Journal of King Saud University – Computer and Information Sciences (2017) 29, 520–525

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

A performance evaluation of in-memory databases

Abdullah Talha Kabakus [a,*], Resul Kara [b]

[a] Abant Izzet Baysal University, IT Center, 14030 Bolu, Turkey
[b] Duzce University, Faculty of Engineering, Department of Computer Engineering, 81620 Duzce, Turkey

Received 1 April 2016; revised 6 May 2016; accepted 29 June 2016
Available online 2 July 2016

KEYWORDS

NoSQL databases;
In-memory databases;
Database performance

Abstract The popularity of NoSQL databases has increased due to the need of (1) processing vast amount of data faster than the relational database management systems by taking the advantage of highly scalable architecture, (2) flexible (schema-free) data structure, and, (3) low latency and high performance. Despite that memory usage is not major criteria to evaluate performance of algorithms, since these databases serve the data from memory, their memory usages are also experimented alongside the time taken to complete each operation in the paper to reveal which one uses the memory most efficiently. Currently there exists over 225 NoSQL databases that provide different features and characteristics. So it is necessary to reveal which one provides better performance for different data operations. In this paper, we experiment the widely used in-memory databases to measure their performance in terms of (1) the time taken to complete operations, and (2) how efficiently they use memory during operations. As per the results reported in this paper, there is no database that provides the best performance for all data operations. It is also proved that even though a RDMS stores its data in memory, its overall performance is worse than NoSQL databases.

1. Introduction

The key reasons behind regarding "data storage mechanism" as the hearth of enterprise software systems can be listed as: (1) it is the most major part of softwares that determines how quick an application responds a request, and (2) the loss of data is mostly unacceptable since the key business operations. Until the rise of NoSQL (Not-only SQL) databases, the relational database management systems (RDMS') were the sole and exclusive remedy. However, with the constant growth of stored data, the limitations of relational database management systems such as scalability and storage, and efficiency losing of query due to the large volumes of data, and the storage and management of larger databases become challenging (Abramova et al., 2014). At the time of writing, there exists over 225 NoSQL databases that provide different features and characteristics (Edlich, 2016). NoSQL databases are more horizontally scalable and flexible when they are compared to RDMS' (Stonebraker, 2010). When it comes to processing vast amounts of data quickly taking the advantage of schema-free data structure and distributed architecture, NoSQL databases are preferred instead of RDMS' (Bartholomew, 2010; Li and

* Corresponding author. Fax: +90 3742534526.
E-mail address: talha.kabakus@ibu.edu.tr (A.T. Kabakus).
Peer review under responsibility of King Saud University.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2016.06.007
1319-1578

Manoharan, 2013). Also, performance of RDMS' decrease with increase in size of data, which causes deadlocks and concurrency issues (Han et al., 2011). While RDMS relies on ACID (Atomicity, Consistency, Isolation, Durability) consistency model that ensures all the transactions are correctly committed and do not corrupt database, and the data are consistent, NoSQL databases are based on BASE (Basically Available, Soft-state, Eventually Consistent) consistency model in order to achieve scalability, high availability, and high performance (Bartholomew, 2010; Carro, 2014; Cook, 2009; Gajendran, 2012; Pritchett, 2008). NoSQL databases serves the data from volatile memory (i.e. random access memory – RAM) instead of non-volatile memory (i.e. hard drive) in order to increase the speed of querying since I/O (Input/Output) data access is slow (Abramova et al., 2014).

The rest of the paper is organized as follows: Section 2 describes categories of in-memory databases and their differences. Section 3 presents related works. Section 4 discusses the proposed experimental setup. Section 5 presents the experimental results and discussion. Finally, Section 6 concludes the paper.

