An Analysis of Load Imbalance in Scale-out Data Serving

Stanko Novakovic
EPFL
stanko.novakovic@epfl.ch

Alexandros Daglis
EPFL
alexandros.daglis@epfl.ch

Edouard Bugnion
EPFL
edouard.bugnion@epfl.ch

Babak Falsafi
EPFL
babak.falsafi@epfl.ch

Boris Grot
University of Edinburgh
boris.grot@ed.ac.uk

ABSTRACT

Despite the natural parallelism across lookups, performance of distributed key-value stores is often limited due to load imbalance induced by heavy skew in the popularity distribution of the dataset. To avoid violating service level objectives expressed in terms of tail latency, systems tend to keep server utilization low and organize the data in micro-shards, which in turn provides units of migration and replication for the purpose of load balancing. These techniques reduce the skew, but incur additional monitoring, data replication and consistency maintenance overheads. This work shows that the trend towards extreme scale-out will further exacerbate the skew-induced load imbalance, and hence the overhead of migration and replication.

CCS Concepts

•**Computer systems organization** → *Distributed architectures;*

Keywords

Load imbalance; Replication

1. INTRODUCTION

A significant portion of contemporary web-scale applications is latency-sensitive. However, designing a datacenter-scale system that guarantees low latency for the majority of user requests is notoriously challenging. Such latency-critical systems are often powered by in-memory distributed key-value stores (KVS), which span hundreds of servers and provide the means for locating and retrieving data fast by using consistent hashing. The flexibility and scalability of using KVS in a scale-out environment has led to its broad use as a state-of-the-art approach for low-latency data serving applications. However, a substantial pain point of the approach is that it leads to severe inter-server load imbalance whenever the data popularity distribution is skewed.

SIGMETRICS '16 June 14-18, 2016, Antibes Juan-Les-Pins, France

ACM ISBN 978-1-4503-4266-7/16/06.

DOI: http://dx.doi.org/10.1145/2896377.2901501

To ensure good quality of service under skewed workloads, providers set strict service-level objectives (SLO), requiring the service deployment to respond to user requests within a short and bounded delay. What makes the challenge even harder is that designing for the average latency is not enough; a good service guarantees that the vast majority of the requests will meet the SLO, and thus targets a 99th or even 99.9th percentile latency of just a few milliseconds [3]. To satisfy such strict latency requirements and at the same time sufficiently utilize the available resources and deliver high throughput, providers rely on various migration and replication schemes.

In this work we first characterize the skew in data serving workloads and show that the trend towards extreme scale-out [4] will further exacerbate the load imbalance problem. We then argue that such increase in load imbalance will raise the frequency of migration and replication operations to keep the system at the desired level of utilization [6].

2. SCALE-OUT-INDUCED IMBALANCE

KVS typically handle very large collections of data items and millions or billions of user requests per second. In such a setting, skewed distributions emerge naturally, as the popularity of the data items varies greatly. Previous work has shown that popularity distributions in real-world KVS workloads follow a power-law distribution, resulting in an access frequency imbalance, commonly referred to as *skew* [1]. This skewed distribution is often modeled by the power-law Zipfian distribution [2]. A classic example of such skewed popularity distribution is a social network, where a very small subset of users is extremely popular as compared to the average user. These two distinct user categories (popular versus the rest) result in a popularity distribution with a skyrocketing peak and a long tail.

Dataset distribution across the deployment's collection of servers is typically done by grouping data items into "micro-shards", each server being responsible for hosting and serving hundreds or thousands of them from its local memory [3]. This data distribution is done by applying a hash function on the *key* of the data items, which maps each of them to a micro-shard and each micro-shard to a server. In practice, a collection of micro-shards mapping to a single server represents a single data *shard* that is served by its corresponding server. Thus, even after grouping data items together into shards, the presence of the original popularity skew is still observable as a popularity skew across shards.

We define the *shard skew* as the access ratio between the hottest server and the average. Shard skew arises in a data-

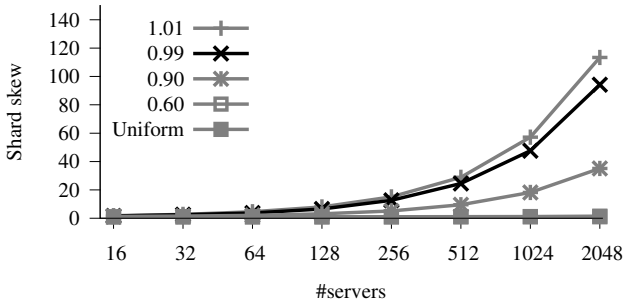

Figure 1: Shard-skew sensitivity as a function of Zipfian exponent and the number of servers.

