MongoDB vs HBase Performance testing using YCSB

BY

ANURAG SINGH

X18104053

FROM

NATIONAL COLLEGE OF IRELAND

Abstract

Technological evolution in every field has led to increase of dynamic data from its various nodes, these has also led to need of up gradation of system which can stores data and can process it fast and give reliable information. Therefore a research is done in the fields of database system by calculating its various factors like throughput, average latency, operational read count, and operational write count etc. to understand the functionality of database system and how it can be employed at business level so that it can work as system for development. The paper aims at giving proper information through output in forms of graph by using all set guidelines like database selection, researching, literature survey, results comparison through graphical view and conclusion of the actual working of database on different workloads. The two database chosen for research purpose is MongoDB and HBase and workloads was chosen with operational counts 0.2 million to 0.5 million.

Introduction

Data can be in form of structured or unstructured, which affects its operation therefore two highly optimized database required for comparison which has no issue with structure of data. MongoDB is a non-relational database system. The system has the property of storing, scalability, performing operation on big data. MongoDB has the tool to penetrate in big data and query it with the help of embedded built-in function. Star Schema built in it automatically adjust according to application. Anyhow of this flexibility, MongoDB still ensures expected functionalities such as full query language and consistency. Flexible nature of MongoDB in handling data without affecting its scalability has led to multiple multi-national companies in adapting these database for developing their application.

H-base was developed to combat the issue of BigTable which was designed by Google to provide efficiency to the search engine. Therefore through set of code an open source BigTable platform was built which later known as H-Base. Under Apache, H-Base has become major source of big data handling tool. H-base is NOSQL database having all functionality of Hadoop by being built on top of Hadoop system. H-base has the function of big table to store data in distributed form. Like MongoDB it has scalability function even by merging data through automatic loading in existing storage system. It provides JAVA API for users. H-base also support block cache technology. H-base gives dynamic accessibility to application for read/write operation on Big data. H-base is independent of schema function.

Features of Database system

We are highlighting some of the features of the MongoDB and H-base which will give clear insight of on how system performed using these features and later it helped in our research part do some features of database helped them in reaching desired results.

Features of MongoDB
Aggregation: Aggregation characteristic of MongoDB enables user for processing data in batch by MapReduce() function. MapReduce() is nothing but set of codes for generating big data by parallel processing of filtering ,sorting and combining data on Hadoop.
BSON: MongoDB uses Binary JSON feature to store documents which is not supported in JSON. BJSON was helpful in our project because it's used unique identifier _id by mongod service. BSON function give MongoDB read/write throughput.
Sharding: Sharding is horizontal scaling technique by distributing large data across various server without affecting its throughput operation. Logical database is formed when all shard having different set of data are combined together. The operation performed by shard depend upon the selection by app server and configuration server.
Ad hoc Queries: MongoDB can create dynamic queries as per requirement to match Ad hoc queries by simply classifying the BSON documents.
SCHEMA-LESS: MongoDB database has high flexibility because it's independent of schema. Arranging the object in order to JSON and supplying to MongoDB saves the object. Schema less means no mapping of data required.
Capped Collection: Static size data is supported by MongoDB capped collection feature to serialize data in proper order till the defined size set by cap collection command.
Indexing: Indexing features is used by MongoDB to increase efficiency of search by using predefined index to relate query statement with fields of documents. Any documents fields can be indexed by using MongoDB.
File Storage: This feature of MongoDB helps in loading different file from various machine on single database by function called Grid file system. MongoDB also give access to the developer to make changes in content of data by set function of system.
Replication: MongoDB replication feature is used to create master & slave dataset. In which slave data set is replica derived from master dataset which helps in uniform distribution of data across various servers. Automatic failover is add on advantage of replication feature, in which secondary instance is promoted by default by replica function when user fail to reach primary instance in 10 second. This makes MongoDB ever ready.
MongoDB Management Service (MMS): MMS web tool gives access to the user to tracks database, hardware metrics, and machines and also provide support for automatic backup of data. Performance is displayed by MMS in web console to the user for optimizing deployment. The user is able to identify & rectify issue in given time shown to him through custom alerts message popping in window, and thus providing immunity to MongoDB Instances.

