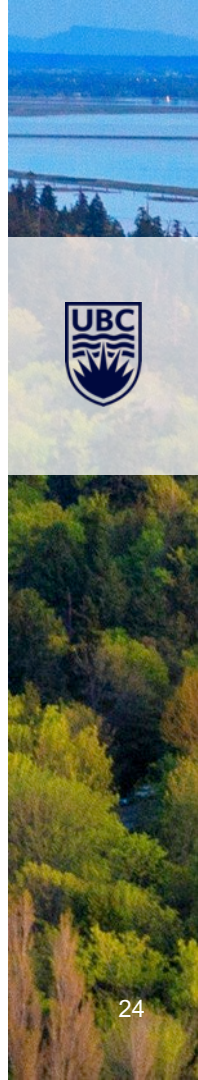
Multi-Paxos

Optimization:

- Select leader for a “view”
- All values in the view get accepted (and then learned) per Paxos
- Must detect and act upon view changes

Similar to Viewstamped Replication

See also Viewstamped Replication Revisited



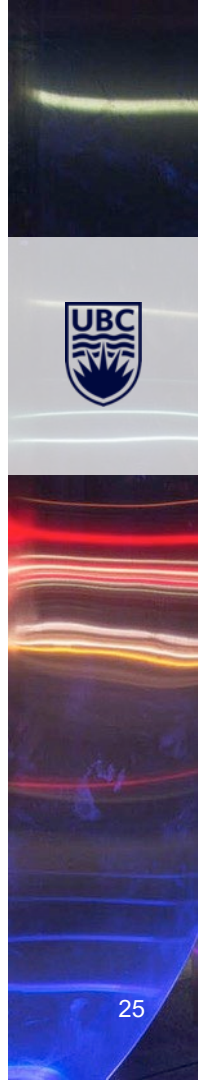
Using Paxos

Digital Equipment Corporation Systems Research Center (“DEC SRC”)

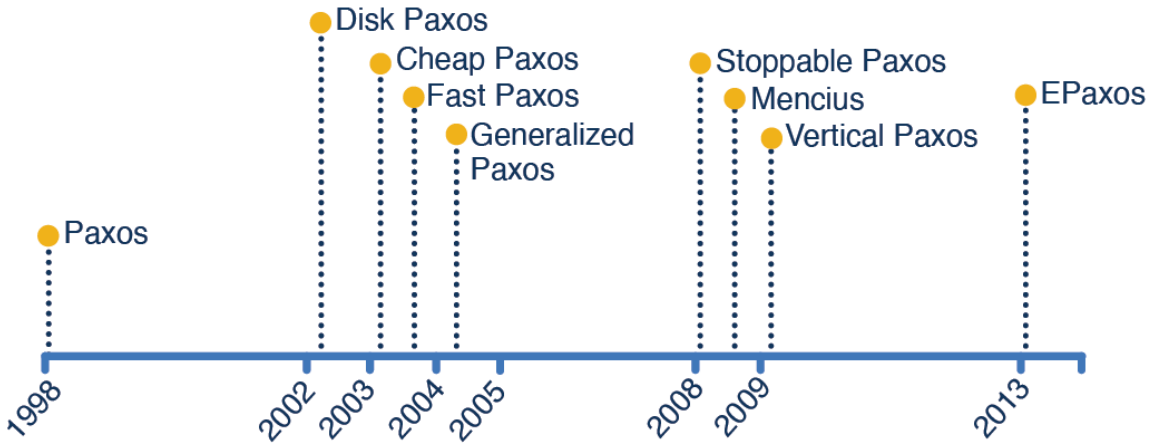
- Petal
- Frangipani
- Note: both used Paxos

Google Chubby

Zookeeper Atomic Broadcast (ZAB)



Paxos: Widely Used and Improved



Many research papers

- Paxos made live
- Paxos made transparent
- Paxos made moderately complex

Paxos in Action (animation)

Questions?


UBC
THE UNIVERSITY OF BRITISH COLUMBIA