ISSN: 0976-3104
SPECIAL ISSUE: Emerging Technologies in Networking and Security (ETNS)
Sasikala

ARTICLE OPEN ACCESS

RESEARCH BASED LITERATURE SURVEY AND ANALYSIS ON VARIOUS SHARDING TECHNIQUES

Sasikala
School of Computer Science and Engineering, VIT University, Vellore, INDIA

ABSTRACT

Aims: *Bigdata applications with large data set and complex structure can challenge the capacity of single server. Moreover the Bigdata applications with high query rate overload the capacity of a single server. Hence to address the issues, database servers follows two techniques :* ***Materials and Method****: Vertical scaling and Horizontal scaling. Increasing the capacity of the server(vertical scaling) leads to more expensive and the problem of single node failure. Hence the Enterprises supports partition(sharding) and distribution of database into multiple servers called Horizontal scaling. Partitioning the data with multiple backups(replication) make the system scalable and fault tolerant.* ***Results:****The paper investigates various sharding and replication schemes and analyze the performance parameters and issues. The paper identified various research issues from the literature survey on partitioning and sharding techniques such as dynamic sharding, initial partitioning with an analysis of load balancing, identifying novel domain based sharding schemes and keys, hybrid solutions like MongoDB-Hadoop connector, sharding techniques on complex queries and In-memory computing techniques.* ***Conclusions:****The paper assists the researchers to move forward to find solutions on setting up scalable and reliable shards in data centres.*

Published on: 28th– August-2016

KEY WORDS

Bigdata, NoSQL, Partitioning, Sharding, Replication, Sharding Key

INTRODUCTION

Infection is a dynamic process involving invasion of body tissues by pathogenic micro-organisms and their toxins. Nosocomial/ hospital/ acquired infections are those which are not present or incubated before admission of patient to the hospital but obtained during the patient's stay in hospital. Lab coats, nurses' uniforms and other hospital garments, materials and articles may play an important part in transmitting pathogenic bacteria in a hospital setting .The hands of healthcare personnel are most commonly implicated in transmitting the pathogens [1]. Various nosocomial pathogens, such as methicillin-sensitive *Staphylococcus aureus* (MSSA), methicillin-resistant *Staphylococcus aureus* (MRSA), vancomycin-resistant *Enterococci* (VRE) and gram negative organisms is well documented [2]. Specifically in the area of dentistry, health care professionals are routinely exposed to potentially pathogenic microorganisms which are present in the surrounding environment. Most of them originate from the mouths of patients [3]. Contamination may occur from instruments through contamination vectors. These contaminated object infections may be transferred from patient to patient or from patient to professionals [4]. Methicillin resisitant *Staphylococcus aureus* which is the most pathogenic microorganism, comes in contact with health care professionals via direct hand contact with contaminated body fluids, devices, items or environmental surfaces [5].

There are very few studies regarding the wearing and laundering of lab coats in hospitals and medical practice. This study highlights the role of lab coats acting as vector for transmitting health care infections to the patients and the common areas where contamination occurs.

Internet of Things(IoT), Enterprise data, Bio medical, scientific and Web applications data are growing every day and also the structure of data are getting more complex. The way the applications storing Databases are varying by number of data in each record, data type of each field, complex format and hence structure of Database plays significant role. Generally the term Bigdata describes massive amount of data. In addition, the term Bigdata [1] defines unstructured nature of data as *3v* model : Volume, Velocity and Variety [2]. Here Volume refers size of data, Velocity refers timeliness and Variety refers unstructured nature of database. The massive amount of data creates more opportunities in business, social network and research world through data analytics, data mining algorithms

THE IIOAB JOURNAL ISSN: 0976-3104

and data storage. Various technologies are used for handling Bigdata. The four different phases of handling Bigdata are : Data generation, Data acquisition, Data storage and Data analysis.

Data Generation : Internet, Web applications, IoT, Business, Research and Scientific applications are various sources of generating massive amount data. Not only the data size is big, the data structure are semi structured and unstructured mostly.

Data acquisition : Data acquired from various sources such as sensors, social network and packet capturing technologies called acquisition. The data are stored locally in centralized server before transfer and data are applied to pre-processing technologies to remove unwanted and noisy data. These three steps are under Data acquisition.

Data Storage : Various storage architectures are used for storing the data permanently and future use. This storage technologies assist the system to improve performance of Bigdata management like scalability, reliability, availability, consistency, query performance and fault tolerance. The technologies used for handling database are NoSQL(No SQL) data store[3], Partitioning(Sharding) & Replication, distributed data nodes and MapReduce.

NoSQL Datastore : Relational Database management system is popular Database technology for decades. Relational databases supports ACID (Atomicity, Consistency, Isolation, Durability) properties for data management systems. Now Bigdata systems led to creation of new technologies. One such technology is NoSQL databases. NoSQL technologies have been popular from 2009 for non relational databases. Various NoSQL Databases evolved such as MongoDB, Cassandra, HBase, CouchDB etc. Also the data models key-value data store, Column, document oriented data store, Graph databases are used for database storage.