Features of H-base
MapReduce: It's in built feature in Hbase works same as MongoDB one by aggregating framework.
Automatic Failover: Same feature of Replication like MongoDB is used by HBase RegionServer. Multiple blocks is assigned to multiple data to recover data by replication feature to automatically load data on different nodes.
Atomic Read write: This feature command HBase to perform only one read/write operation at time and forcefully stopping of all process from read/write operation by default.
Sharding: Similar function like sharding in MongoDB to increase throughput operation automated division of regions into block region as soon as threshold size is achieved.
Hadoop/HDFS integration: HBase can be integrated on top of hadoop and HDFS system, to achieve pseudo state by adjusting some configuration in system and helps in communicating to both Hadoop/HDFS for distribution of data which does not happen in case of MongoDB. Integration feature of HBase provides consistency in data.
LSM-Tree: Log-Structured Merge Tree is in built features of HBase to combine small files to form linear data to reduce storage seeks.
JAVA API: These feature HBase is useful in programming for development.
Thrift Gateway and REST API: For accessing and embodying HBase through non-java programming a web service gateway feature is provided.
Real-time Processing: Hbase use block cache to support for query processing with respect to time.
JMX extension: Java Management Extension feature provided by HBase for scanning exported matrix by some automated tools.
Schema-Less: Same like MongoDB HBase use schema-less features by working only on columns in table.
Scalability: Feature of scalabilty of HBase is to scale out rather than scaling up like MongoDB. Therefore addition of nodes to cluster is required to match scalability.
Column oriented: HBase system is specifically designed for working on column base storage rather than row based by dividing each column distinctly.
HBase Shell: HBase shell provides interface tool to communicate with HBase which helps in creation of table, summation, checking, and deletion of data from table.
Snapshot: Snapshot feature is supported by HBase to get history of metadata in corrected order.

Database Architecture:

MongoDB architecture:

Database: MongoDB server provides a virtual storage system for different set of data segregating from main database.
Collection: MongoDB provides tools for collection of different documents of different fields by combining them without schema function in single database.
GUI: Graphical user interface is provided to mongo shell for viewing and editing documents.

HBase architecture:

Regions: Regions provides complete management of r/w operation by dividing table into rows from start to end.
HMaster: Hmaster provides data to regions and and also responsible for modifying table data by adding new data or deleting data.
Zookeepers: Server availability is provided by zookeepers to HBase to keep server phase on ON mode in HBase shell.

Research in Transaction Management

The operation performed by user to run database by modifying data, network to database, providing storage by using set query language is known as Transaction management. MongoDB transaction starts on documents with atomicity. Atomicity used provide read and write operation on single documents and hence all operation stops which is related to read and write. The necessity arises for management of multiple files. This is provided by replica set, which support multiple file transaction. Failure in transaction operation leads to stopping of transaction of data and therefore leads to deletion of data which were modified during transaction process until it's started again. Multi Document transaction is managed by read concern/Write concern/ Read preference. The user set read concern at starting point of transaction to let operation performed on basis of it. Read concern of transaction can be replaced by write concern depend upon MongoDB selection to perform transaction. Write concern specifies the requirement of node to developer before starting the operation. Snapshot read concern works on queries in read transaction to generate a data to user for viewing purpose even if the data is changed by developer parallel. Workloads performance is untouched by transaction management. Transaction refrains developers from any changes to be done to documents unless and until transaction is complete. There is no objection on read operation during transaction process but problem is that it can only see data not the modified one. During write operation like addition of column data custom alerts is generated. Transient error is activated and error message is shown in temporary form and permanent form. The temporary error one caused by any operation gives assurance of transaction safety and can be retried, because this gives option to user to add logical query and start transaction again, by retryable write function of MongoDB will help user in writing and adding query to transaction. Permanent error gives clear message there is no use of running again the transaction. This kind of error message helps developer in correcting the issue on time and making transaction time ready. Transaction execution time can be changed if the multiple data on machine is less. To allow transaction within a threshold time the transaction should be segregated. Fast transaction is only possible by serial order of query execution. Due to transaction read operation can be performed on endless number of documents. DDL operation cannot stop the modified transaction.

HBase has huge difference related to transaction management guidelines set by ACID compliance, rather than this it provide a standard approach set by Apache. Starting with atomicity which happens by fusion in row. While performing operation on rows by modifying data will only succeed or fails. If operation is succeeded then rows is modified and if is not able to change row data fail operation is shown. Same operation is performed on column families in rows. The fusion operation performed by API does not combine all rows it will return only successful operation rows. The only add on advantage of having API is that it puts complete row from table history. After modification only rows can be added and move to next stage. HBase provide scan option which does not have property of snapshot isolation. Scan option provides linear view of row from history, but it should satisfy following conditions like the data should be written before scanning, scan shows all combination of rows before scanning operation was performed. Visibility transaction avails user and clients see mutation of rows if operation is successful. Fusion of rows which happened in linear order by read operation will only be visible as final table, and these mutated cell is conveniently stored after completion of read operation because only convenient data is passed through read operation. The durability follows the same function as functionality which exist in atomicity. Tunability transaction offers option of changing visibility with respect to read operation. Tunability can also affect durability to remove unnecessary data with respect to time. If simple mechanism to be followed in architecture of HBase it is like region servers where data is stored it works on data tables to divide in regions. When write operation is performed on rows by region server it refrains any write operation to be performed on that rows. Then write operation is signalled to perform write-ahead-log (WAL) in HDFS and then only data is feeded in column after put operation. So even after auto failover of system the data stored in HDFS will act as backup and will be retrieved by region server and there is no loss of data.

