Egalitarian Paxos

Iulian Moraru (student author), David G. Andersen, Michael Kaminsky
{imoraru, dga}@cs.cmu.edu, michael.e.kaminsky@intel.com
Carnegie Mellon University, Intel Labs

Abstract

We introduce Egalitarian Paxos, a new distributed consensus algorithm that achieves three goals: (1) optimal commit latency in the wide-area when tolerating one and two failures, under realistic conditions; (2) uniform load balancing across all replicas (thus achieving high throughput); and (3) graceful performance degradation and uninterrupted availability when replicas are slow or crash.

1 Problem Statement

Distributed computing today places two main demands on replication protocols: (1) high throughput for replication inside a computing cluster and (2) low latency for replication across data centers. Today's clusters use fault-tolerant, highly available coordination engines such as Chubby [2], Boxwood [6], or ZooKeeper [5] for activities including operation sequencing, coordination, leader election, and resource discovery. Modern databases are accessed simultaneously from different continents, requiring geo-replication [1, 4].

An important limitation in these systems is that during efficient, failure-free operation, all clients communicate with a single master (or leader) server at all times. This optimization, sometimes termed "Multi-Paxos," is important to achieving high throughput in practical systems [3]. Changing the leader requires invoking additional consensus mechanisms that substantially reduce throughput.

This algorithmic limitation has several important consequences. First, when performing geo-replication, clients will incur additional latency for communicating with a remote master. Second, it can impair scalability by placing a disproportionally high load on the master, which must process more messages than the other replicas [7]. Third, traditional Paxos variants are sensitive to both long-term and transient load spikes and network delays that increase latency at the master. Finally, this single-master optimization can harm availability: if the master fails, the system cannot service requests until a new master is elected. Previously proposed solutions such as partitioning or using proxy servers are undesirable because they restrict the type of operations the cluster can perform. For example, a partitioned cluster cannot perform atomic operations across partitions without using additional techniques.

2 Contribution

Egalitarian Paxos (EPaxos) has no designated leader process. Instead, clients can choose, at every step, which replica to submit a command to, and in most cases the command will be committed without interfering with other concurrent commands. This allows the system to evenly distribute the load to all replicas, eliminating the first bottleneck identified above (having one server that must be on the critical path for all communication). EPaxos's flexible load distribution is better able to handle permanently or transiently slow nodes than previous Paxos variants, as well as the latency heterogeneity caused by geographical distribution of replicas, substantially reducing both the median and tail commit latency (EPaxos has optimal median commit latency in the wide-area). Finally, the system can provide higher availability and higher performance under failures because there is no transient interruption because of leader election: there is no leader, and hence, no need for leader election, as long as more than half of the replicas are available.

References

[1] J. Baker, et al. Megastore: Providing scalable, highly available storage for interactive services. In *Proc. of CIDR*, pp. 223–234. 2011.

[2] M. Burrows. The Chubby lock service for loosely-coupled distributed systems. In *Proc. 7th USENIX OSDI*. Nov. 2006.

[3] T. D. Chandra, R. Griesemer, J. Redstone. Paxos made live: an engineering perspective. In *Proc. 26th ACM SOSP*, PODC '07, pp. 398–407. 2007.

[4] J. C. Corbett, et al. Spanner: Google's globally-distributed database. In *Proc. 10th USENIX OSDI*. 2012.

[5] P. Hunt, et al. ZooKeeper: wait-free coordination for internet-scale systems. In *Proc. USENIX ATC*, USENIX-ATC'10. 2010.

[6] J. MacCormick, et al. Boxwood: abstractions as the foundation for storage infrastructure. In *Proc. 6th USENIX OSDI*. Dec. 2004.

[7] Y. Mao, F. P. Junqueira, K. Marzullo. Mencius: building efficient replicated state machines for WANs. In *Proc. 8th USENIX OSDI*, pp. 369–384. Dec. 2008.