CATS: A Linearizable and Self-Organizing Key-Value Store

Cosmin Arad Tallat M. Shafaat Seif Haridi

Distributed key-value stores provide scalable, fault-tolerant, and self-organizing storage services, but fall short of guaranteeing linearizable consistency in partially synchronous, lossy, partitionable, and dynamic networks, when data is distributed and replicated automatically by the principle of consistent hashing [14]. This work introduces *consistent quorums* as a solution for achieving atomic consistency. We present the design and implementation of CATS, a key-value store which uses consistent quorums to guarantee linearizability and partition tolerance in such adverse and dynamic network conditions. CATS is *scalable*, *elastic*, and *self-organizing*; key properties for modern cloud storage middleware. Our system evaluation shows that consistency can be achieved with practical performance and modest overhead: 5% decrease in throughput for read-intensive workloads, and 25% throughput loss for write-intensive workloads. CATS delivers submillisecond operation latencies under light load, single-digit millisecond operation latencies at 50% load, and it sustains a throughput of one thousand operations per second, per server, while scaling linearly to hundreds of servers.

Distributed key-value stores, such as Cassandra [15] and Dynamo [8], employ principles from DHTs [21] to build scalable and self-managing data stores. In contrast to CATS, these systems chose availability over atomic consistency, hence only providing eventual consistency [22]. While eventual consistency is sufficient for some applications, the complexities of merging divergent replicas can be non-trivial. We avoid the complexities entailed by eventual consistency while providing scalable storage for critical applications which need atomic consistency, guaranteeing it at the cost of a modest decrease in throughput.

SoCC'13, 1–3 Oct. 2013, Santa Clara, California, USA.
ACM 978-1-4503-2428-1.
http://dx.doi.org/10.1145/2523616.2525945

To handle dynamic networks, atomic registers were extended by protocols such as RAMBO [17], RAMBO II [9], RDS [7] and DynaStore [1] to be reconfigurable. Similarly, SMART [16] enabled reconfiguration in replicated state machines. With consistent quorums we provide high-throughput read/write operations without paying the full cost of state machine replication which needs coordination for every operation. Moreover, our design does not depend on electing a single leader and the complexities that come with that [5]. While these systems can handle dynamism and provide atomic consistency, they are not scalable as they were not designed to partition the data across a large number of machines. The novelty of CATS is in extending the reconfiguration techniques contributed by these works, such that they can be used at large scale, in order to build a system that is completely decentralized and self-managing.

Distributed coordination systems such as Chubby [4] and ZooKeeper [11, 13], provide linearizability and crash-recovery, but are not scalable. Master-based key-value stores, such as Bigtable [6], HBase [12], and MongoDB [18], rely on a central server for coordination and data partitioning. Similarly, Spinnaker [19] uses Zookeeper [11]. Since these systems are centralized, their scalability is limited. In contrast, CATS is decentralized and all nodes are symmetric, allowing for unlimited scalability.

Similar to CATS, Scatter [10] is a scalable and consistent key-value store. Scatter employs an extra subsystem and policies to decide when to reconfigure replication groups. In contrast, CATS has a simpler and more efficient reconfiguration protocol – both in the number of messages and message delays – which does not require distributed transactions. We focus on consistent-hashing at the node level, which makes our approach directly implementable in existing key-value stores like Cassandra [15]. The main distinguishing advantage of CATS over Scatter is CATS' ability to handle network partitions and mergers, an aspect largely ignored in Scatter. Once network partitions cease, CATS merges partitioned subsystems into a single overlay, while Scatter will continue to operate as separate overlays. Where Scatter provides scalability and consistency, CATS provides scalability, consistency, and partition tolerance.

The design, implementation, and evaluation of CATS are available in a separate technical report [3, 2, 20].

References

[1] M. K. Aguilera, I. Keidar, D. Malkhi, and A. Shraer. Dynamic atomic storage without consensus. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, PODC '09, pages 17–25, New York, NY, USA, 2009. ACM.

[2] C. Arad. *Programming Model and Protocols for Reconfigurable Distributed Systems*. PhD thesis, KTH – Royal Institute of Technology, Stockholm, Sweden, June 2013.