Literature Survey

The question came into mind before starting research on MongoDB and HBase was why NOSQL over RDBMS? Answer to question in these field was given by Moniruzzaman, A. and Akhter Hossain, S. (2013). NOSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. *International Journal of Database Theory and Application*, 6(4). The Journal cleared whole picture of NOSQL database by providing the necessity to match pace of big data, and how traditional database is facing problem related to unstructured data and compared with some features of NOSQL which can help in handling Big data, then research was focussed on separate data base and comparing them on same platform to give purpose of selecting database. Search initiated on several NOSQL database to find property of all database and main features on which they can be compared. After reviewing most of journal the search end on article provided by Jain, V. and Upadhyay, A. (2017). MongoDB and NoSQL Databases. International Journal of Computer Applications, 167(10), which showed different database operational style and compared MongoDB with other NOSQL database on terms security, functionality, features, implementation method and checking performance. The paper clearly shows by graphical view how MongoDB execution time in updating 1000 records increases with respect to time compared to oracle. The second important aspect which was highlighted in paper was transferring database to MongoDB with ease by conversion of data by insertion, replication and dynamic joining of tables by simple use of query language. Security of MongoDB was vulnerable because access to database is possible on easy authentication through API codes because it use java language. Performance of MongoDB was compared with traditional storage by adding 10000 notebooks information in both database. MongoDB emerged clear winner in every aspect, it showed flexibility of MongoDB in handling data storage, providing query language automation to perform several operation in given time at faster rate.

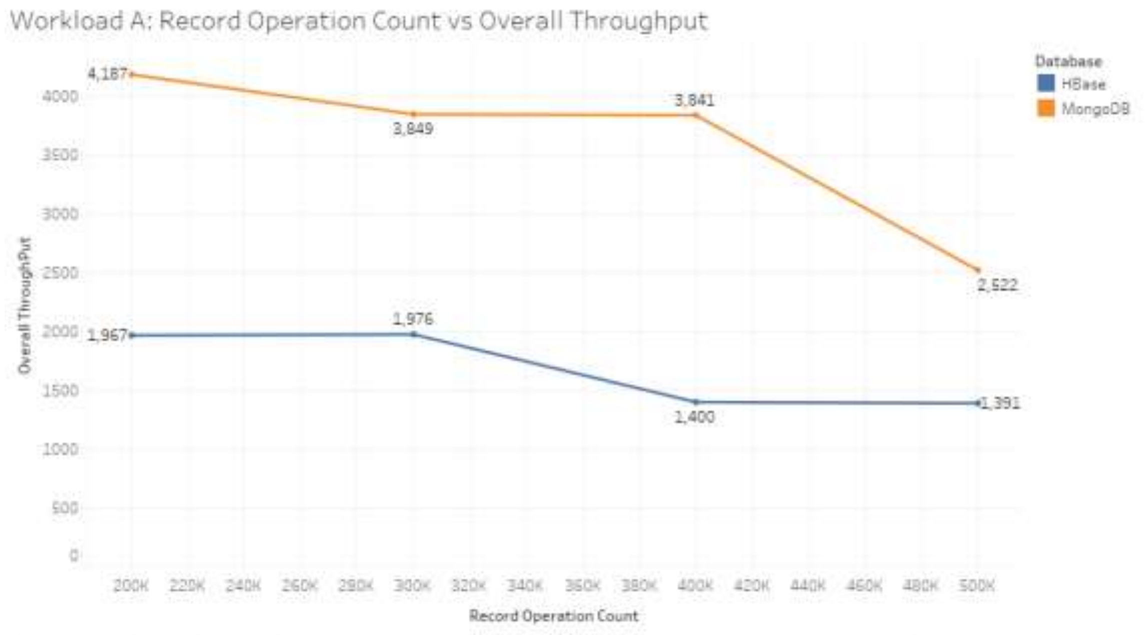
Scalability issue was solved by integrating to MongoDB features. And easy migration is possible from any SQL database to MongoDB. Security issue highlighted in paper regarding MongoDB keeps its development in cloud storage a concern.
Goyal, A. (2017). HBase - Hadoop Database highlights on working of HBase by showing its feature operation derived from various nodes. Paper showed add on advantage of HBase of integration on various machines. The architecture of HBase is explained by Region server, Hmaster, Zookeepers function how it helps on working on data tables. The main purpose for choosing this paper was it described HBase functionality in details. Data handling by HBase was done by proper segregation of data according to column. The paper showed the feature which was highlighted in my feature section. HBase strongly supports CAP theorem of providing consistency, availability and partition. The motive behind choosing HBase is to compute large volume of unstructured database. How HBase integrates with hardware while working on database was some important points to be considered. HBase provides quick access to data but transferring database from SQL is clearly issue which is need to be addressed. How HBase performed when implemented by some tech-giants was shown use case section which showed advantage of HBase like formation of big table to store data easily, exist working system which is not affected by scalability, HBase supports to CRM tools by handling numerous transaction details with ease, another advantage was providing high read/write, throughput operation consistently without performance being get affected. The later part of paper compared HBase with Cassandra another NOSQL database in terms of CAP theorem, optimization, partitioning, aggregation and development, which showed the HBase as efficient except the development part where Cassandra was more user friendly in building application. The conclusion derived from paper was HBase is strong top notch database from Apache with the help of cluster being provided from Hadoop. The key areas where HBase needs improvement is providing simple query language, easy installation, automatic failover and over dependency on hardware because it leads to declining performance.
Related work to our project by MATALLAH, H. (2017). Experimental comparative study of NoSQL databases: HBase vs MongoDB by YCSB shows the same procedure used in project for performance check for different workloads. The comparison shown between adoption of different NOSQL database on basis of their storage technique like document storage, graph storage, wide column storage, key value storage. The results obtained was adoption of document storage database like MongoDB by 55% by web survey. This helps us to understand people inclination towards documents storage system. In paper performance test was by choosing 8 workloads which have measurable value of read/ write operation at different percentage, throughput, insert operation value, update operation. Opcounts of 600000 was chosen for workloads. Only in read operation MongoDB emerged good than the HBase. Rest all part like write operation, execution time & updation time HBase emerged a winner. The conclusion can be biased based on the use of system, which will be clarified by our reports. The main purpose of reviewing these 3 journals and 1 paper was to have clear understanding of the two database on basis of comparison to suits to industrial infrastructure, and according to their features where development is required so companies can easily deploy there application. The use of NOSQL has large market in today’s world of Big Data by adapting to markets by use of its existence features is shown from all 4 literature review we have done.