Partitioning(Sharding) & Replication : As database size increases, single server may not be sufficient to store entire data and rises an issue of centralized failure. Hence entire database is partition(shard) into chunks and distributed into multiple nodes. Failover mechanism is achieved by replication and replication factor is normally fixed from 3. The partitioning and replication achieves high availability, scalability, reliability and faster query execution time. Sharding key is used to assist data partitioning but sharding key cannot be fixed and common for all the applications. The objective of this research review paper is to discuss various database partitioning & replication strategies, sharding keys to partition the data and performance analysis of various schemes. Section 2 describes survey papers on Bigdata handling mechanisms and NoSQL DB stores. Section 3 describes various sharding schemes, sharding keys and analysis of performance parameters. Section 4 gives conclusion and future research directions on sharding.

EXISTING SURVEY ON BIG DATA

From 2009, various NoSQL and Bigdata handling technologies are developed and issues of bigdata management are solved effectively. The existing survey papers describes general study on Bigdata, Bigdata management schemes and NoSQL DB technologies.

Survey of Large-Scale Data Management Systems for Big Data Applications

Lengdong Wu, Liyan Yuan et al have done detailed survey on large scale data management system for Bigdat, describe Data management model, consistency model and architecture and analyse critical aspects and scalability limitations[4]. The data management systems are divided into two categories : Relational and Non Relational. The relational data model are traditional data model to support database of type structured. The non relational database model supports Big data with semi and unstructured databases. For this research review we have taken only non relational data part of this survey paper[4]. Non relational data model representatives are Key-value data store, Hadoop and Big Table. The Data model are classified as Conceptual and Physical data model. How the data is stored in database is called as Physical data model and how schema is used to represent the structure of database is called Conceptual data model. The data model is classified further as Structured, Semi structured and unstructured in conceptual model and Row oriented, Column oriented and Hybrid oriented in physical model. Consistency model is classified based on two properties: ACID and BASE. Moreover my research review required how big data is handled by contemporary method of Database storage architecture. Hence this survey paper concentrated the technologies related to storage system architecture of Big Data management. The System architecture of Big data management are classified as,

Symmetric Multi-Processing (SMP) on Shared-Memory Architecture : It consists of same type(homogenious) and tightly coupled pooled processors. The data will be stored on shared memory. When the database size increases, complex nature of this architecture is not suited to scale well. The paper concludes that SMP on shared memory has limited scalability for Big data management.

THE IIOAB JOURNAL
ISSN: 0976-3104

Massively Parallel Processing (MPP) on Shared disk Architecture : The program will be executed parallely on SMP cluster of nodes with Shared disk is scalable. But the system is not fault tolerant due the centralized behaviour of shared disk.

Sharding on Shared-Nothing Architecture : This is the technique widely used by large scale big data management systems. The data will be distributed among number of nodes and process on shards will be parallel. The nodes can share the routing data using centralized server or all the nodes are treated equally called decentralized topology[5-11]**.**

Mapreduce/Staged Event-Driven Architecture(SEDA): Some systems use MapReduce and SEDA as hybrid solution[12].

Finally this survey paper[4] concluded that MapReduce/SEDA architecture, unstructured data model with strong consistency and BASE properties are well suited for handling Bigdata management systems.

Choosing the Right NoSQL Database for the Job: a Quality Attribute Evaluation

Ricardo Lourenço1, Bruno Cabral et al have done survey on NoSQL Technologies and analyse quality attributes which helps to map NoSQL technology to particular usecase or job[13]. The paper summarized the following points :

- A survey of the literature on NoSQL Technologies
- Future research directions on NoSQL technologies with respect to software quality attributes
- Identifying the NoSQL technology to particular Job

The authors have done detailed study on NoSQL databases Aerospike, Cassandra, Couchbase, CouchDB, MongoDB, Voldemort and HBase and analysed quality attributes on these technologies. For this research survey, only Partitioning scheme and Native partitioning are taken into consideration

Aerospike : A key-value data store with Proprietary based partitioning scheme and it basically supports native partitioning. It performs auto partitioning and replication on several layers[14].

Cassandra : A Column oriented data store with Consistent Hashing partitioning scheme and it basically supports native partitioning. It supports different types of partitioning schemes and auto replication[15,16].

Couchbase : A document oriented data store with Consistent Hashing partitioning scheme and it basically supports native partitioning. It allows partitioning and performs inter and intra cluster sharding schemes[14].

CouchDB : A document oriented data store with Consistent Hashing partitioning scheme and it does not support native partitioning. It does not support sharding and allows master-master or master-slave replication[14].

MongoDB : A document oriented data store with consistent hashing partitioning scheme and it basically supports native partitioning. It supports both partitioning and replication[14,15].

Voldemort : A key-value data store with consistent hashing partitioning scheme and it basically supports native partitioning. It supports replication and partition. Also it can add and remove nodes dynamically for partitioning[14].

