An Empirical Evaluation of Distributed Key/Value Storage Systems

Group No. 11
Harsh Vora
Nishant Pathare
Sathyaveer Karmarkar

CS 550 - Advanced Operating Systems (Fall 2019)
Prof. Ioan Raicu

Motivation:

1. There are various disadvantages of using centralized databases:
   - Single point of failure.
   - Bottleneck on IO or CPU
   - Not flexible enough for high scale computing.
2. Relational Databases are not flexible.
3. Key-value stores can easily be used by other applications (e.g. ZHT).

Proposed Work:

- Compare throughput and latency performance of three NoSQL systems on insert, lookup and delete operations on:
  - Cassandra
  - MongoDB
  - Redis
- Deploy each system over 8 m1.medium instances.
- Scale the nodes executing concurrent 100K requests from 1 to 8.
- Compute throughput in Ops/sec and latency in ms.
- Visualize the results to know which system provides the best throughput, the lowest latency.

1. Cassandra:

Apache Cassandra is an open-source, distributed, horizontally scalable, highly available, fault tolerant and wide column NoSql database. It is written in Java and is a column family store database. All nodes in cassandra cluster are peers and there is no master-slave paradigm in Cassandra. This makes cassandra highly available, fault tolerant and no-single point of failure. Cassandra clusters can be scaled horizontally and can be distributed to multiple data centers.

Writes are very fast in cassandra, since it doesn't search for something and then write it. The data is first written to the commit log and then the reflection of this data to the table is taken care by cassandra algorithm. In cassandra, model your data model around queries, i.e first determine the application queries and then data model it. Cassandra creates sub-directories for the tables within each keyspace directory. This allows the capability to move a table to faster media like SSD's for better performance.

2. MongoDB:

MongoDB is an open-source, cross-platform, document-oriented, highly available, scalable, and flexible NoSql database written in C++. It works on collections and documents and provides high availability through replica sets.

MongoDB uses JSON like documents that can have variety of structures. Since it's schema less, you don't need to create document structure before creating documents. MongoDB uses MongoDB QL

(Query Language) for accessing the data stored in MongoDB. MongoDB has very powerful aggregate functions and an expressive aggregate framework.

3. Redis (Remote Dictionary Server):

The goal of Redis Virtual Memory (VM) is to swap infrequently accessed data from RAM to disk, without drastically changing the performance characteristics of the database while supporting data sets that are larger than main memory. Redis (Remote Dictionary Service) is an open-source and scalable data store, which can be used as a database, cache, and also as a message broker. Its written in ANSI C. Redis is an in-memory data store that can persist its state to disk, which can recover its state even after restart of Redis nodes. It's in-memory storage makes it super-fast. Redis architecture is based on BASE approach (Basically Available, Soft-state and Eventually consistent).

4. ZHT:

ZHT aims to be a building block for future distributed systems, with the goal of delivering excellent availability, fault tolerance, high throughput, scalability, persistence, and low latencies. ZHT has several important features making it a better candidate than other distributed hash tables and key-value stores, such as being light-weight, dynamically allowing nodes join and leave, fault tolerant through replication and by handling failures gracefully and efficiently propagating events throughout the system, a customizable consistent hashing function, supporting persistence for better recoverability in case of faults, scalable, and supporting unconventional operations such as append (providing lock-free concurrent key/value modifications) in addition to insert/lookup/remove.

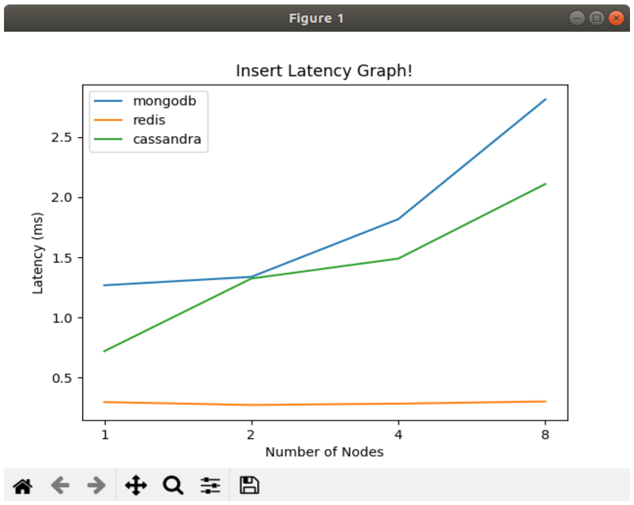
Storage System	Cassandra	MongoDB	Redis	ZHT
Primary database model	Wide column store	Document store	Key-value store	Event-driven model
Implementation Language	Java	C++	C	C++
Supported Languages	C++, Go, java, php, python.	Python, php, ruby, java, Go	R, C , C#, java , python.	C++
Replication Methods	Selectable replication factor	Master-slave replication	Master-slave replication , Multi-master replication	Asynchronous replication