Performance Test Design

Test is designed to get comparative results between two database by graphical view. Test was performed on system with configuration as follows
>Dell machine
>Window 10 Pro operating system
>8GB RAM
>I5 processor
>40 GB Storage
for comparison purpose two database 'HBase-1.4.8' version and 'MongoDB-3.2.10' version is used. Initially the two database is installed on putty through proxy server by using floating IP address given by openstack platform. And later YCSB tool of version 'YCSB 0.15.0' is installed for extracting result from the two database.
MongoDB is installed on top of Hadoop and HBase is pseudo in nature it is installed on top of HDFS. After installation using jps command all nodes are checked whether running or not. Two workloads of different operation percentage is chosen for test count workload A and workload B, their specification are
Workload A specification is divided in 50-50 in read and update operation respectively, Read operation (A) =50% and Update (A)= 50%.
Workload A specification is divided in 95-5 for read and update operation respectively, Read operation (B) =95% and Update (B)= 5%.
The two workloads are run on two database with different operational counts. 4 operational counts are chosen 200K, 300K, 400K, 500K to understand the operational nature of database with respect to time. The simple mechanism is used of running all opcounts individually on two database three time and then averaging there consideration values to obtain a graphical data. First HBase shell is activated and table is created for user to store value under column name family and then downloaded ycsb tool is made connection with HBase. Testharness function is used to run 3 times for every opcounts and data is collected. And then MongoDB shell is activated through connection of mongod to host server and same testharness connection to MongoDB is provided and rest same procedure of running 3 times for every opcounts. 4 set of data is obtained for 2 workloads and single opcounts of which two dataset is regarding loading operation into database and rest two is about running operation in database. And we need to select running operation dataset. Similarly we obtain for every single opcount and single workload 6 dataset. Therefore on adding all workload data and subtracting load data from it we get 24 set of data for each database that counts for total 48 set of data set which is later converted to 16 dataset by diving in ratio of three to get average value of every operation for different opcounts. And then these value of 16 dataset having same data from same opcounts operation and value is fed in visualization tool Tableau differently to create tabular form from which line chart data is extracted for visualization purpose to make comparison on database performance. In Tableau total 6 graphs are prepared like OPcounts vs overall Throughput, Read operation vs Read average latency, Update operation vs Update average latency for workload A and Workload B and checking performance of database.The values of average latency, read operation, update operation, throughput is calculated and compared to check performance of database.