HBase : A Column oriented data store with Range based partitioning scheme and it basically supports native partitioning

Authors have taken software quality parameters : Availability, durability, maintenance, consistency, performance, reliability, robustness and scalability. Considering above parameters, the NoSQL data store MongoDB, Cassandra and Aerospike strongly supports 4 parameters. Next Couchbase supports 3 parameters and other data stores supports less than three. With the above parameters and theoretical study on different type of NoSQL data store Cassanda, Aerospike and MongoDB supports most of the important parameters. This is listed in Table 1. Also authors showed research direction as, this work can be extended with implementation of a particular usecase on all the above datastores directs the Software Architects to choose right kind of data store.

THE IIOAB JOURNAL
ISSN: 0976-3104

Table 1 Analysis of Bigdata Management Survey Papers

Survey Paper Title	Survey on	Nature of the paper	Parameters	Conclusion	Future scope
Survey of Large-Scale Data Management Systems for Big Data Applications	Large scale data management systems	Study	Storage architecture, data model, Consistency	MapReduce/SEDA architecture, unstructured data model with strong consistency and BASE properties are well suited for handling Big data management systems	Implementation for detailed analysis
Choosing the Right NoSQL Database for the Job: a Quality Attribute Evaluation	Aerospike, Cassandra, Couchbase, CouchDB, MongoDB, Voldemort and HBase	Study	Availability, durability, maintenance, consistency, performance, reliability, robustness and scalability	The NoSQL data store MongoDB, Cassandra and Aerospike strongly supports 4 parameters. Next Couchbase supports 3 parameters and other data stores supports less than three.	Implementation of usecase with all DBs to perform performance analysis

SURVEY ON SHARING SCHEME

Sharding schemes performance is depends on sharding key, number of shards, load balance and distribution of related information. The sharding key plays an important role to fix number of shards, load of each shard and distribution behaviour. To partition the data, shard key need to be selected effectively. In addition, to increase the query execution performance and improve migration cost, shard key must be fixed and static. The existing schemes Range and Hash based sharding are not well suited for all the applications and node balancing. Hence various sharding schemes are developed and analyzed its performance with existing schemes.

Knowledge Driven Query Sharding

.

Adam Krasuski and Marcin Szczuka proposed how the knowledge of data structure is used to perform sharding and proved that how sharding improves performance of data analytics on large database[17]. The proposed work provides the solution to search the repositories of scientific information using semantic content. The system is called SONCA(Search based ONtologies and Compound Analytics). The queries taken by the authors involve join and GROUP By operations on large databases and the resultant database size exceeds RAM allocation. Here Sharding is used for decomposing complex queries into small queries(Query Sharding). The smaller queries are executed concurrently and independently in a multicore processor or multiple machines in a networked environment. The Explicit Semantic Analysis(ESA) method assists to find semantic relationship between documents and the knowledge base MeSH. The MeSH is a knowledgebase consists of vocabulary for the purpose of indexing journal articles and book in the life sciences. The experiments are conducted with and without sharding. Three major database technologies used for experiments : Infobright(column Oriented), PostgeSQL(row oriented) and MongoDB(column oriented). The results showed that query sharding in the SONCA system utilize the computing resources optimally and execution time is considerably less than traditional(without sharding) technique.

Clustering-based Fragmentation and Data Replication for Flexible Query Answering in Distributed Databases

Flexible query assist the system to find related information if query cannot be answered exactly. Lena Wiese proposed the clustering based fragmentation and replication for finding related information in a sharded or replicated databases if exact information is not retrieved[18]. The paper proposed clustering based fragmentation and derived fragmentation to distribute the database into cluster of nodes. Here fragmentation means divides the database into number of fragments and fragments are assigned into servers In addition the paper suggested query rewriting and redirecting to decompose and direct query into multiple servers. The main focus of this paper is how sharding and replication improves performance. Hence we focus more on fragmentation and replication part of this paper.

Bin Packing Problem : Objects of different volumes must be stored in a finite numbers of bins and provides a way to minimize number of bin used.

Replication : To achieve availability, reliability and fault tolerance, fragmented data should be replicated into several servers. An extension of Bin Packing Problem(BPP), BPP with conflict proposed two constraints: Conflict objects(fragments) should not be placed in same server and fragmented part of database stored in one sever will be replicated in *m-1* other servers, where *m* is a replication factor.

Clustering Fragmentation: The fragments are clustered and number of cluster are formed to support flexible query system. Lena Wiese proposed the following rules as Definition[18] :

- Horizontal fragmentation
- Clustering
- Threshold
- Completeness

THE IIOAB JOURNAL
ISSN: 0976-3104

- Re-constructability
- Non-redundancy

Derived Fragmentation: In fragmentation, data accessing together should also be in same cluster or server will make the system faster. This is achieved by derived fragmentation technique. To achieve better locality, derived fragmentation also supports redundancy in fragments. That is few fragments might be stored in more than one servers to give better performance in query answering.

