Article consistency

Paxos vs. Quorum-based Consistency

Lu Pan
16 Jul 2021 • 2 min read

Paxos, is the famous synonym for consistency in the context of distributed system. Unfortunately, *consistency* alone, is such an overloaded term, it's often practically meaningless. In this post, I will explain the difference between Paxos-consistency vs. quorum-consistency.

I assume you know what Paxos is and what problem it solves. If you need a quick refresher, or you are in the mood of trying to read about a new way of explaining how Paxos works, click this link.

When I say quorum-based consistency, I mean systems like Dynamo, Cassandra, etc. that claim to support "strong" consistency when you read and write to a quorum of the replica. E.g. if a shard has a replication factor of 3, we say read and write has quorum if every time you write to 2 replicas and read from at least 2 replicas synchronously. It seems straightforward. You always read the latest value – "strong consistency" (hopelessly overloaded). Or really?

Not-so-consistent Quorum

If there's a single keyword in distributed system, it's *order*. Multi-Paxos or Raft provides linearizability – an unambiguous description of one kind of "strong consistency" (applied to simple k/v stores without worrying about transactions).

Let's take a look at an example in quorum-based systems, e.g. Cassandra with a replication factor of 3.

1. client sets `x=24`. It sends the request to replica `A` and `B`. It received acknowledgements from both replicas.
2. client sets `x=42`. It sends the request to replica `B` and `C`. It received acknowledgements from both replicas.

That's enough. We already have a lot of interesting questions.

- Now from A's perspective, `x` should be `24`. C thinks `x` should be `42`. What about `B`?
- Should the "second" request `x=42` overwrite the "first" request `x=24`?
- Who said `x=24` "happens-before" `x=42`? What is it based on?
- Can we use client timestamp? No, unless you have TrueTime from Google.
- Can we use client sequence number? No, unless two requests are from the same client.
- Can we just let B decide which request happened first? Well, not really. If B is not available, a client reads from A and C, how can it figure out the order between the two requests?

Now you see the problem. The same problem doesn't exist in linearizable systems e.g. Multi-Paxos, or Raft.

Quorum-based consistency use case

That being said, quorum-based consistency can still be very appealing. First of all, it's simple and it's a huge deal. Second, if your use case doesn't care about ordering as much, quorum-based consistency can be a good fit. Not coincidentally, Dynamo (from Amazon) was developed to support the shopping cart use case. It doesn't matter, if you added shampoo or soap to your shopping cart first. As long as both items are there, it's fine.

Topic **consistency** **paxos** **quorum** Share

Show Comments

How FoundationDB works...
FoundationDB is a very impressive database. Its paper...
21 Jul 2021

Build `folly::coro` with GCC
You have heard about Coroutine in C++, and you want to use...
31 May 2021