CAS CS 561: Data Systems Architectures
Data-intensive Systems and Computing Lab
Department of Computer Science
College of Arts and Sciences, Boston University
http://bu-disc.github.io/CS561/

CS561 Spring 2022 - Research Project

Title: Exploring Data Placement Strategies in Distributed Databases

Background: Distributed databases have exploded in popularity due to the rise in large-scale data analytics. One question that is common amongst all distributed data systems is: where do I store each piece or unit of data? Common strategies include round robin or uniform hashing across nodes to evenly distribute the load across a network. If the database uses a relational model, strategies may try to optimally pair up tables to prevent joins across a network. Each of these strategies do not include any assumptions about the workload [1]. Rather they try to minimize the cost of any expensive overhead during operation.

In this particular project, we will focus on distributed object and key-value databases. We choose these data models as the workloads are easier to analyze since they are composed of simple operations such as get, put, update, etc.

Objective: With modern day logging and close coupling of the application to database, we can potentially create a better strategy for placing or partitioning data in distributed data systems. Before diving into exploring different methodologies, we first want to analyze the short comings on current strategies. Then we can see how information or assumptions about the incoming workload can help inform a database on where to place its data.

(a) Implement common placement strategies on a particular distributed database (Redis, MongoDB, etc).
(b) Create a pipeline to apply artificially created workloads (i.e. skewed workloads or uniform). You may need to define how we categorize workloads.
(c) Compare their performance and understand their short comings. (Where would uniform data placement hurt load balancing and by how much).
(d) Design an algorithm that, given a set of of past workloads, finds an "optimal way" to place data. Note that, optimal is not always just latency response, but can be for minimizing replication cost, network traffic, etc.
(e) Benchmark this algorithm against common strategies.

Responsible mentor*: Andy Huynh*

References

[1] Mazumdar, Somnath, Daniel Seybold, Kyriakos Kritikos, and Yiannis Verginadis. 2019. "A Survey on Data Storage and Placement Methodologies for Cloud-Big Data Ecosystem." *Journal of Big Data* 6 (1): 15.