Implementation is based on PostgreSQL and the UMLS and the database is MeSH taxonomy. The results shows high execution time and number of fragments, when row count is more than 1000. The authors suggested to improve scalability by the implementation of parallel clusters

Social BIMCloud: a Distributed Cloud-based BIM Platform for Object-based Lifecycle Information Exchange .

The AEC(Architecture, Engineering, Construction) industry adopted the BIM information exchanges through networks for sharing files, designs, models etc. The current file based information exchange leads problems like slow data transfer, lack of interoperability etc. Moumita Das, Jack CP Cheng et al proposed Social BIMCloud, a simple and less expensive distributed cloud based BIM platform for information exchange[19]. The existing BIM do not provide any platforms for social interaction between teams. Hence the author proposed a Social BIMCloud that integrates NoSQL database management systems, dynamic splitting and merging of BIM information, social interactions and partitioning(sharding) of BIM models. The Social BIMCloud framework divided into three layers: data capture and flow controller layer, data upload and extraction layer, and data storage layer.

- The data capture and flow controller layer obtains the input from users in four different ways : BIM files, web pages, BIM software and system commands
- The data upload and extraction layer receives data from input extracts the key information then convert it into NoSQL format
- The Data Storage layer uses partitioning data into number of shards. Partitioned data or shards stores in different nodes of cloud

The Data storage layer of BIMCloud proposed partitioning of Big BIM model using horizontal partitioning and fragments are stored in cloud nodes. This shards facilitate the parallel read and write and faster query execution rate. The horizontal partitioning makes the faster query performance. The BIMCloud also supports automatic replication and replication factor is decided by the end users. The Social BIMCloud supports

- large volume of data
- data can be added dynamically with different size
- dynamic schema

The BIMCloud supports column oriented data model. Each row consists of unique key with set of column values. The value of column can be added dynamically. The column family has two types : Regular column and Super column. The super column supports tree like data structure i.e., one column is nested with another column. Since Building elements consists of many properties and sub layers, BIM cloud supports tree like data structures. New instances can be created and deleted. For example, more number of cloud instances are needed during construction phase than maintenance phase. For demonstration, the BIMCloud is deployed and Tested using Tomcat webserver, Cassanda NoSQL database and PHP scripting language and hosted in Amazon Web Services(AWS).

Research on the Improvement of MongoDB Auto-Sharding in Cloud Environment

MongoDB Auto sharding creates shards in network environment without external intervention but most of the time data are not distributed evenly among shards. Yimeng Liu, Yizhi Wang et al proposed an algorithm which effectively balance the data among shards[20]. Data in shards are distributed evenly based on Frequency Of Data Operation(FODO). The paper focuses MongoDB Auto sharding which supports

- Auto Balancing
- Easy addition of new servers, if needed
- No single point of failure
- Automatic failover

MongoDB uses shard key for distributing the data among servers. The key assists to create the chunks and chunk will be divided again, if size is increasing beyond the limit. This is called auto sharding. The components of MongoDB sharding are

Shards : Servers which stores partitioned data
Config servers : Servers stores metadata which gives the details of which data stored in which server
Mongos : Servers receiving and directing requests from user to shards.

Auto-sharding moves the chunk among shards based on the size, but the data operation is not taken into consideration. Hence, Auto-sharding cannot achieve effective balance. The proposed algorithm takes FODO as value based on number of Insert, Delete and Update operations on particular chunk. The FODO algorithm works as follows:

Step 1: Calculates $FODO_i$ value for each chunk.

Step 2 : If number of chunks between shards is greater than 8, then balancer algorithm will start work till it reaches 2.

THE IIOAB JOURNAL
ISSN: 0976-3104

Step 3 : Migrate the chunk from one shard to another shard based on $FODO_i$ value. Move the chunk to front shard if $FODO_i$ value higher and keep lower $FODO_i$ value shard at the end.

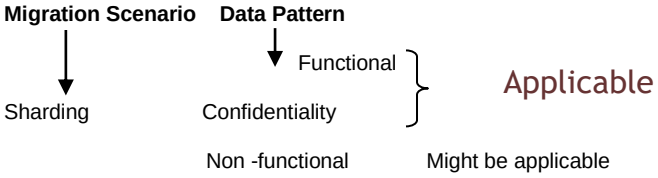
The implementation part chooses 10000000 documents(records) of random data and auto sharding cluster consists of 10 virtual machines. The results showed that performance of concurrent read and write operations is improved than traditional auto-sharding algorithm.

