CPSC 416 Distributed Systems

Winter 2023 Term 1 (November 7, 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

Self-Assessment

This week

- Usual self-assessment activity (Thu @ 17:00)
- DP3 Implementation (Code) (Thu @ 17:00) **Must compile**
- DP3 Implementation Report (Thu @ 23:59)

Next week

- No class (Tue 2023/11/14)
- Usual self-assessment activity (Thu @ 17:00)
- Capstone Week 5 Report (Thu @ 17:00)
- DP3 Implementation Report Peer Review (Thu @ 17:00)
- Capstone Project Team Declaration (Thu @ 17:00)

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.**

Final Exam

Official Final Exam: December 22, 2023, 19:00-22:00

- Format: Design Focused Questions
- You will be presented with a choice of four design questions
  - You must choose **one** design question and propose a solution to the scenario.
  - Grading based upon the thoroughness of your design analysis
    - Does it incorporate concepts/methods from the course.
    - Does it describe a solid design: Goals, non-Goals, proposed design's fitness for the scenario, quality of the analysis, validity of the proposed validation scheme.
- A set of sample design questions will be provided in early December.

Final Exam: Alternate Path

December 7, 2023

- Normal scheduled class time
- 75 minute **optional** exam
- Format:
  - Design Scenario focused
  - 20 True-False Questions
  - 20 Multiple-Choice Questions

Note: this will be a written examination. It is **entirely optional**.

- Your final exam grade will be the maximum of this alternate option or the actual final exam.
- You **do not** need to pass either final exam to pass the course.