[3] C. Arad, T. M. Shafaat, and S. Haridi. CATS: Linearizability and partition tolerance in scalable and self-organizing key-value stores. Technical Report T2012:04, Swedish Institute of Computer Science, 2012.

[4] M. Burrows. The Chubby lock service for loosely-coupled distributed systems. In *Proceedings of the 7th symposium on Operating systems design and implementation*, OSDI '06, pages 335–350, Berkeley, CA, USA, 2006. USENIX.

[5] T. D. Chandra, R. Griesemer, and J. Redstone. Paxos made live: an engineering perspective. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, PODC '07, pages 398–407, New York, NY, USA, 2007. ACM.

[6] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2):4:1–4:26, June 2008.

[7] G. V. Chockler, S. Gilbert, V. Gramoli, P. M. Musial, and A. A. Shvartsman. Reconfigurable distributed storage for dynamic networks. *J. Parallel Distrib. Comput.*, 69:100–116, January 2009.

[8] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, and W. Vogels. Dynamo: Amazon's highly available key-value store. In *Proceedings of Twenty-first Symposium on Operating Systems Principles*, SOSP '07, pages 205–220, New York, NY, USA, 2007. ACM.

[9] S. Gilbert, N. A. Lynch, and A. A. Shvartsman. Rambo II: rapidly reconfigurable atomic memory for dynamic networks. In *International Conference on Dependable Systems and Networks*, DSN '03, pages 259–268, 2003.

[10] L. Glendenning, I. Beschastnikh, A. Krishnamurthy, and T. Anderson. Scalable consistency in Scatter. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 15–28, New York, NY, USA, 2011. ACM.

[11] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed. ZooKeeper: wait-free coordination for internet-scale systems. In *Proceedings of the 2010 USENIX Annual Technical Conference*, ATC '10, pages 1–14, Berkeley, CA, USA, 2010. USENIX Association.

[12] HBase. `http://hbase.apache.org/`, 2012.

[13] F. P. Junqueira, B. C. Reed, and M. Serafini. Zab: High-performance broadcast for primary-backup systems. In *Proceedings of the 41st International Conference on Dependable Systems & Networks*, DSN '11, pages 245–256, Washington, DC, USA, 2011. IEEE Computer Society.

[14] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin. Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, STOC '97, pages 654–663, New York, NY, USA, 1997. ACM.

[15] A. Lakshman and P. Malik. Cassandra: a decentralized structured storage system. *SIGOPS Oper. Syst. Rev.*, 44(2):35–40, Apr. 2010.

[16] J. R. Lorch, A. Adya, W. J. Bolosky, R. Chaiken, J. R. Douceur, and J. Howell. The SMART way to migrate replicated stateful services. In *Proceedings of the 1st EuroSys European Conference on Computer Systems*, EuroSys '06, pages 103–115, New York, NY, USA, 2006. ACM.

[17] N. A. Lynch and A. A. Shvartsman. RAMBO: A reconfigurable atomic memory service for dynamic networks. In *Proceedings of the 16th International Conference on Distributed Computing*, DISC '02, pages 173–190, London, UK, 2002. Springer-Verlag.

[18] MongoDB. `http://www.mongodb.org/`, 2012.

[19] J. Rao, E. J. Shekita, and S. Tata. Using Paxos to build a scalable, consistent, and highly available datastore. *Proc. VLDB Endow.*, 4:243–254, Jan. 2011.

[20] T. M. Shafaat. *Partition Tolerance and Data Consistency in Structured Overlay Networks*. PhD thesis, KTH – Royal Institute of Technology, Stockholm, Sweden, June 2013.

[21] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '01, pages 149–160, New York, NY, USA, 2001. ACM.

[22] D. Terry, M. Theimer, K. Petersen, A. Demers, M. Spreitzer, and C. Hauser. Managing update conflicts in Bayou, a weakly connected replicated storage system. In *Proceedings of the fifteenth ACM symposium on Operating systems principles*, SOSP '95, pages 172–182, New York, NY, USA, 1995. ACM.