Migrating Application Data to the Cloud using Cloud Data Patterns

The advantage of cloud computing is to design application for cloud or after implementation, we can migrate the application to cloud for utilizing resources economically. Steve Strauch, Vasilios Andrikopoulos et al analyzed various migration scenarios and characteristics to migrate the application data to the cloud[21]. This paper mentioned the following scenarios :

- Database layer outsourcing : Complete or partial database layer outsourcing without changing the datastore.
- Using Highly scalable data store : Migrating non-highly scalable database to NoSQL or BLOB data store
- Geographical Replication : Replicating and maintaining consistency of the database layer
- Sharding : Distributes the data into cluster of nodes and distribution can be done geographically
- Cloud bursting : outsourcing database temporarily to cloud for managing resources
- Working on Data copy - Keeping complete or partial database layer on cloud
- Data synchronization - Working with database offline and synchronize with the cloud copy to maintain consistent database

In the direction of supporting application data migration to cloud, authors have done literature research focusing on cloud data patterns. The research is focused reports from various industries who already moved to the cloud for application layer data. Strauch et al., says " A Cloud Data Pattern describes a reusable and implementation technology independent solution for a challenge related to the Data Layer of an application in the Cloud for a specific context" [21]. There data patterns have been identified : functional, non functional and confidentiality. The next step is mapping of migration scenarios to cloud data patterns. Various migration scenarios and its data patterns are discussed in the paper. For example,



In the case of cloud data store selected for migration does not support sharding, to scale the database layer read and write operations, Sharding-based Router is used. It decides which part of database layer should not move into cloud store, for example business confidential data. Sharding based router direct confidential data to local server instead of public cloud. Geographical replication can also be combined with sharding when we replicate frequently accessed data. For Health Insurance Company(HIC), the client health records are stored in local server and financial transaction are stored in both local and public cloud. Moreover part of financial transaction are necessary for auditing, hence critical data filter and Pseudonymizer are used for configuration before Router based sharding operation. The authors guiding researchers to analyze various mapping of scenarios to data patterns and choose one among to implement as a single framework. The authors also concluded that the query results should not contain any client medical records and queries from the auditors must be analyzed in advance to save confidential data. This review focused more on sharding part and other mappings also discussed in the literature review[21].

Point Collection Partitioning in MongoDB Cluster

Spatial data has its own properties such as topological relationships, spatial locality and spatial resemblance. Partition and distribute the data into cluster of nodes and balance the data size on node is a critical issue. Shuai Zhang and Bolei Zhang et al proposed the following techniques for applying sharding on spatial data[22]:

Random Partitioning Strategy (RPS) - In order to shard the collections of data, shard key must be required. The shard key must be in every document so that data can be partitioned. RPS chooses the objectID(like Serial no) as the shard key. Hence partitioning is random

Space filling Curve Partitioning Strategy(SPS) - It is a mapping technique to map multi dimensional data to 1-D data. The data is divided into multiple blocks and thread is passing through all the blocks atleast one time to create a single shard. This will be repeated to create cluster of shards.

K means Partitioning Strategy (KPS) - It creates K centroids (shards) and the partitioning of data into number of shards is based on K Mean clustering formula. Distance measure between a data point and the cluster centre , is an indicator of the distance of *n* data points from their respective cluster centres. The above three algorithms are applied on 3 set of data and the average response time was measured. The results concluded that KPS provided better results than SPS and RPS techniques.

Data Management in Cloud Environments: NoSQL and NewSQL Data Stores

Katarina Grolinger1 and Wilson A Higashino et al have done detailed study on NoSQL and NewSQL data stores considering various features such as read and write requests, partitioning, replication, consistency and concurrency control[23]. In addition

THE IIOAB JOURNAL
ISSN: 0976-3104

this paper identified challenges, benchmarking criteria and diversity and inconsistency of terminologies. They have done detailed analysis of various usecases on different sets of data. This paper focusing on three scalability oriented parameters partitioning, Replication, consistency and concurrency.

Partitioning : Two techniques of horizontal partitioning techniques are discussed: range partitioning and consistent hashing. In range based partitioning, range of data on particular key will be grouped into one shard. Hence adjacent data will be stored in a shame shard. The disadvantages are unbalance shards and processing node always be focused on one or cluster of shards. The data will be represented as ring and the ring will be divided into ranges based on number of shards. The range of data will be assigned to particular shard based on hashing algorithm. Partitioning Graph database is more challenging task than other NoSQL Databases. The NewSQL databases clustrix and VoltDB use consistent hashing algorithm for partitioning. Google spanner uses spanservers for partitioning.

Replication : Replication can be done in three ways : Master-Slave, Master-Master and Master less replications. In Master-Slave, one node is assigned for write operation(Master). slave nodes are informed by Master. In Master-Master, all the nodes are updated and communicated simultaneously. In Masterless is working same as Master-slave, but all the nodes are treated equally.

An Overview of Newly Open-Source Cloud Storage Platforms

Gu, Genqiang et al provided an overview of open source cloud stores in terms of architecture, implementation technology, architecture and cloud storage[24]. They have taken MongoDB, Cassandra and Hadoop open source technologies for comparison. This paper focusing more on storage architecture for analysis which gives insight to readers to choose technologies for their applications.

Storage mechanisms of Cassandra and MongoDB are taken for reviewing.

Cassandra : Cassandra supports Column Family based data model which is more effective than traditional key-value data store. Cassandra follows Write-back cache technique for writing and updating the data into database. Before updating, the commitlog file should be submitted. Then the data will be updated in Memtable. Memtable content will be flushed into disk batch wise after some conditions are satisfied. The data are stored as Sorted String Table (SSTable). Read operation is not allowed before the write. Bloom Filter algorithm is used for searching the data by key.