Today’s Failure

Cloudflare

Event Start: 2023/10/30 18:58 UTC

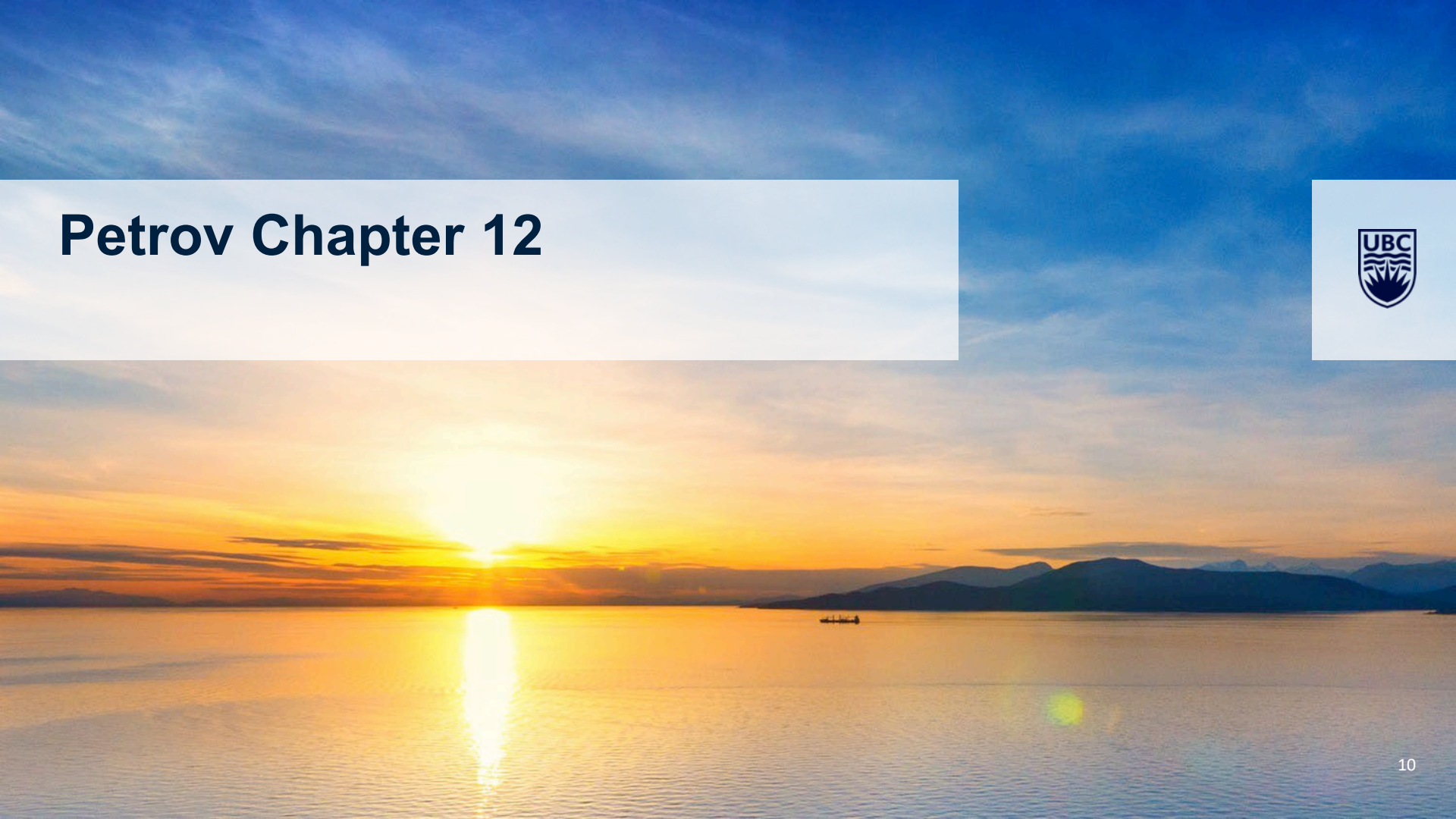
Event End: 2023/10/30 20:31 UTC

Duration: 93 minutes

Problem synopsis: “Workers KV is our globally distributed key-value store. It is used by both customers and Cloudflare teams alike to manage configuration data, routing lookups, static asset bundles, authentication tokens, and other data that needs low-latency access.
During this incident, KV returned what it believed was a valid HTTP 401 (Unauthorized) status code instead of the requested key-value pair(s) due to a bug in a new deployment tool used by KV.”

Source: Cloudflare incident on October 30, 2023

Petrov Chapter 12



Learning Goals (Petrov Chapter 12)

Understanding Communications Patterns in Distributed Systems

Propagation of Data

Cluster-wide Metadata

Scalability Issues

Anti-entropy mechanisms

Entropy in distributed systems

Background and foreground processes

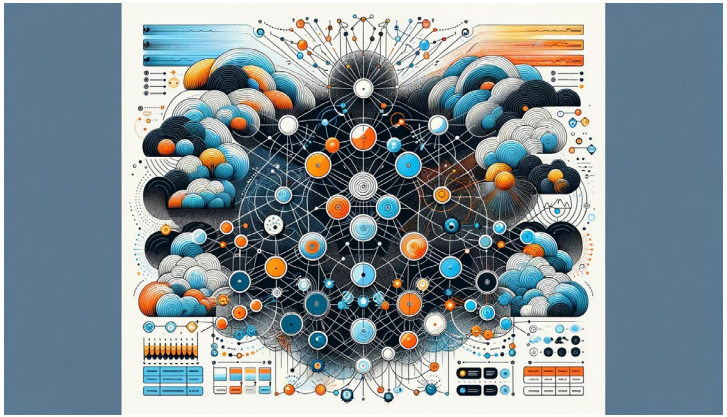
Intro to Anti-Entropy in Distributed Systems

Entropy in Distributed Systems:

- Measure of disorder or randomness
- Level of state divergence across nodes

Minimizing entropy

- Essential to maintain consistency
- Ensure up-to-date and correct data



Anti-entropy mechanisms: processes to reduce entropy

- Quite important in eventual consistency

Communication Patterns for Data Propagation

Peer-to-peer versus multicast:

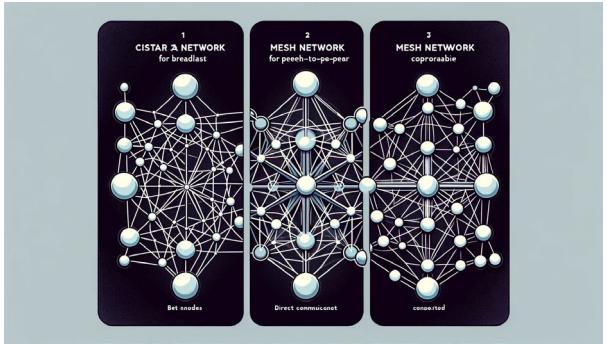
- Peer-to-peer: direct data exchange between nodes
- Multicasting: one source sends data to multiple recipients