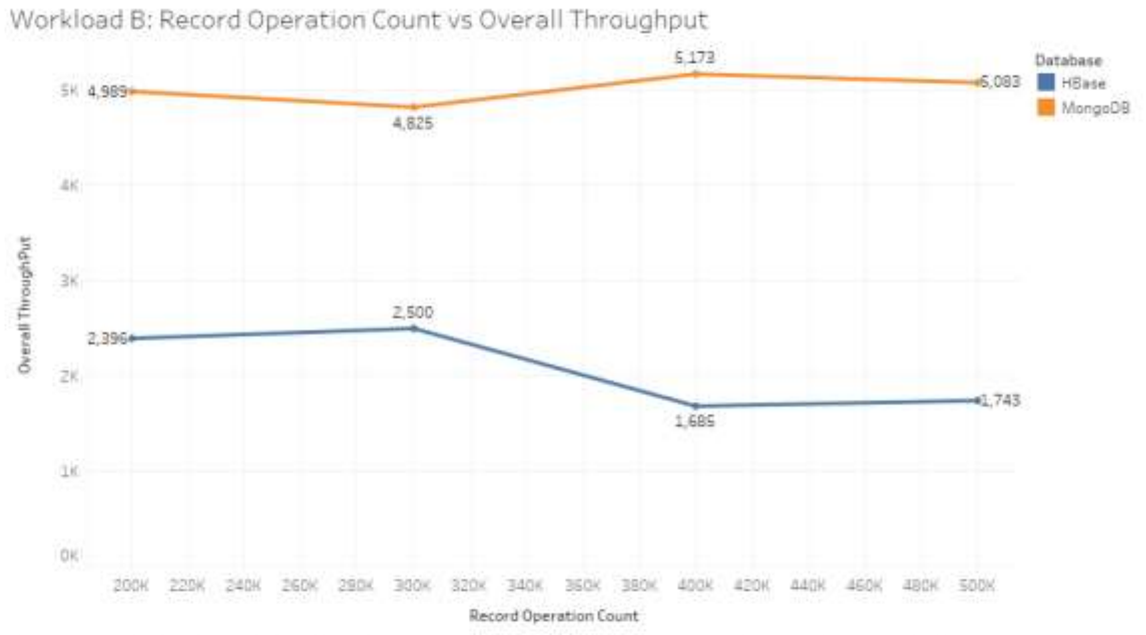
Evaluation of analysis and Results

Record Operation Count Vs Overall Throughput:
Our first analysis is on comparing overall through put against operational counts to give performance measurement of two database against two workloads.



The trend of sum of Overall ThroughPut for Record Operation Count. Color shows details about Database. The data is filtered on Workload, which keeps WA.

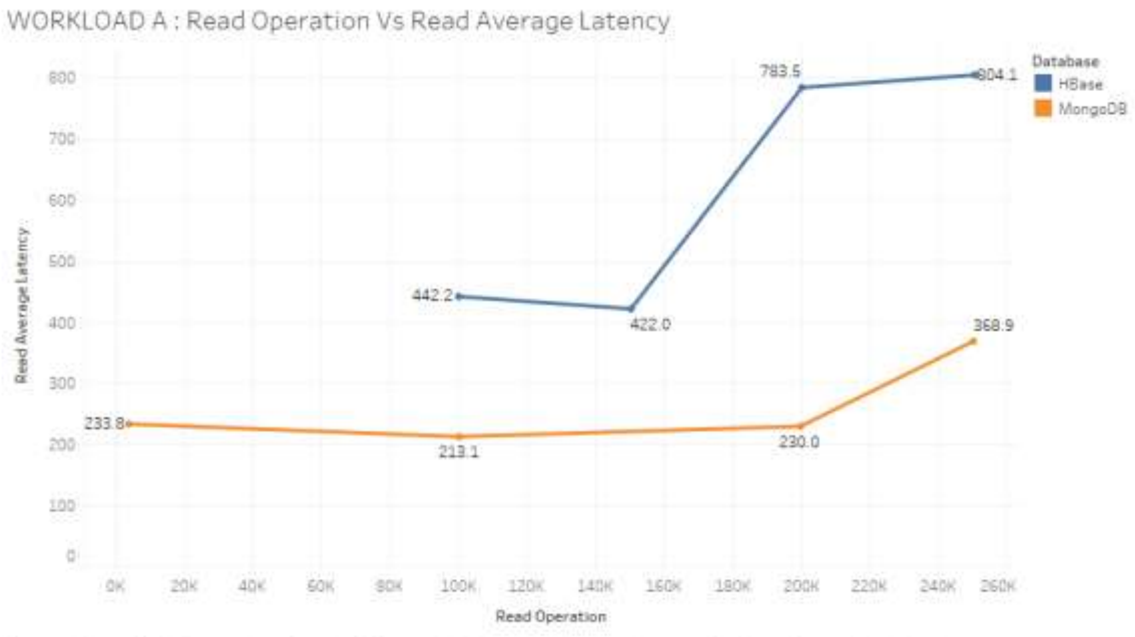
Though there is sharp decline against workload A for MongoDB & HBase, But at final opcounts we can see throughput output of MongoDB is still bigger than H-base in every case. Therefore concluded for workload A MongoDB has high throughput.



