CSE 486/586 Distributed Systems

Consistency --- 3

Steve Ko
Computer Sciences and Engineering
University at Buffalo

Recap

- Consistency
  - Linearizability
  - Sequential consistency
- Chain replication
- Primary-backup (passive) replication
- Active replication

Two More Consistency Models

- Even more relaxed
  - We don't even care about providing an illusion of a single copy.
- Causal consistency
  - We care about ordering causally related write operations correctly.
- Eventual consistency
  - As long as we can say all replicas converge to the same copy eventually, we're fine.

Relaxing the Guarantees

- Do we need sequential consistency?

- Does everyone need to see these in this particular order? What kind of ordering matters?
  - Causal

Relaxing the Guarantees

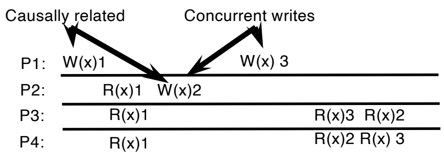
- For some applications, different clients (e.g., users) do not need to see the writes in the same order, but causality is still important (e.g., a post and a reply).
- Causal consistency
  - More relaxed than sequential consistency
  - Clients can read values out of order, i.e., it doesn't behave as a single copy anymore.
  - Clients read values in-order for causally-related writes.
- How do we define "causal relations" between two writes? (Hint: think about a message and a reply on a facebook wall---what events are involved?)
  - One client reads something that another client has written; then the client writes something.
  - Two writes from the same client

Causal Consistency

- If two writes are causally related, we apply those writes in the same order across all replicas.
- If two writes are not causally related (concurrent), then we don't need to apply those writes in the same order across all replicas.
- The storage system doesn't give an illusion that there is a single copy.

Causal Consistency

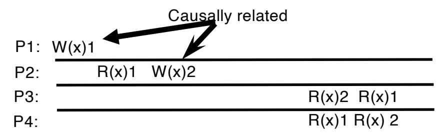
- Example 1:



This sequence obeys causal consistency

Causal Consistency Example 2

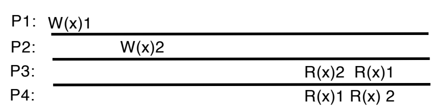
- Causally consistent?



- No!

Causal Consistency Example 3

- Causally consistent?



- Yes!

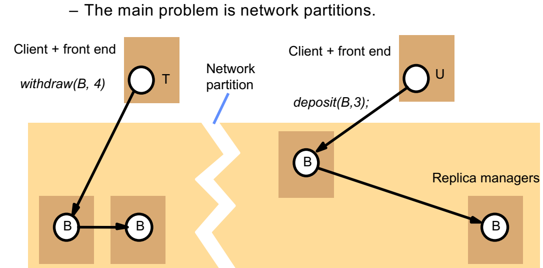
Implementing Causal Consistency

- We drop the notion of a single copy.
  - Writes can be applied in different orders across copies.
  - Causally-related writes do need to be applied in the same order for all copies.
- Need a mechanism to keep track of causally-related writes.
- Due to the relaxed requirements, low latency is more tractable.

CSE 486/586 Administrivia

Relaxing Even Further

- Let's just do best effort to make things consistent.
- Eventual consistency
  - Popularized by the CAP theorem.
  - The main problem is network partitions.



Dilemma

- In the presence of a network partition:
- In order to keep the replicas consistent, you need to block.
  - From an outside observer, the system appears to be unavailable.
- If we still serve the requests from two partitions, then the replicas will diverge.
  - The system is available, but no consistency.
- The CAP theorem explains this dilemma.

CAP Theorem

- Consistency
- Availability
  - Respond with a reasonable delay
- Partition tolerance
  - Even if the network gets partitioned
- In the presence of a partition, which one to choose? Consistency or availability?
- Brewer conjectured in 2000, then proven by Gilbert and Lynch in 2002.

Coping with CAP

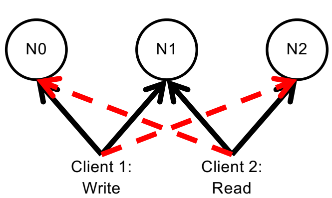
- The main issue is the Internet.
  - As the system grows to span geographically distributed areas, network partitioning sometimes happens.
- Then the choice is either giving up availability or consistency
- A design choice: What makes more sense to your scenario?
- Giving up availability and retaining consistency
  - Your system blocks until everything becomes consistent.
- Giving up consistency and retaining availability
  - Eventual consistency
  - Your system lets different partitions to serve read/write requests and later reconcile the differences.

Static Quorums

- A way to control partition behavior
- Provides control knobs in terms of how many replicas should be involved in an operation
- Quorum rules state that:
  - At least r replicas must be accessed for read
  - At least w replicas must be accessed for write
  - r + w > N, where N is the number of replicas
  - w > N/2
  - Each object has a version number or a consistent timestamp
- If the network is partitioned, a partition that has a majority can still function.
  - Smaller partitions can perhaps serve read requests
  - Providing partial availability and consistency

Static Quorums

- r = 2, w = 2, N = 3: r + w > N, w > N/2



Static Quorums

- What does r + w > N mean?
  - The only way to satisfy this condition is that there's always an overlap between the reader set and the write set.
  - There's always some replica that has the most recent write.
- What does w > N/2 mean?
  - When there's a network partition, only the partition with more than half of the replicas can perform write operations.
  - The rest will just serve reads with stale data.
- R and W are tunable:
  - E.g., N=3, r=1, w=3: High read throughput, perhaps at the cost of write throughput.

Summary

- Causal consistency & eventual consistency
- Quorums

Acknowledgements

- These slides contain material developed and copyrighted by Indranil Gupta (UIUC).