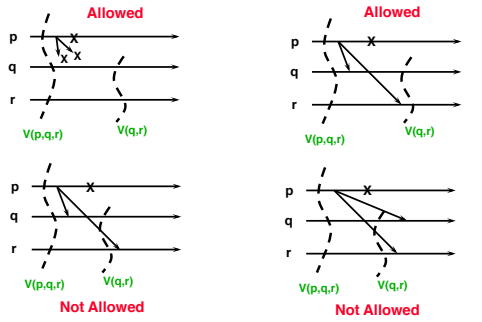
CSE 486/586 Distributed Systems
Consistency --- 1

Steve Ko
Computer Sciences and Engineering
University at Buffalo

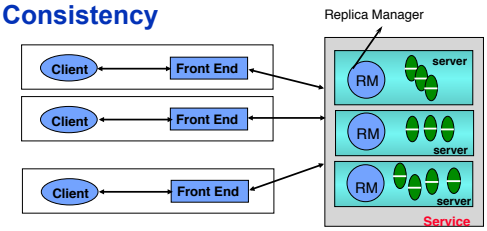
Recap

- Views?
  - Versioned membership
- View-synchronous group communication?
  - Providing group communication with a dynamic group
  - A way to design replicated state machines
  - "What happens in the view, stays in the view."

Examples



Consistency



- Consider that this is a storage service that serves read/write requests.
- Need consistent updates to all copies of object

Consistency Question

- How do we ensure that multiple copies have the same object?
- Let's think about this in terms of read/write operations…
- From the client's perspective, when do you know if an object has a new value?
- It depends on when writes become visible to reads.
- There are several guarantees we can provide.
  - Linearizability
  - Sequential consistency
  - Causal consistency
  - …
- We'll see the first two; and later the third.

Linearizability

- What would be the strongest (and probably most natural) form of consistency?
- Linearizability
  - A read operation returns the most recent write, regardless of the clients.
- Think of a single system read/write. What happens for a write followed by a read?

Linearizability Subtleties

- An operation takes time to finish.
  - E.g., a read op R starts at X ms and finishes at Y ms.
- A value written by a write operation becomes (physically) visible at some point during the operation.
  - E.g., a write op W starts at X ms and finishes at Y ms. At Z ms (X < Z < Y), the value gets actually written and becomes visible.
- What's a reasonable thing to do with this?
  - If W happens at X, R happens at Y, and X < Y, then R should read what W wrote.
  - If R finishes later than W but overlaps with W, then it can read either the previous value or the value written by W.

Linearizability Subtleties

- Write (black) & read (red)
- Definite guarantee
- Relaxed guarantee when overlap
- Case 1
- Case 2
- Case 3

Linearizability

- Let's say you're an oracle.
- Let your clients make requests (concurrent read/write).
- Let your system (with replicas) execute the requests.
- Write down the *real-time* execution of operations of your system. Two things to write down:
  - At what points in time each operation starts and ends.
  - Real-time precedence among operations: if A ends then B starts in real time, then A precedes B. (Caution: this is not a total order.)
- See if you can come up with an ordering of operations that meets three conditions:
  - All operations in the ordering appear one at a time as if each operation happened atomically.
  - The ordering gives the correct result as if it was done over a single copy.
  - The ordering preserves the real-time precedence of operations (i.e., the ordering written down from the above).

Linearizability

- Let the sequence of read and update operations that client i performs in some execution be oi1, oi2,….
  - "Program order" for the client
- (Textbook definition) A replicated shared object service is linearizable if for any execution (real), there is some interleaving of operations (virtual) issued by all clients that:
  - meets the specification of a single correct copy of objects
  - is consistent with the real times at which each operation occurred during the execution
- Main goal: any client will see (at any point of time) a copy of the object that is correct and consistent
- The strongest form of consistency

CSE 486/586 Administrivia

- PA3 deadline: 3/29 (Friday)
- Anonymous feedback form still available.
- Please come talk to me!

Sequential Consistency

- The question to answer in order to provide sequential consistency: can you come up with a interleaving of operations that preserve the program order?
- Example
  - P1: write A
  - P2: write B
  - P3: read B read A
  - P4: read B read A
- What's an interleaving that makes sense?

Sequential Consistency

- Example
  - P1: write A
  - P2: write B
  - P3: read B read A
  - P4: read A read B
- What's an interleaving that makes sense?

Sequential Consistency

- For understanding the intuition, rough verification of sequential consistency goes like the following.
- Let's say you're an oracle
- Run your system and get the result
- See if you can come up with an ordering of operations where
  - All operations in the ordering appear one at a time as if each operation happened atomically.
  - The ordering gives the correct result as if it was done over a single copy.
  - The ordering preserves the program order of each client.

Sequential Consistency

- Sequential consistency is less strict.
- (Textbook definition) A replicated shared object service is sequentially consistent if for any execution (real), there is some interleaving of clients' operations (virtual) that:
  - meets the specification of a single correct copy of objects
  - is consistent with the program order in which each individual client executes those operations.

Sequential Consistency

- This approach does not require absolute time or a single fixed total order.
  - Only that for each client the order in the sequence be consistent with that client's program order (~ FIFO).
- Linearilizability implies sequential consistency.
  - Not vice-versa!
- Challenge with guaranteeing seq. cons.?
  - Ensuring that all replicas of an object are consistent.

Summary

- Consistency
  - Linearizability
  - Sequential consistency

Acknowledgements

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