The trend of sum of Overall ThroughPut for Record Operation Count. Color shows details about Database. The data is filtered on Workload, which keeps WB.

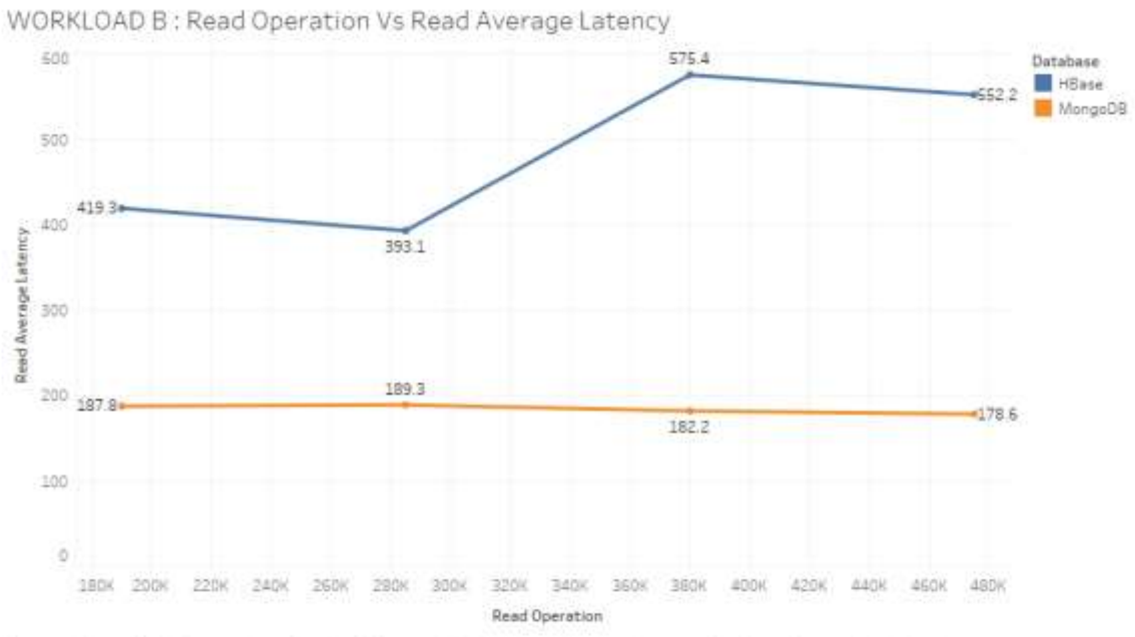
For workload B MongoDB throughput is balanced & HBase show same throughput trend as its shown for workload A. MongoDB have high throughput for workload B. Therefore we conclude that from graph MongoDB has high throughput efficiency.

Read Operation Vs Read Average Latency:



The trend of sum of Read Average Latency for Read Operation. Color shows details about Database. The data is filtered on Workload, which keeps WA.

Read average latency is high for HBase in comparison to MongoDB read average latency against read operation for workload A. Therefore it’s easy to state that against increasing number of read operation, average read latency rate of MongoDB is efficient than H-Base for workload A.

