Role and Performance Evaluation of an Order Protocol in Building Replicated Databases

Ye Liu, Yingming Wang, Paul Ezhilchelvan
School of Computing
Newcastle University
Email: [Y.Liu197, Y.Wang303, Paul.Ezhilchelvan]@Newcastle.ac.uk

Jim Webber
Neo4j UK
London, SE1 0LH
Email: Jim.Webber@neo4j.com

Abstract

Theory of distributed computing has long established that (i) total ordering of messages or *Atomic Broadcast* and solving *Consensus* are reducible to each other under crash failures [1] (ii) the two phase commit (2PC) is only a simplified instance of consensus [2], and (iii) maximum throughput is achieved when message ordering is done over a logical, unidirectional ring network [3]. Leveraging these findings together for the first time to our knowledge, we address three principal challenges in building crash-resilient databases: concurrency control for 1-copy serialisability, 2PC implementation, and high throughput expected of modern day database systems. At the core of our approach is a ring-based total order protocol that we had designed and implemented. Database replicas use it to reach consensus on conflicting transactions that should be aborted to ensure serialisability and to atomically commit surviving transactions. We will present the architecture for managing database replication and then our protocol performance when replication degree is two and three, tolerating at most one replica crash. While 2-fold replication requires perfect crash detection, three-fold can do with weak detectors [1]. Performance evaluation will focus on response times for replicas to reach consensus on transactions to be aborted.

References

[1] T. D. Chandra and S. Toueg, "Unreliable failure detectors for reliable distributed systems," *Journal of the ACM (JACM)*, vol. 43, no. 2, pp. 225–267, 1996.

[2] J. Gray and L. Lamport, "Consensus on transaction commit," *ACM Transactions on Database Systems (TODS)*, vol. 31, no. 1, pp. 133–160, 2006.

[3] R. Guerraoui, R. R. Levy, B. Pochon, and V. Quéma, "Throughput optimal total order broadcast for cluster environments," *ACM Transactions on Computer Systems (TOCS)*, vol. 28, no. 2, pp. 1–32, 2010.