Inner CALM: Concurrency control protocols through the looking glass

Peter Alvaro

ABSTRACT

The CALM theorem—which states that monotonic programs produce consistent distributed outcomes without coordination [2, 4]—originated with the (mirror image) observation that mechanisms for ensuring consistency (e.g., concurrency control and other coordination protocols) cannot be implemented with purely monotonic logic [5]. Since that time, CALM has been a useful guiding principle for deciding whether costly protocols are necessary to ensure correct program outcomes [1, 3, 10]. Recalling CALM's roots in protocol implementation, we might be tempted to ask if we can apply a finer-grained monotonicity analysis to study the implementations of coordination mechanisms themselves. For example, do certain concurrency control protocols fundamentally require *more* coordination or synchronization than others?

In 2011 and again in 2012, Joe Hellerstein and I taught a class at Berkeley that presented fundamental distributed systems concepts and mechanisms through the lens of software development in the Bloom language. Students began with a simple key/value store implemented in Bloom; class projects continually enriched the store with additional mechanisms, including reliable and causal delivery, quorum replication, multi-key transactions supported by 2PL and MVCC, and commit protocols. Along the way, the students turned the CALM static analysis machinery inwards, using it to study the coordination protocol implementations themselves.

Some interesting results emerged that we will share in this talk. Monotonicity analysis allowed us isolate and study the *fundamental cost* of concurrency control protocols—namely, when (and for what) processes are forced to wait, hence losing the opportunity to do useful local work. The results of such analyses are invariant to the scale (ranging from processor cache to planetary) of the systems that use them, as well as to underlying (but rapidly changing) system assumptions about capacity, random I/O latency and network reliability.

Of particular interest is our assessment of the MVCC protocols, which can be expressed almost entirely using monotonic logic: CALM analysis indicates that waiting is fundamentally only required for timestamp assignment. Recent hardware trends make many of the historical shortcomings of MVCC, including random I/Os and storage requirements, less concerning—together, these observations help to explain the renewed interest in multi-valued concurrency control in large-scale distributed storage systems [6–9].

7^{th} *Biennial Conference on Innovative Data Systems Research (CIDR '15)* January 4-7, 2015, Asilomar, California, USA.

1. REFERENCES

[1] S. Abiteboul, E. Antoine, and J. Stoyanovich. The Webdamlog System Managing Distributed Knowledge on the Web. *CoRR*, abs/1304.4187, 2013.

[2] P. Alvaro, N. Conway, J. Hellerstein, and W. R. Marczak. Consistency Analysis in Bloom: a CALM and Collected Approach. In *CIDR*, 2011.

[3] P. Alvaro, N. Conway, J. M. Hellerstein, and D. Maier. Blazes: Coordination analysis for distributed programs. In *ICDE*, 2014.

[4] T. J. Ameloot, F. Neven, and J. Van den Bussche. Relational Transducers for Declarative Networking. In *PODS*, 2011.

[5] J. M. Hellerstein. The Declarative Imperative: Experiences and conjectures in distributed logic. *SIGMOD Record*, 2010.

[6] P.-A. Larson, S. Blanas, C. Diaconu, C. Freedman, J. M. Patel, and M. Zwilling. High-performance Concurrency Control Mechanisms for Main-memory Databases. *Proc. VLDB Endow.*, Dec. 2011.

[7] J. Lee, M. Muehle, N. May, F. Faerber, V. Sikka, H. Plattner, J. Krüger, and M. Grund. High-Performance Transaction Processing in SAP HANA. *IEEE Data Eng. Bull.*, 2013.

[8] W. Lloyd, M. J. Freedman, M. Kaminsky, and D. G. Andersen. Stronger Semantics for Low-latency Geo-replicated Storage. NSDI'13.

[9] Y. Sovran, R. Power, M. K. Aguilera, and J. Li. Transactional Storage for Geo-replicated Systems. SOSP '11.

[10] M. Takada. Distributed Systems for Fun and Profit. `http://book.mixu.net/distsys/`, 2012.