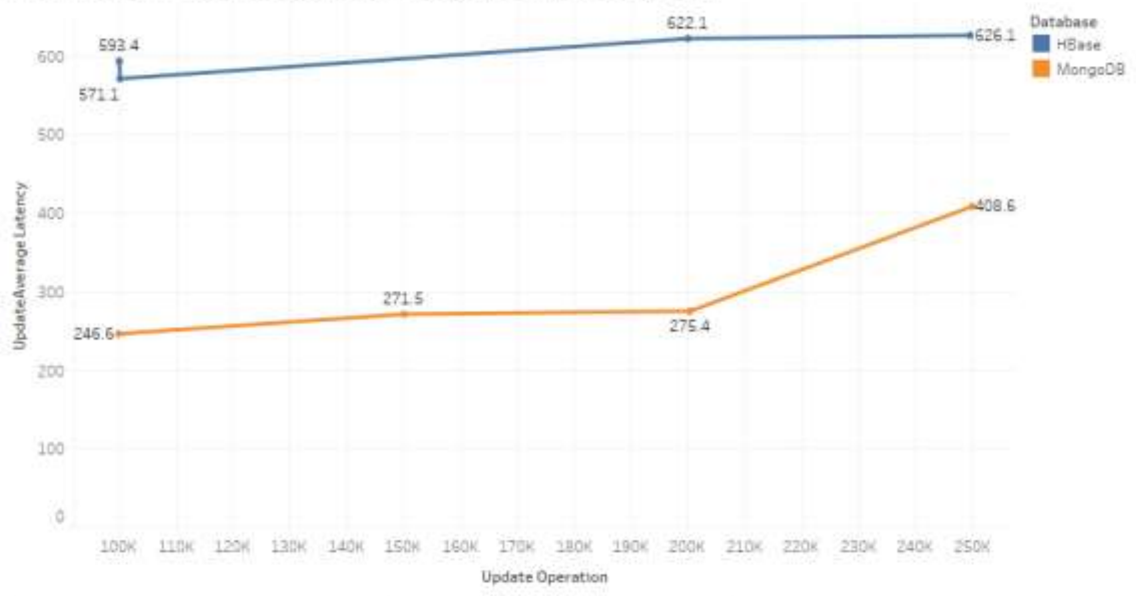


The trend of sum of Read Average Latency for Read Operation. Color shows details about Database. The data is filtered on Workload, which keeps WB.

Read average latency rate is still high for HBase compared to MongoDB against read operation for workload B. MongoDB maintains constant ratio of read average latency. MongoDB is highly efficient in terms of average read latency for both workload A & B against read operation.

Update Operation Vs Update Average Latency:

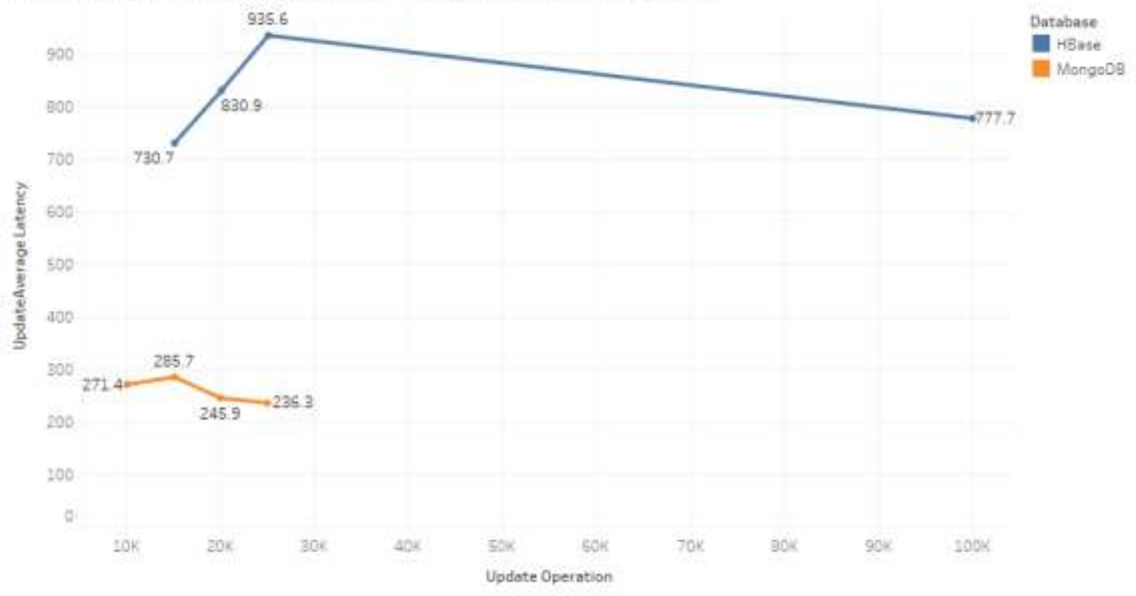
WORKLOAD A : Update Operation Vs Update Average Latency



The trend of sum of UpdateAverage Latency for Update Operation. Color shows details about Database. The data is filtered on Workload, which keeps WA.

