Summary: Benchmarking Cloud Serving Systems with YCSB

Chris Willmore
Software Engineering Seminar, Fall 2011

1 Introduction

Cloud storage systems represent a new paradigm in distributed data storage. In contrast to traditional ACID-based storage, these systems focus on simple, flexible record types, extremely high availability, and online scaling. By emphasizing these qualities, cloud storage solutions often sacrifice hallmark features of ACID databases, such as transactions. Application developers require benchmarks which can empirically identify a cloud data serving platform which best fits anticipated usage patterns. Seemingly related benchmarks for relational / ACID databases such as the TPC-* family are ill-suited for the task.

In the article by Cooper et al. [1], the authors first analyze a range of dimensions in which cloud storage solutions differ. The authors then propose benchmarks to test such systems that can empirically identify a cloud data serving platform which best fits anticipated usage patterns. Seemingly related benchmarks for relational / ACID databases such as the TPC-* family are ill-suited for the task.

In the article by Cooper et al. [1] the authors first analyze a range of dimensions in which cloud storage solutions differ. The authors then propose benchmarks to test such systems that mimic common use cases. The authors introduce an implementation of these benchmarks in the form of a tool called the Yahoo! Cloud Serving Benchmark (YCSB) which they use to test several cloud platforms.

2 Cloud Database Design

There has been an explosion in recent years in the number and variety of cloud storage systems, primarily driven by the demands of large web companies such as Google, Yahoo! and Facebook. Web sites serving millions of requests per day require storage that both scales well and performs operations with the low latency users expect. In addition to a wide range of feature sets, cloud storage systems range widely in the usage they are optimized for. In the paper summarized, the authors focus on cloud data platforms designed for user facing services, as opposed to batch systems like Hadoop.

The key qualities of cloud systems are scale-out, elasticity, and high availability. Scale-out refers to the ability of a system to increase both data and transaction capacity linearly with the system size. The system is almost always composed of commodity hardware. Elasticity is the ability of a system to grow while still supporting live traffic. A live system should programmatically redistribute load to make use of the new capacity. Elasticity supports the final quality of high availability. In Service Oriented Architectures, cloud services are used by multiple other services which in turn themselves have high availability requirements. Could services must be resilient to both sudden shifts in demand and the likely failure of commodity components.

A system cannot be optimized for all use cases. The authors identify the following tradeoff when designing a cloud storage system:

- Read vs. write performance
- Latency vs. durability
- Synchronous vs asynchronous replication
- Data partitioning (column-based vs. row-based)

3 Benchmark Definition

The authors propose two initial benchmark tiers: performance and scaling. The performance benchmark measures how a system’s transaction latency degrades as throughput is increased. This measurement is made by increasing the request rate on a system with fixed hardware capacity. The second tier of benchmarking, scaling, can be quantified in terms of scaleup and elastic speedup. Scaleup is a measurement of a system’s performance change when the capacity and load are proportionally increased. The expectation is that system performance should remain constant. Elastic speedup measures how performance changes when new machines are added to a live system.
The benchmarking measurements are conducted using workloads which define the composition of requests. A single request is defined in terms of the type of operation and the data being operated on. The YCSB tool provides four types of record operations: insert, update, read, and scan. The scan operation represents a range query, which is not supported by all cloud platforms. YCSB is architected as a plugin framework in order to accommodate testing present and future systems.

When making a decision about which operation to use and which record to access, YCSB makes use of one of several random distributions. The uniform distribution gives equal weight to all records. The Zipfian distribution favors a small set of records. The latest distribution is similar to Zipfian, but favoring most recently inserted records. The multinominal distribution allows flexible user-defined distributions. A user may mimic write-heavy traffic by configuring a test with 90% writes and 10% reads. YCSB allows the user to choose from any combination of predefined workloads or extend the tool to define his own. By this method the user may test competing storage products against a predicted or known application usage pattern.

4 Methodology and Results

The authors compared benchmark results of four systems with YCSB: Cassandra, HBase, PNUTS, and sharded MySQL. For a description of the systems and the authors’ classification with regards to design tradeoffs, the reader is referred to the paper. It is important to note that the sharded MySQL system is not an elastic cloud platform, but is included as it is a popular solution to “cloud scale” loads.

The benchmarks were conducted using server class machines. The authors optimized each system’s configuration to the best of their ability, sometimes with the assistance of the system developers. Five different workloads of varying operation type ratios and data distributions were employed. The authors applied the workloads in three scenarios: static resource count, scaling resources, and elastically scaling resources.

In the first scenario, each system was installed on a set of six machines and benchmarked against the various workloads, revealing strengths and weaknesses. These results generally conformed to the authors’ expectations. For example, Cassandra which is optimized for reads, performed better on read heavy workloads compared to HBase which is optimized for writes.

The authors measured scaleup by testing the systems on sets of 2 to 12 machines and increasing the dataset proportionally. Results show the systems generally maintained consistent throughputs, as expected.

Elasticity results revealed strong differences between the systems. In each test, the machines were added to a live system and the changes in request latency monitored as the system reconfigured itself. When benchmarking Cassandra, the authors had to pause the test as the system was incapable of redistributing data and satisfying requests with acceptable latency. HBase and PNUTS demonstrated a much quicker adaptation to the addition of new hardware.

5 Conclusions

The number and variety of cloud data service systems place a large burden on architects and developers to choose the right platform for their application. The authors of YCSB have provided a well structured classification for cloud services and developed benchmarks to assist developers when choosing a system. The YCSB is a flexible and extensible implementation of the proposed benchmarks.

References