2. NoSQL databases

NoSQL databases can be categorized into four classes according to different optimizations (Indrawan-Santiago, 2012):

- Key-value store: The data are stored as key-value pairs. This data structure is also known as "hash table" where the data are retrieved by keys. Most well-known examples of key-value stores are *Redis*[1], *Memcached*[2].
- Document store: The data are stored in collections that contain key-value pairs which encapsulate key value pairs in JSON (Javascript Object Notation) or JSON like documents (Hecht and Jablonski, 2011). Most well-known examples of document stores are *MongoDB*[3], *CouchDB*[4]. Since values are not opaque to the system, data can be queried by values as well as keys (Hecht and Jablonski, 2011).
- Column family: The data are stored as a set of rows and columns where columns are grouped according to the relationship of data (Abramova et al., 2014). Most well-known examples of document stores are *Cassandra*[5], *HBase*[6].
- Graph database: This type of databases is best used to represent data in the form of graph. The most well-known example of graph databases is *Neo4j*[7].

3. Related works

Bartholomew (Bartholomew, 2010) compares SQL and NoSQL databases with providing a brief history and the use case of each one. Tiwari provides a detailed introduction on NoSQL databases with a comparison on the basis of following features: (1) scalability, (2) transactional integrity and consistency, (3) data modeling, (4) query support, and (5) access and interface availability (Tiwari, 2011). Hecht and Jablonski present a use case oriented survey on NoSQL databases (Hecht and Jablonski, 2011). They compare NoSQL databases by their data models, query possibilities, concurrency controls, partitioning and replication opportunities.

Abramova et al. (2014) use Yahoo! Cloud Serving Benchmark (Cooper et al., 2010) in order to evaluate and compare the performance of NoSQL databases. They randomly generate 600,000 records and used them with different workloads by changing ratios of *read*, *update* and *insert* operations. The databases used in the experimental evaluation are *Redis*, *Cassandra*, *HBase*, *MongoDB*, and *OrientDB*[8]. They report that as overall the in-memory database *Redis* provides the best performance. Also, they report that column family databases *Cassandra* and *HBase* showed good update performance since they are optimized for *update* operations.

Li and Manoharan (2013) compare performances of NoSQL databases through five experiments: (1) Time to instantiate database bucket, (2) time to read values corresponding to given keys, (3) time to write key-value pairs, (4) time to delete key-value pairs, and (5) time to fetch all keys. These experiments are also tested for various data from 10 records to 100,000 records. The databases they tested are *MongoDB*, *RavenDB*[9], *CouchDB*, *Cassandra*, *Hypertable*[10], *Couchbase*[11], and *MS SQL Express*[12]. They report that *Couchbase* and *MongoDB* are the fastest two overall for *read*, *write*, and *delete* operations. They also note that *Couchbase* lacks fetching all keys from database.

Boicea et al. (2012)) compare *MongoDB* and *Oracle*[13] databases in order to compare NoSQL and SQL database performance through the three experiments: (1) Elapsed time to insert data, (2) elapsed time to update data, and (3) elapsed time to delete data. These experiments are also tested for various data from 10 records to 1,000,000 records. They report that for all operations, *MongoDB* provides better performance than *Oracle*.

Our contribution in this paper is developing our own software to measure performance of widely used in-memory databases for various experiments. Despite that memory usage is not a major criterion to evaluate performances of algorithms, since these databases serve the data from memory, it is necessary to reveal their memory usages especially when the size of data gets bigger. For this reason, unlike the related works, we also dig into the memory usages of in-memory databases alongside their performances in term of the time taken to complete different database operations.

4. Experimental setup

The in-memory databases that are experimented in this paper are listed in Table 1 with their database models and versions. There exists at least one database from each NoSQL database category (key-value store, document store, column family,

[1] http://redis.io.
[2] https://memcached.org.
[3] https://www.mongodb.org.
[4] http://couchdb.apache.org.
[5] http://cassandra.apache.org.
[6] https://hbase.apache.org.
[7] http://neo4j.com.
[8] http://orientdb.com.
[9] https://ravendb.net.
[10] http://www.hypertable.org.
[11] http://www.couchbase.com.
[12] https://www.microsoft.com/en-us/server-cloud/products/sql-server/overview.aspx.
[13] https://www.oracle.com/database/index.html.