Data Propagation

- Notification broadcast – sending updates from one node to others
- Periodic exchange – pair-wise periodic data synchronization
- Cooperative Broadcast – relay data between nodes

Scalability & Reliability:

- Communication pattern choice is key
- Appropriate for failure model



Large Distributed Systems Challenges

Broadcast inefficiency

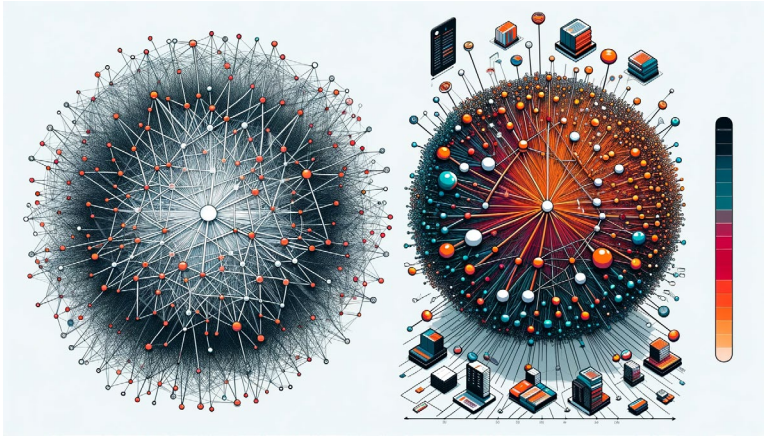
- Increase node count = inefficient/bottlenecks

Single process dependency

- Single point of failure
- Poor system resiliency/reliability

Speed/efficiency of metadata propagation

- Node members
- Schema changes
- Quick propagation
- Reliable update



Anti-Entropy

Eventual Consistency

- Great for performance
- Divergent state
- Anti-entropy facilitates state convergence

Background/Foreground Process

- Background
  - Independent operation
  - Merkle trees identify differences between nodes
- Foreground
  - Part of normal I/O
  - Read repairs – correct inconsistencies during data read

Entropy & State Divergence

Entropy

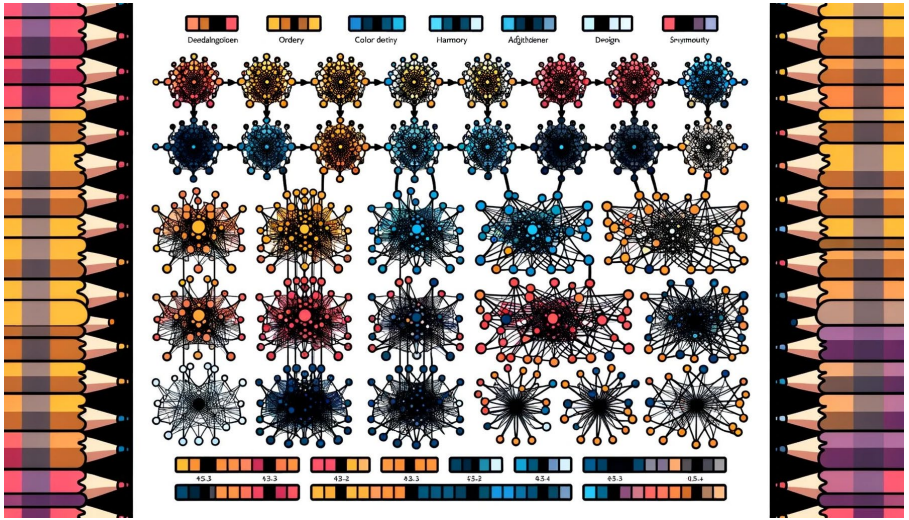
- Measure of system state divergence
- Level of inconsistency across the distributed system

