Client-centric benchmarking of eventual consistency for cloud storage systems

Wojciech Golab1, Muntasir Raihan Rahman2, Alvin AuYoung3, Kimberly Keeton3, Jay J. Wylie4, and Indranil Gupta2

1University of Waterloo, wgolab@uwaterloo.ca
2University of Illinois at Urbana-Champaign, {mrahman2,indy}@illinois.edu
3HP Labs, Palo Alto, firstname.lastname@hp.com
4LinkedIn Inc., jwylie@linkedin.com

Eventually consistent storage systems give up the ACID semantics of conventional databases in order to gain better scalability, higher availability, and lower latency. A side-effect of this design decision is that application developers must deal with stale or out of order data. As a result, substantial intellectual effort has been devoted to studying the behavior of eventually consistent systems, in particular finding quantitative answers to the questions “how eventual” and “how consistent”?

Existing tools for evaluating eventual consistency have two primary shortcomings. First, they measure deviation from strong consistency based on the behavior of the storage system using either a white-box system model [2] or active measurement [4], as opposed to the behavior of the client. Second, these tools do not capture the lack of precision inherent in measuring consistency from various vantage points, such as a collection of storage servers and clients. To overcome the first shortcoming, in our earlier work [5], we proposed the Δ (Delta) metric [3], which captures the “deviation” of a given execution trace from Lamport’s atomicity property (see [5] for more details). The Δ-atomicity property allows reads to return values up to Δ time units stale.

To overcome the second shortcoming, in this work, we propose the Γ (Gamma) metric which can quantify both stale reads and out of order write operations simultaneously. In contrast to the Δ computation, where we just shift the start point of each read to the left, in the Γ metric computation, we shift both the start point to the left and the finish point to the right for every operation. Due to this, the Γ metric can quantify how badly out of order write operations can be compared to the “happens before” order in a trace, thereby overcoming both shortcomings.

Figure 1 demonstrates a separation between Γ and Δ scores2 for the same trace for an experiment using Cassandra [1]. We have also shown that the Γ metric is sensitive to various configuration and workload parameters.

1This work is supported in part by the NSERC Discovery Grants Program, in part by AFOSR/AFRL grant FA8750-11-2-0084, and in part by NSF CCF grant 0964471. We are grateful to Ashraf Aboulnaga, Robert van Renesse and Hakim Weatherspoon for computing support and insightful comments.

2This work is supported in part by the NSERC Discovery Grants Program, in part by AFOSR/AFRL grant FA8750-11-2-0084, and in part by NSF CCF grant 0964471. We are grateful to Ashraf Aboulnaga, Robert van Renesse and Hakim Weatherspoon for computing support and insightful comments.

Copyright © 2013 by the Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

SoCC’13, 1–3 Oct. 2013, Santa Clara, California, USA.
ACM 978-1-4503-2428-1.
http://dx.doi.org/10.1145/2523616.2525935

3Please see [5, 3] for the technical definition of score functions.
References