MongoDB : MongoDB supports document oriented database. The row of table is called document. The storage architecture of MongoDB comprises of Config server, Shard server and Router Process. Shard server contains actual database partition and its replicas. Config server assists to identify which data stored in which shard. Route process redirect the client request to exact shards with the help of Config server.

The storage mechanism of Hadoop is Hadoop Distributed File Systems(HDFS), Abicloud is appliances repository virtual storage systems, Cassandra is structured key value and MongoDB is document oriented data store. The technology used in storage mechanism for Hadoop is multiple repilcas of data blocks, Abicloud is virtual storage factory, cassandra is column family data store and MongoDB is auto-sharding. Authors concluded that there is a perfect balance between performance and the scalability, MongoDB has been selected as storage mechanism for their own experiments.

High Dimensional Biological Data Retrieval Optimization with NoSQL Technology .

In medicine studies, microarray experiments data are stored and accessed frequently. The RDBMS technologies used for data warehouse tranSMART for accessing patient gene expressions make the query performance slow. Wang, Shicai, et al. introduced a key value data model implemented in HBase to support faster queries on large scale microarray[25]. The result analysis shows that the new data model on HBase outperforms traditional RDBMS and NoSQL Data store MongoDB.

Microarray Data model Implementation using HBase:

Determining Column Key and Row key : Transcriptomic Database is taken for implementation. PATIENT_ID and TRAIL_NAME are used as row key for two reasons: Column Family can manage single data type easily and BigTable cache will help to retrieve the data related to single patient from StoreFiles quickly.

Optimizing Row Key to Speedup Query Performance: Order of placing keys in a row is also important. Here authors recommended to place TRAIL_NAME before PATIENT_ID. The composite key with proper placement keys will assist to retrieve the identical patient easily.

Optimizing Column Key to Increase Cache Hit Ratio : Design a column key which consists of different type of data to increase hit rate. For example, GENE_SYMBOL+PROBESET_ID as column key retrieve millions of record of patients who have same type of PROBESET_ID. The theoretical performance of ideal key value data model performance is 83% higher than RDBMS model. Also the key value data model is implemented using HBase with dataset loaded in tranSMART. Various test cases are tried with HBase, MongoDB and RDBMS and the results showed that HBase retrieval rate is higher than MongoDB and RDBMS. Authors also concluded that RDBMS consumes more memory than NoSQL data store

Sharding for Literature Search via Cutting Citation Graphs

Haozhen Zhao proposed a sharding policy for search document using cutting citation and co-citation graphs[26]. Scientific literature is growing rapidly every year, hence the shards are used to partition and distribute the documents into different nodes.

THE IIOAB JOURNAL ISSN: 0976-3104

Hence the data are distributed to number of nodes. Distributed Information Retrieval(DIR) is required to enhance the performance. DIR has the following features:

- Parallel execution of the queries : Speedup the process
- Searching promising shards : instead of searching all the shards, search only promising shards gives faster response
- Small size shards : even it is unscalable, it makes effective search on specific shards

First step is to build a shards based on citation and co-citation graph partition. Secondly, effectiveness of sharding based on citation and co-citation graph cutting is experimented and results are demonstrated. Citation and Co-citation graphs are constructed based on documents are considered as nodes and edges are relationship of citation and co-citation. The subgraphs are constructed to build meaningful shards. Hence the shard contains the documents under same category makes searches are easy and fast. Here iSearch test collection and Graph clustering algorithm Graclus is used for experiments. The results showed that the cutting citation and co-citation graph effectively producing relevant information retrieval. The experimental results also showed that co-citation graph performance is better than citation graph.

Morphus: Supporting Online Reconfigurations in Sharded NoSQL Systems

Mainak Ghosh and Wenting Wang et al proposed a system called Morphus proposed the features and incorporated into MongoDB[27]:

- Online reconfiguration
- allows read and write operations concurrently
- Flexible data placement
- Master-slave replication
- range partitioning

Here reconfiguration phase is significant which allows read and write operations concurrently. The reconfiguration allows the shard key changes, splitting the chunks and altering the chunk size dynamically. The proposed Morphus : Online reconfiguration can be done in five sequential steps:

- Create partition with shard key
- Isolate one secondary server from replica set
- Operations can be done on primary server
- Recovery phase - reconfiguration can be done on isolated secondary server
- Finally secondary server copies everything into primary server and same will be repeated in all secondary servers.

The purpose of reconfiguration is online resharding operation which achieves load balance the new shards to be placed to reduce network traffic. This paper proposed new techniques : Greedy Assignment and Bipartite matching.

Greedy Assignment : A centralized server that runs greedy algorithm collects all the information an inform its new decisions to servers. But it creates a bottleneck by allocating more chunk at few servers.

