Comparing SSD-placement strategies to scale a Database-in-the-Cloud

Yingyi Bu[*]
University of California, Irvine
yingyib@ics.uci.edu

Hongrae Lee
Google Inc.
hrlee@google.com

Jayant Madhavan
Google Inc.
jayant@google.com

ABSTRACT

Flash memory solid state drives (SSDs) have increasingly been advocated and adopted as a means of speeding up and scaling up data-driven applications. However, given the layered software architecture of cloud-based services, there are a number of options available for placing SSDs. In this work, we studied the trade-offs involved in different SSD placement strategies, their impact of response time and throughput, and ultimately the potential in achieving scalability in Google Fusion Tables (GFT), a cloud-based service for data management and visualization [1].

1. GFT ARCHITECTURE

The GFT system is built on top of cloud storage layers such as Colossus (a distributed file system – DFS) and Bigtable (a key-value store) that provide persistent storage and transparent replication. Our frontend servers have very stringent requirements on the respond time, e.g., 100 milliseconds, to support interactive visualizations. To meet the tight latency bound, GFT has an in-memory column-oriented query execution servers (QESs). Datasets are entirely loaded and indexed on demand into the QES column store. This architecture, though simpler, limits our ability to scale to (a) large individual datasets, and to (b) large numbers of simultaneously active datasets. Our goal is to use SSDs as a means to address both these challenges.

2. SSD PLACEMENT STRATEGIES

We explored the following placement strategies:

[*]Work done at Google Inc.

SoCC'13, 1–3 Oct. 2013, Santa Clara, California, USA.
ACM 978-1-4503-2428-1. http://dx.doi.org/10.1145/2523616.2525949

SSD strategy	Improves loading throughput	Speeds-up data loading	Scales-up to larger datasets	Offers cache reliability	Cache response time
KS1	Yes	limited	No	N/A	N/A
KS2	Yes	limited	No	N/A	N/A
QS3	No	No	Yes	No	low
QS4	No	No	Yes	Yes	high
QS5	No	No	Yes	Yes	medium

Table 1: A comparison of SSD placement strategies

- **KS1**. Bigtable on SSD: The log and data files of Bigtable are stored on SSD-powered DFS.
- **KS2**. Bigtable cache on SSD-powered local file system (LFS): The internal cache of a Bigtable is on locally attached SSDs.
- **QS3**. QES column store on SSD-powered LFS: Column arrays in QES are placed on local SSDs.
- **QS4**. QES column store on SSD-powered DFS: Similar to QS3, except the column arrays are placed in SSD-powered DFS.
- **QS5**. QES column store on SSD-backed Bigtable: The table content and column indices for the QES are loaded into a Bigtable backed by SSD-powered DFS, such that column store accesses correspond to Bigtable reads/writes.

We identified experiments that isolated the benefits (and downsides) of each configuration. Our findings are summarized in Table 1. KS1 and KS2 enable faster loading of datasets into the QES, but no scaling. QS4 and QS5 offer opportunities for sharing column arrays between different QESs, but incur higher response time as compared to QS3.

3. CONCLUSION

We found that to meet our latency needs in GFT, the QS3 strategy was the most suitable. We further explored changes that were needed to our column store to better realize the potential of the locally placed SSDs. Our observations and guidelines, though made in the contents of GFT, are largely applicable many cloud-based data management services in general.

4. REFERENCES

[1] H. Gonzalez, A. Y. Halevy, C. S. Jensen, A. Langen, J. Madhavan, R. Shapley, and W. Shen. Google Fusion Tables: data management, integration and collaboration in the cloud. In *SoCC*, pages 175–180, 2010.