Update average latency of MongoDB is still less than H-base against update operation for Workload A which proves MongoDB efficiency in terms of update operation for workload A. There is increase in average latency for increase update operation in which HBase increases constantly but MongoDB increases steeply.

WORKLOAD B : Update Operation Vs Update Average Latency



The trend of sum of UpdateAverage Latency for Update Operation. Color shows details about Database. The data is filtered on Workload, which keeps WB.

For workload B HBase show unusual trend for update average latency which rises and decreases but still it’s below par in comparison to MongoDB performance which constantly decreases. Thus from observation of graph we state that MongoDB is highly efficient for workload B for update average latency against update operation, because even after update operation increases average latency decreases.

Conclusion and Future work

Defining aim for developing project and clearing purpose of doing benchmarking testing, and getting some detail by literature review and developing analysis content. The findings the project has got is that MongoDB is efficient database than HBase in terms of throughput, average read latency and update average latency. MongoDB efficiency in update operation was visible from the fact replica set was clearly keeping the track of previous record and transacting that records to database. Using two workloads of different proportion of read/write operation didn’t affected the performance of MongoDB. Though at some points we have seen some balancing nature of both database in handling operation it happens only due to atomicity operation of MongoDB which has led to higher latency rate but still HBase wasn’t able to match performance of MongoDB. The feature of map reducing in MongoDB is highly efficient than HBase because we can see from graph that for read and updation operation MongoDB provided less latency rate act for both workloads. Results comparison were done on basis of important parameter, but still there can be debate around increasing operational counts to match performance, because in some case seen from the graph there might be possibility of performance being matched. Parameters set by us helps in defining at basic level MongoDB is go through option for every enteripise level, also paper reviewed in the project points in same direction. HBase have features of storing cache value which can increase its updation operation efficiency which we didn’t evaluated. The advantage of using HBase is its column oriented technique for data storing. MongoDB needs only advancement as we can see it’s trending in market plus it highly efficient in performing major operation. With the evolving unstructured data from various nodes and continuous progress in NOSQL development it has great market ahead for MongoDB if its bring addition change to its document store technique by using column-fields big table technique used by HBase. Integration with cloud storage is to look in things for future because still security issue is there for NOSQL database. MongoDB is highly efficiency completely depends upon it’s clustering with Hadoop which provides it with some of its feature and also helps in auto backup by sharding technique. Similarly Apache also provides a good support to HBase. Security is not only concern of NOSQL there is one more problem it has and it’s in complex installation process when two database are put on top of their service providers. Therefore in future with development of cloud storage NOSQL development is also related which can provide automatic convenient features to user.

References

Kumar, L., Rajawat, S. and Joshi, K. (2015). Comparative analysis of NoSQL (MongoDB) with MySQL Database. *International Journal of modern trends in engineering and research.*

Patel, H. (2017). HBase-NOSQL Database. [online] Research Gate. Available at: https://www.researchgate.net/publication/317399857_HBase_A_NoSQL_Database [Accessed 19 Dec. 2018].

Data Flair (2018). Best Features of HBase | Why HBase is Used? - DataFlair. [online] DataFlair. Available at: https://data-flair.training/blogs/features-of-hbase/ [Accessed 19 Dec. 2018].

Arora, M. (2017). What Are The Top 10 Key Features Of MongoDB? - *TutorialsJar*. [online] TutorialsJar. Available at: https://www.tutorialsjar.com/key-features-of-mongodb/ [Accessed 19 Dec. 2018].

Shriparv, S. (2014). Learning HBase. Packt.

Horowitz, E. (2017). What is MongoDB. [online] Intellipaat.com. Available at: https://intellipaat.com/blog/what-is-mongodb/ [Accessed 19 Dec. 2018].

MongoDB (2018). Transactions — MongoDB Manual. [online] MongoDB. Available at: https://docs.mongodb.com/master/core/transactions/?_ga=2.263880395.615548714.1545160200-643914588.1545061663 [Accessed 19 Dec. 2018].

Keep, M. and Cabral, A. (2018). MongoDB Multi-Document ACID Transactions are GA. [online] MongoDB. Available at: https://www.mongodb.com/blog/post/mongodb-multi-document-acid-transactions-general-availability [Accessed 19 Dec. 2018].

Moniruzzaman, A. and Akhter Hossain, S. (2013). NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison. International Journal of Database Theory and Application, 6(4).

Jain, V. and Upadhyay, A. (2017). MongoDB and NoSQL Databases. International Journal of Computer Applications, 167(10)

Goyal, A. (2017). HBase - Hadoop Database.

Matallah, H. (2017). Experimental comparative study of NoSQL databases: HBase vs MongoDB by YCSB