Bipartite matching : The algorithm has more advantages than Greedy Technique: reduces read and write bottlenecks and latency and prevent allocation of too many chunks to few servers. The techniques are implemented by using pymongo interface with Amazon review as datasets. ProductID as old key and userID as new shard key. The results showed that mild degradation in read and write latency during reconfiguration phase. Morphus scales well with increasing replica set.

Research on Improvement of Dynamic Load Balancing in MongoDB .

Data partitioning on distributed nodes and migration dynamically makes the system costly. Xiaolin Wang, Haopeng Chen proposed heat based dynamic load balancing algorithm which reduce the cost of the sharding process[28]. The following are the steps of Heat based dynamic load balancing algorithm :

Exception Detection Algorithm : To identify the load of each shard, upper and lower bound are fixed for three resources : CPU, Memory and bandwidth. The monitored utilization of each resource is compared with upper and lower bound value and decision is taken whether the shard is overloaded or underloaded.

VM overloaded balancing : This step identify the hot chunk(more requests) and that chunk will be divided and then autosharding will be executed to balance the load of shard.

VM underloaded balancing : Each overloaded node will identify one pair which has status of underload. Then overloaded node will migrate data to this underloaded node.

Physical overloaded balancing : once the physical node is identified as overloaded, that primary server status will be changed as secondary. One secondary node will also be changed as primary. Hence the write request will be reduced. Also the migration step is prevented. If no node is identified to convert from secondary to primary, migration step comes into action.

Simulation work has been done for client access and virtual nodes are created with Xenserver and MongoDB is used in virtual nodes. The results showed that resource utilization is controlled and reduced migration cost.

Scalable Transactions in Cloud Data Stores

THE IIOAB JOURNAL ISSN: 0976-3104

The hash and range based partitioning schemes are easy but not good at current online transactions scenario. Swati Ahirrao1 and Rajesh Ingle proposed workload driven data partitioning for Online Transaction processing(OLTP) web applications[29]. The advantages and disadvantages of various partitioning schemes are discussed

Static Partitioning : once designed, it will not be reconfigured. lower number of migrations and more number of distributed transaction

Dynamic Partitioning : Partitioning can be done online. lower number of distributed transactions and more migration steps

Scalable workload driven partitioning : not fully static or dynamic. initial transaction logs are analyzed and partition can be done. Periodically partitions can be reconfigured. Less number of migrations and distributed transactions

The objectives of the scalable workload driven data partitioning are reduce number of distributed transactions and even load distribution in shards. The algorithm steps are as follows :

- The algorithm takes partitions and complete transaction log file. The algorithm starts with static distribution of shards.
- Genetic algorithm step mutation is applied on partitions to find all the combination of shards to form the partition.
- Calculate load distribution for that partition
- The combination association is calculated based on executed transactions and number of distributed transactions.
- Rank value will be assigned and the values will be ordered based on the rank
- The lower rank value combination is taken as a partition which reduces distributed transaction and efficient load balancing among servers

Experimental setup is done on Amazon SimpleDB, EC2 and TPC-C standard benchmark and results are analyzed on parameters throughput, response time and distributed transactions. The results showed that it provides better results than Graph and Schema based partitioning

Sharding Social Networks

Quang Duong and Sharad Goel proposed a sharding technique for social website data. Since social network website user communalities are geographically closer, tightly knit cluster of users are stored in single shard[30]. This solution solves the problem of reduced distributed access for a single query. This paper showed that Random partitioning is not suitable for social network database and network aware sharding is an NP complete problem. This paper proposed two steps to apply sharding on social network databases:

- VBLabelProp is a technique for identifying tightly group of knit nodes and place the nodes in a single shard.
- BlockShard is a greedy method for zero replication partitioning which minimizes sharding cost. The inputs of Blockshard is adjaceny matrix, Maximum sharding capacity and VBLabelProp output. The BlockShard assigns node to only a single shard. The technique used excess storage by replicating locally popular nodes in each shard.

The system is implemented on two social network websites LiveJournal and Twitter. The experiments done on with and without replication in terms of load balancing and average access on shards. The results are compared with Random partitioning, Geo, network aware sharding with METIS and VBLAbelProp. The results showed that network aware sharding with METIS and VBLabelProp outperform Random and Geo based sharding techniques. The authors concluded that knowing network architecture in sharding provides better performance

Performance Evaluation of a MongoDB and Hadoop Platform for Scientific Data Analysis

E. Dede, M. Govindaraju et al have taken scientific data from Advanced Light Source, Joint Genome Institute and Materials Project for analysis[31]. This paper evaluated the performance, fault-tolerance and scalability of using MongoDB with Hadoop framework for scientific data analysis. The MongoDB and Hadoop connector is an open source framework supported by Hadoop. The sharding facility of MongoDB is used by Hadoop instead of HDFS. But the results showed that MongoDB exhibits poor performance for parallel write operation because of global write block.

