Managing Large Scale Transactional Data in The Cloud*

Divyakant Agrawal
Department of Computer Science
University of California,
Santa Barbara, CA 93106
agrawal@cs.ucsb.edu

Amr El Abbadi
Department of Computer Science
University of California,
Santa Barbara, CA 93106
amr@cs.ucsb.edu

October 31, 2014

At UC Santa Barbara, we have been exploring innovative methods for managing large data sets in the cloud, and more specifically in a scalable data center settings. Our research has focused on the development of data management methods that are fault-tolerant, efficient, scalable and elastic. These systems have been built and experimentally validate using local clusters of servers as well as on AWS.

The past decade has witnessed an increasing adoption of cloud database technology, ie, scalable and highly-available database systems that provide ACID guarantees for distributed transactions. However, only a small number of cloud databases provide strong consistency guarantees for distributed transactions, due to practical challenges that arise because of distributed lock management in the cloud setting, where failures are the norm, and human administration is minimal. Most distributed optimistic concurrency control proposals in the literature deal with distributed validation but still require the database to acquire locks during two-phase commit, when installing updates of a single transaction on multiple machines. In MaaT, we re-design optimistic concurrency control to eliminate any need for locking during two-phase commit, while handling the practical issues of resource waste due to transaction restarts, as well as the need for exclusive locking during the final phase of the two-phase commit. Our preliminary experimental results demonstrate that MaaT provides many practical advantages over lock-based methods in the cloud context [3].

Our most recent work has focused on fault-tolerance in the face of catastrophic disasters. This requires geo-replication of data across different data centers. Various geo-replication approaches have been recently proposed, including Spanner [1] and MDCC [2]. Unlike other approaches, we have recently proposed Replicated Commit [4], which provides a high level abstraction of the individual data centers, and specifically distinguishes between the low communication costs within a data center and the high communication costs across different data centers. This is in contrast to commercial enterprises that have been designed, largely, with the notion that the network between data centers is dedicated and fully controlled by the cloud operator (e.g. Google does not share its cross data center networks with other vendors). Thus we believe Replicated Commit offers the right abstraction for implementing geo-replication in the Cloud. Replicated Commit builds on the abstraction of viewing each data center as a single and autonomous entity with low cost communication within the data center, but with significantly more communication latency across data centers. In order to execute transactions, a commit protocol needs to be executed on the different objects being accessed. To achieve consistency across different copies of an object, a replication protocol needs to be executed. Replicated Commit uses two phase commit to atomically execute

*This work is partially supported by NSF Grants CNS-1053594 and IIS-1018637

transactions within a data center, when a transaction accesses multiple objects. For replication across data centers, Replicated Commit uses Paxos to ensure consistency across the different copies of objects in different data centers.

In general, maintaining consistency across data centers is expensive and requires wide-area communication. This renders current solutions to either settle for weaker forms of consistency or suffer from large delays. In Message Futures [5], we propose a strongly consistent concurrency control manager with low commit latency to ensure mutual consistency of replicas across data centers. By judicial message passing of relevant information and at opportune time intervals, Message Futures can also enforce different priority levels of access, where each data center experiences a commit latency relative to its priority. In fact, in many common cases, transactions can be committed locally without the need for any communication. An experimental evaluation of Message Futures on a geo-replicated multi-data center setting demonstrates that Message Futures achieves a commit latency around one RTT (Round-Trip Time) for data centers with identical priority, and a latency comparable to committing locally for high priority data centers.

Motivated by the challenges in reducing commit latency, more recently, we derived a lower-bound on commit latency, namely, that the sum of the commit latency of any two data centers is at least the Round-Trip Time (RTT) between them. We use the insights and lessons learned while deriving the lower-bound to develop a commit protocol, called Helios, that achieves low commit latencies. Helios actively exchanges transaction logs (history) between data centers. The received logs are used to decide whether a transaction can commit or not. The earliest point in the received logs that is needed to commit a transaction is decided by Helios to ensure a low commit latency. Helios is theoretically able to achieve the lower-bound commit latency. We are currently experimentally validating that in real world deployments, Helios has a commit latency that is close to the optimal.

Our research would clearly benefit from an experimental cloud platform, which would enable us to explore the various tradeoffs among our diverse approches as well as in comparison to other proposed systems.

References

[1] J.C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, JJ Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, et al. Spanner: Google's globally-distributed database. In *OSDI*, pages 251–264, 2012.

[2] Tim Kraska, Gene Pang, Michael Franklin, Samuel Madden, and Alan Fekete. Mdcc: Multi-data center consistency. In *EuroSys*, pages 113–126, 2013.

[3] Hatem A. Mahmoud, Vaibhav Arora, Faisal Nawab, Divyakant Agrawal, and Amr El Abbadi. Maat: Effective and scalable coordination of distributed transactions in the cloud. *PVLDB*, 7(5):329–340, 2014.

[4] Hatem A. Mahmoud, Faisal Nawab, Alexander Pucher, Divyakant Agrawal, and Amr El Abbadi. Low-latency multi-datacenter databases using replicated commit. *Proceedings of the VLDB Endowment*, 6(9):661–672, 2013.

[5] Faisal Nawab, Divyakant Agrawal, and Amr El Abbadi. Message futures: Fast commitment of transactions in multi-datacenter environments. In *CIDR 2013, Sixth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 6-9, 2013, Online Proceedings*, 2013.