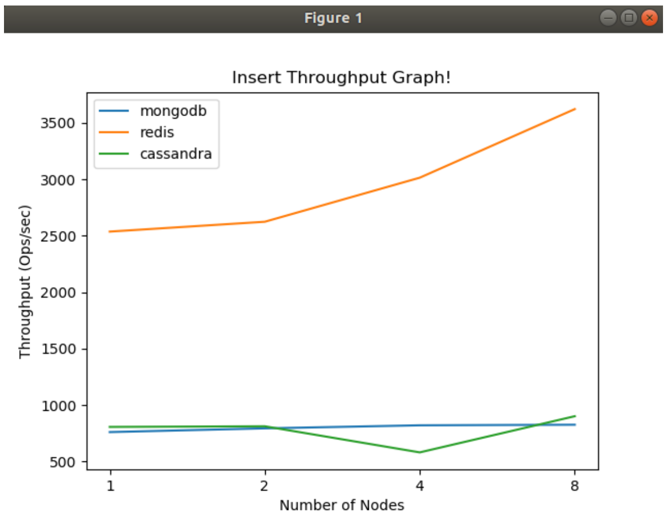
Transaction concepts	No	Multi-document ACID Transactions with snapshot isolation	Optimistic locking, atomic execution of commands blocks and scripts	No
In-memory capabilities	No	Yes	Yes	Yes

Evaluation:

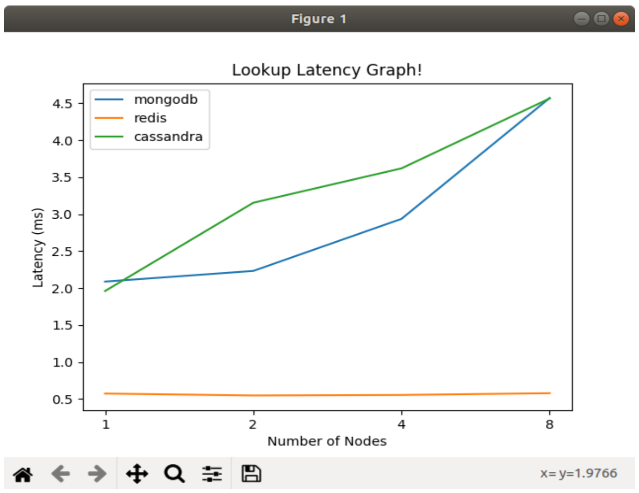
We have compared the performance (via throughput and latency) of three No SQL database systems over three operations: insert, lookup and delete.

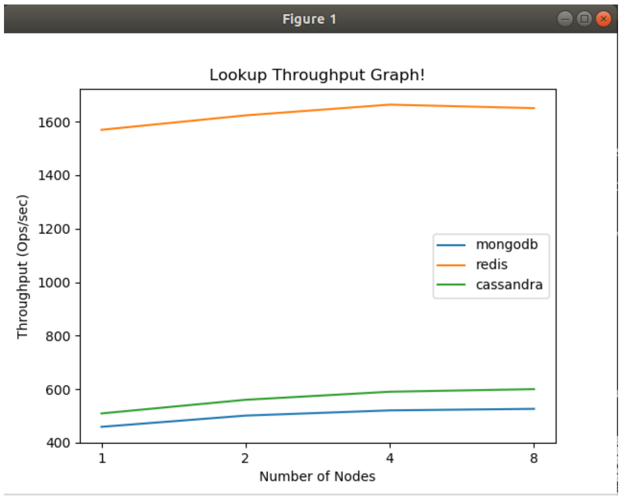
Insert Operation:



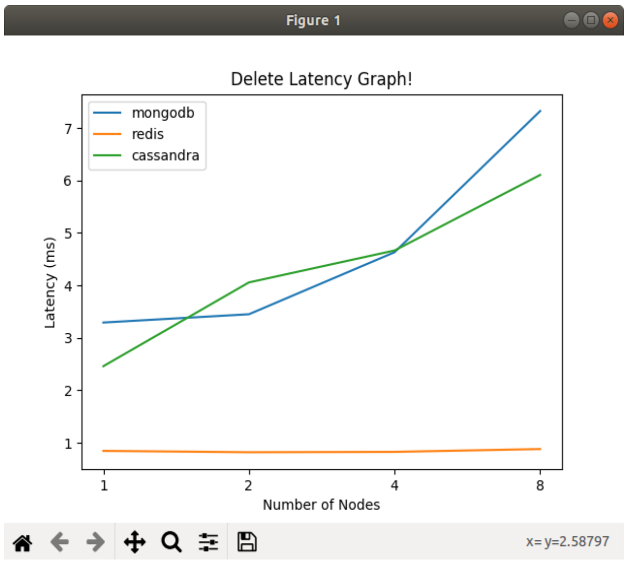


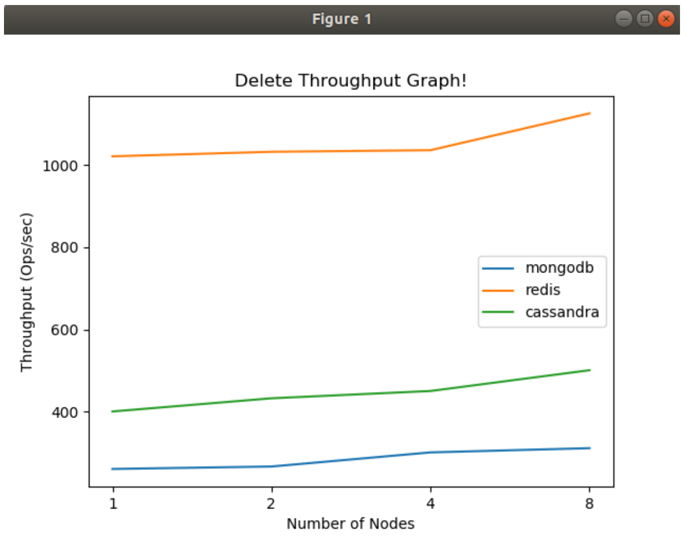
Lookup Operation:



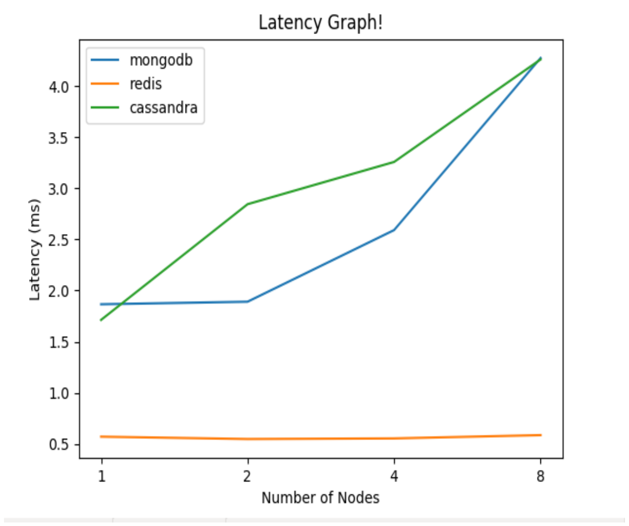


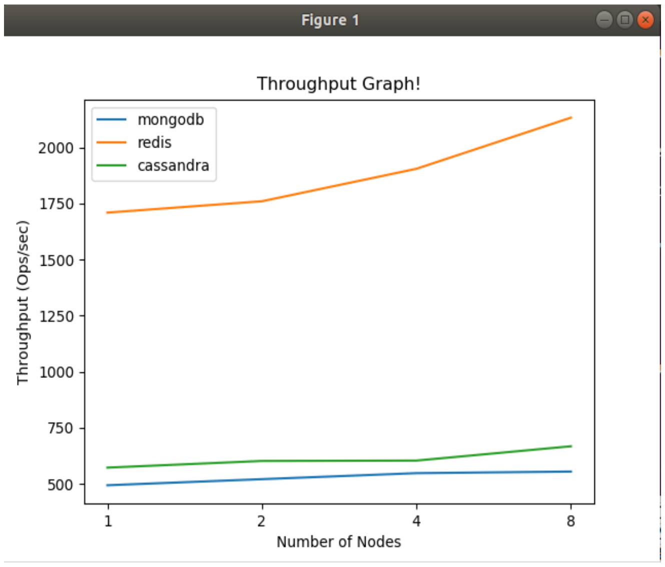
Delete Operation:





Average of 3 operations:





The reason for high number of results produced by Redis is that particular experiment was carried out on localhost. The reason why Redis gives high performance is that it has in-memory data store that can persists its state to disk, which can recover its state even after restart of Redis nodes.

Latency Chart:

System/Scale	1	2	4	8
ZHT	0.243	0.362	0.408	0.428
MongoDB	2.215	2.339	3.128	4.902
Cassandra	1.713	2.844	3.257	4.259
Redis	0.845	0.819	0.826	0.88

Throughput Chart:

System/Scale	1	2	4	8
ZHT	4117	5524	9813	18680
MongoDB	493.28	520.4	547.21	554.27
Cassandra	571.86	601.57	603.26	667.18
Redis	1708.9	1759.59	1904.4	2132.4

Conclusion:

After evaluating the three systems on chameleon, we notice that Redis is the most scalable system overall. It is observed that the operation lookup takes more time in all our system than other operations (insert and delete). In terms of latency and throughput, we conclude that Redis gives the best performance as compared to MongoDB and Cassandra.

References:

- Tonglin Li , Xiaobing Zhou , Kevin Brandstatter, Dongfang Zhao , Ke Wang , Anupam Rajendran, Zhao Zhang , Ioan Raicu. “ZHT: A Light-weight Reliable Persistent Dynamic Scalable Zero-hop Distributed Hash Table” .
- Cassandra http://cassandra.apache.org
- Redis http://redis.io
- MongoDB https://www.mongodb.org
- https://www.javacodegeeks.com/2019/02/nosql-databases-cassandra-vs-mongo-vs-redis-db-comparison.html
- https://db-engines.com/en/system/Cassandra%3BMongoDB%3BRedis