Review of NetCache: Balancing Key-Value Stores with Fast In-Network Caching

Jake Pitkin
University of Utah

November 18, 2018

Introduction

NetCache is a novel approach to a caching layer for a key-value store distributed across a storage cluster. The innovation this work provides comes in where they place the cache: inside a network switch. Network switches are becoming increasingly programmable and NetCache can run at line rate: processing 2+ billion queries per second for 65k items with 16-byte keys and 129-byte values on a single switch. A dedicated network switch is programmed to contain NetCache and is placed on the path between clients and the storage cluster.

Rather than replicate hot items, NetCache instead keeps a single copy of each key-value pair. Without a caching layer, this highly-skewed workload will leave the servers with the key-value extremely over worked while leaving the other servers under utilized. NetCache's goal is to statistically detect these hot items and place them in the cache. For example, prior studies have shown that 10% of items account for $60 - 90\%$ of queries in the Memcached deployment at Facebook [1]. Only 10% of the items being hot at a given time works in favor of NetCache as network switches have limited on-chip memory. As such, NetCache uses switches as a load-balancing cache with medium cache hit ratio ($< 50\%$) with a goal of caching these very hot items. The authors show that NetCache improves the throughput by $3 - 10x$ and reduces the latency of up to 40% of queries by 50%, for high-performance, in-memory key-value stores.

Key Aspects

NetCache relies on a key theoretical analysis. That is, a cache only needs to store O(N log N) items to balance the load for a hash-partitioned key-value cluster with N store nodes [2]. Given N nodes and N*T total load, we don't want a single node to experience more than T load with a high probability. Given N isn't ever huge, the on-chip memory of a NetCache switch has the capacity for O(N log N) key-value pairs.

NetCache keeps caching simple by never replicating hot key-value pairs. This removes the need for cache consistency and query routing which keeps the system simple and remove overhead. To avoid loss of data in the event of a network switch crash, NetCache uses write-through to stable storage for all write requests from clients.

Cache hits don't need to visit the node that contains the key-value pair on its stable storage. Instead, as the queries are routed through a NetCache switch a hit is detected and the query is answered by the switch. This novel approach provides an incredible speedup in RTT for highly-skewed workloads.

The recent advent of highly programmable network switches brings about a new way to think about distributed systems. Switches have high I/O but limited programmability and resources. On the other hand servers have low I/O but are highly programmable and have a large amount of compute and storage power. By combining the two, we could strive for a high I/O, highly programmable system with ample storage.

Future work

Text

Limits

Text

Discussion

Text

References

[1] Berk Atikoglu, Yuehai Xu, Eitan Frachtenberg, Song Jiang, and Mike Paleczny. 2012. Workload Analysis of a Large-scale Key-value Store. In ACM SIGMETRICS.

[2] Bin Fan, Hyeontaek Lim, David G. Andersen, and Michael Kaminsky. 2011. Small Cache, Big E$ect: Provable Load Balancing for Randomly Partitioned Cluster Services. In ACM SOCC.