Consequences

- Data conflicts
- Stale reads
- Loss of trust in data integrity

Manage Entropy

- Synchronization protocols
- Versioning
- Conflict resolution



Propagation Methods

Trade-offs

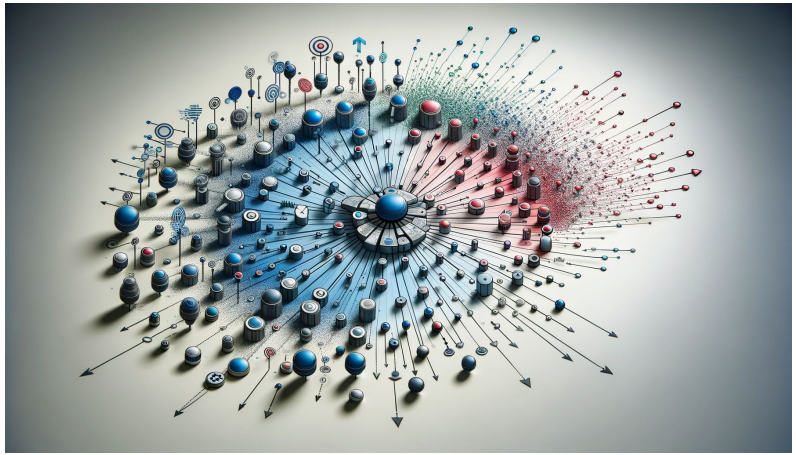
- Resource usage
- Speed of convergence
- System load

Node count

- More nodes = more complex (scalable) strategies

Reliability: eliminate single node dependencies

- Cooperative broadcast



Foreground versus Background

Foreground:

- Immediate consistency
- Critical read/write operations
- More expensive, stronger consistency

Background

- Eventual consistency
- Minimal impact to primary performance

Real-world:

- Foreground for systems with high consistency
- Background for systems where eventual consistency is acceptable

Handling Failures with Anti-Entropy

Failure types

- Network partitions
- Node loss
- Recovery strategies depend upon *type* of failure

Anti-Entropy & Recovery

- Ensure consistent data replication
- Disseminates data *across* all working nodes

Strategies

- Hinted handoff
- Full state synchronization
- Depends upon failure scenario

Performance Considerations

Consistency versus Performance

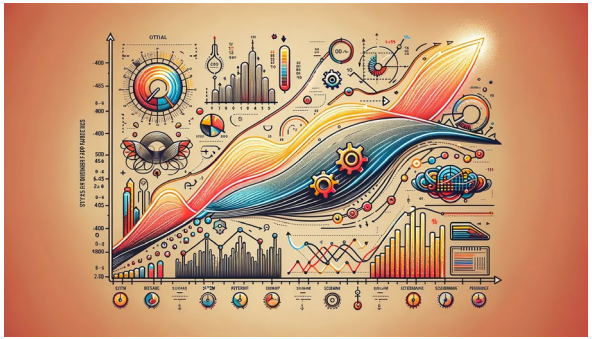
- Anti-entropy can be *expensive*
- High-load situations (“limited excess capacity”)

Scheduling

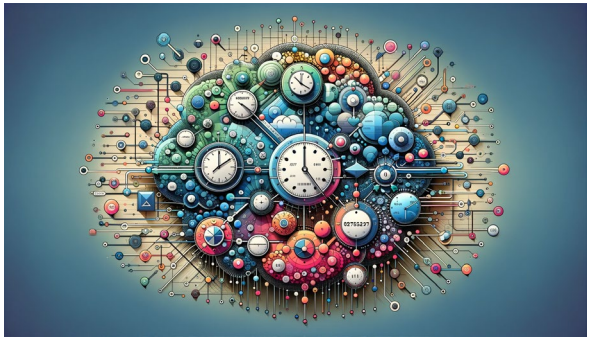
- Use idle/slack times
- Balance against strong consistency needs

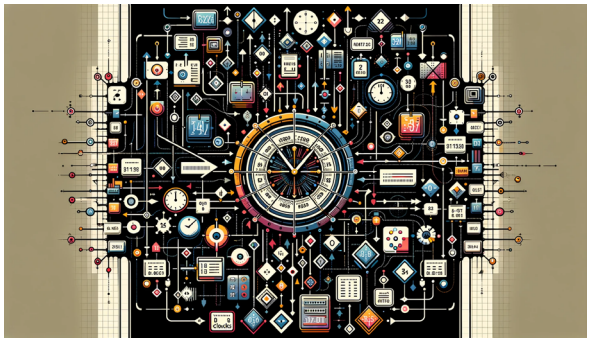
Minimizing impact

- Incremental synchronization
- Prioritize “critical” data



Advanced Techniques





State reconciliation

- Vector versioning
- Conflict-free replicated data types (CRDTs)
- Quorum recovery methods

Factors

- Data model complexity
- Network reliability
- Consistency requirements

Examples: DynamoDB, Cassandra

Questions?


UBC
THE UNIVERSITY OF BRITISH COLUMBIA