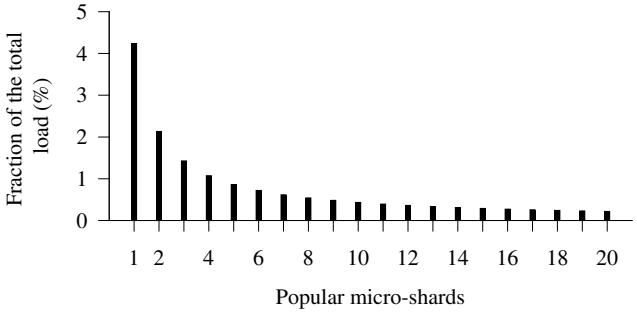

Figure 2: The 20 most popular micro-shards out of 512000 micro-shards (sorted), $\alpha = 0.99$

center deployment as a function of the exponent α of the dataset's power-law distribution. The exponent α is determined by the dataset it models. $\alpha = 0.99$ is the typical data popularity distribution used in KVS research. Some studies show that the popularity distribution skew in real-world datasets can be lower than that, but also even higher (e.g., up to $\alpha = 1.01$ [5]).

Fig. 1 shows that the shard skew becomes more dramatic as the number of servers grows. The number of servers is proportional to the dataset size and inversely proportional to the per-server DRAM capacity. For $\alpha = 0.6$, the shard skew only grows to 1.4× from 1 to 2048 servers, meaning that such a distribution would result in only 1.4× lower efficiency than a perfectly balanced distribution. At the same time, both $\alpha = 0.99$ and $\alpha = 1.01$ show unambiguously that the shard skew dramatically limits performance. For example, doubling the number of servers from 1024 to 2048 with $\alpha = 0.99$ leads to shard skew increasing near-linearly by 1.97×. Although capacity would double, the throughput would remain nearly identical.

3. DYNAMIC REPLICATION

Service providers are well aware of the problems arising from skew-induced load imbalance [3, 6]: servers that hold the most popular micro-shards can quickly become overwhelmed with user requests, degrading the whole system's performance and service quality. Data replication is a widely-used technique for dealing with such load imbalance in the datacenter. We focus on dynamic replication, where the system takes replication decisions based on the current load. Dynamic replication is never free, it requires: (i) close load monitoring; (ii) the actual replication of micro-shards and keeping the metadata up to date; (iii) maintaining consistency of a large number of micro-shard replicas.

To evaluate the impact of the trend towards extreme scale-out on the overhead of dynamic replication, we designed a model of a distributed key-value store. The model is configured with 1 billion keys, with Zipfian $\alpha = 0.99$ key popularity, hashed into 512×1000 micro-shards on the cluster. Using the information about the load associated with each of the micro-shards, we can compute the upper bound on how many replicas are required for each one of them. A suitable metric for computing this bound is the ratio between the micro-shard's load (λ_i) and the average server load ($\lambda_{avg} = \frac{1}{N}$, where N is the number of servers in the cluster).

Fig. 2 shows the 20 most popular micro-shards in our dataset. The hottest micro-shard alone accounts for 4.2% of the total traffic, which is 20× the average load of an entire server in a 512-node cluster, and 85× in a 2048-node cluster of the same aggregate capacity. For a read-only workload, one solution is to combine aggressive replication across the cluster with caching at the application tier [5, 6]. Unfortunately, neither replication nor caching is free for read-write workloads, as each copy needs to be updated on each write, regardless of the consistency model. The cost of keeping the replicas consistent increases with the size of the cluster, as more replicas are required to achieve better balance.

We conclude that the problem of serving large datasets will increase with the trend towards extreme scale-out [4], i.e., the dynamic replication of micro-shards scales unfavorably with the increase in server count. Emerging low-latency rack-scale fabrics [7] may provide a way to efficiently aggregate memory to reduce the pressure on dynamic replication.

Acknowledgments. This work has been partially funded by the *Workloads and Server Architectures for Green Datacenters* project of the Swiss National Science Foundation, the Nano-Tera *YINS* project, and the *Scale-Out NUMA* project of the Microsoft-EPFL Joint Research Center.

4. REFERENCES

[1] B. Atikoglu, Y. Xu, E. Frachtenberg, S. Jiang, and M. Paleczny. Workload analysis of a large-scale key-value store. In *SIGMETRICS '12*, 2012.

[2] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web Caching and Zipf-like Distributions: Evidence and Implications. In *INFOCOM '99*, 1999.

[3] J. Dean and L. A. Barroso. The tail at scale. *Commun. ACM*, 56(2):74–80, 2013.

[4] Facebook. Introducing "Yosemite": the first open source modular chassis for high-powered microservers, 2015.

[5] B. Fan, H. Lim, D. G. Andersen, and M. Kaminsky. Small cache, big effect: provable load balancing for randomly partitioned cluster services. In *SOCC '11*, 2011.

[6] Q. Huang, H. Gudmundsdottir, Y. Vigfusson, D. A. Freedman, K. Birman, and R. van Renesse. Characterizing Load Imbalance in Real-World Networked Caches. In *HOTNETS-XIII*, 2014.

[7] S. Novakovic, A. Daglis, E. Bugnion, B. Falsafi, and B. Grot. Scale-out NUMA. In *ASPLOS-XIX*, 2014.