CloudCmp: Comparing Public Cloud Providers
A. Li, X. Yang, S. Kandula, M. Zang

Katrina LaCurts, MIT (6.897 Spring 2011)

Choosing a public cloud provider is a difficult problem. How is a customer meant to compare public cloud providers when each offers a myriad of services, instace types, and pricing schemes? Even if a customer could make a systematic comparison, would they be able to discern which cloud provider was best—in terms of cost-effectiveness or otherwise—for a particular application?

CloudCmp, published at IMC 2010, provides a systematic framework for comparing cloud providers. Li at al. apply their methods and compare Amazon AWS, Rackspace CloudServers, Google App-Engine, and Microsoft Azure[1]. Rather than compare the different feature sets (e.g., which programming languages are offered by each provider, the type of tech support available), CloudCmp measures the performance of the core services that are common to the four providers: the elastic compute cluster, persistent storage (table storage, which offers key/value functionality; blob storage, which allows users to store unstructured data; and queue storage, which is designed to store messages that are later passed between cloud isntances), and networking (both intra-cloud and WAN). Their main finding is that no one cloud is best; different clouds are suited for different applications. More specifically, they find the following:

- Based on finishing times for well-known Java benchmarks (SPECjvm2008), different providers offer different levels of cost-effectiveness. For example, a low-end instance of Microsoft Azure is 30% more expensive than the comparable Amazon AWS instance, but able to finish tasks twice as quickly. Moreover, higher-end instances within one cloud do not necessarily offer better performance across all benchmarks. As a result, parallel applications *might* do well to use more low-end instances rather than a few high-end ones.
- The performance of different storage types (table, blob, and queue) is highly variable. For example, AWS's response time for table queries is orders of magnitude faster than any other provider, but it takes longer for their tables to reach consistency (around 100ms on average, but sometimes as high as 500ms; other providers see no such inconsistencies). Blob storage and queue storage see similar variability.
- The wide-area network latency roughly corresponds to geographic location, suggesting that the more geographically diverse a cloud provider is, the better. Throughput, both intra-cloud and wide-area, is relatively consistent across all providers, except for Rackspace, which seems to have a rather under-provisioned network in these measurements; its intra-datacenter TCP throughput is between 600 and 700Mbps lower than all other providers, and its WAN throughput is between 100 and 200Mbps lower than all other providers.

Given these results, it's still difficult to pick a "winning" public cloud provider. No provider is best overall, and it's not obvious which providers are suited for which applications. However, the authors apply the results of their study to specific applications. For example, an e-commerce website deals with many table operations, but is less concerned with networking throughput or latency. Based on the results from CloudCmp, e-commerce sites should use AWS; it has the best performance on table storage (the paper verifies this conclusion by running a set of e-commerce-specific benchmarks on each cloud). The paper gives similar examples for parallel scientific computing applications and

[1]The authors list their results in terms of clouds C_1, C_2, C_3, and C_4. We give our own inferences as to which provider each C_i corresponds to.

latency-sensitive websites. Scientific computing applications are CPU-intensive, and thus should use a low-end instance of Microsoft Azure; Latency-sensitive websites should use Google AppEngine, which has the lowest wide-area network latency distribution.

Overall, CloudCmp is a good first step in comparing public cloud providers. The features it measures are likely to be common amongst all cloud providers, not just the four studied in this paper, and the comparison to real-world applications is helpful in giving more insight into their results. However, the paper lacks the "stress test" of running a large distributed application (e.g., MapReduce) on a cloud. It's also not clear how far in the future the measurements will hold, or whether average consumers would find this performance comparison as useful as a simple feature comparison between cloud providers.