Experimental setup configured Hadoop, HDFS, MongoDB and Hadoop-MongoDB connector. Hadoop framework selected number of Maps and set Reduce as single. Each mapper selects MongoDB shard and this leads to load balancing. Java and Python scripts are used for implementation and data set as US Census records. Authors concluded that HDFS performs better than MongoDB. The MongoDB with Hadoop framework supports isolation of data nodes from process nodes that increase fault tolerance percentage. The following are the key insights :

- MongoDB with shards will have better performance than single node
- Output of the analysis could be written into HDFS provides better performance
- Serving MongoDB for storage and Hadoop for computing nodes make the system more elastic
- MongoDB with Hadoop connector provides better performance than MongoDB map-reduce
- MongoDB with Hadoop connector makes worse performance in read and write operations. It is due to design difference of HDFS and MongoDB
- MongoDB with Hadoop connector provides fault tolerant system

Using Paxos to Build a Scalable, Consistent, and Highly Available Datastore

.

THE IIOAB JOURNAL
ISSN: 0976-3104

Jun Rao, Eugene J. Shekita, et al proposed Spinnaker's Paxos a replication protocol for spinnaker experimental data store[32]. Its performance on read and write operation is better than traditional method. Generally master-slave replication is fault tolerant for single node failure and not working well for two node failure with replication factor 3. The Paxos protocol is the proven solution for the replication factor 3 or more.

Spinnaker's Architecture:

- Spinnaker apply range partitioning and row of a table is spread over cluster.
- The data in a node is replicated into *N-1* node with *N=3*. This step is similar to chained declustering [33].
- The replicated cluster is called cohort. Cohorts can overlap. For example *A-B-C* is one cohart and *B - D - E* is another cohort. Each cohort will have one leader.

- Leader election phase selects leader in case of failure. In the quorum phase , leader propose write operation and followers will accept it. After write operation, the message will be informed to followers. The followers will send acknowledgement after write is over. This is the way it maintains consistency among nodes in cohort.
- If leader fails, the follower who committed all the write operation initiated by the old leader will be selected as new leader. Even the old leader comes back, it can join as follower.

The experiment results are compared with Cassandra. The results showed that read operation latency is significantly reduced compared to Cassandra. Write operation results are 5% to 10% worse than Cassandra. This is due to wait for acknowledgement from followers in Cohort. The authors also concluded that this is little pay to achieve strong consistency.

SWORD: Scalable Workload-Aware Data Placement for Transactional Workloads

The main objectives of any database sharding on nodes are how to partition the database effectively, high availability, reducing number of distributed transactions and fault tolerance. Abdul Quamar, K. Ashwin kumar et al proposed a scalable workload aware data partitioning(SWORD) scheme for online transactions[34]. The following are the different steps applied in this paper to improve the performance of system:

- hypergraph compression technique to reduce the overheads of partitioning
- Incremental data repartitioning technique for dynamic load changes
- SWORD manages load balancing and increases availability
- fine grained quorum improves throughput and reduce the cost of distributed access with different read and write patterns

The experiments uses TPC-C bench mark for evaluations and the results showed that improved query routing time, overall end-to-end time and throughput. The results are compared with Random technique and baseline approach.

The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data

Aaron McKenna and Matthew Hanna et al proposed a Genome analysis toolkit for analyzing Next Generation DNA Sequencing(NGS) data[35]. The Genome Analysis ToolKit(GATK) architecture uses the MapReduce functional programming to design the framework. The MapReduce framework allow the researchers to write efficient NGS toolkit. Here MapReduce divides the computation into two steps : Map and Reduce. Single Nucleotide Polymorphism (SNP) discovery and genotyping are fed into Map function and chromatin immunoprecipitation is fed into Reduce step. GATK partitioning Genome, SNP and SMP format of genome into shards for Map Reduce operation. In addition, the shards are divided further by traversal engine. The authors also suggested to implement sharding with data localizing systems such as Hadoop or Sun Grid Engine or LSF for performance improvement. The distributed clusters(shards) and shared memory machines facilitate to optimize GATK engine, CPU and memory efficiency and parallelize analysis tools.

A Novel Clustered MongoDB-based Storage System for Unstructured Data with High Availability

Wenbin Jiang , Lei Zhang et al proposed various strategies to support complex query functions and provide high availability of the storage system[36]. The various strategies proposed are

- MyStore a new highly available data storage system for unstructured data
- Consistent hashing algorithm and virtual node are used for partitioning and distribute data on multiple nodes
- NWR mode is sued for automatic backup and data consistency
- Gossip protocol is used for failure information exchange among nodes
- Cache module and user friendly interface are used for improving the usability
- VeePalm virtual education experiment platform is used for implementation

This research review is focusing on how data partitioning and distribution are effectively done. Hence consistent hashing algorithm is taken for study and results are discussed. The nodes in cluster is assigned a random value. The data to be written is applied to hash function called Ketama hashing algorithm[37], the output of hash function is a key to map data to the nodes. When the number of nodes are limited in a cluster, the basic consistent hashing algorithm is not effective. Hence the virtual nodes are introduced, which assist to place the data on physical node effectively.

The system uses replication with replication factor *N*. The data firstly stored in a primary node and replicated in *N-1* nodes. replication done in clock wise direction of a ring. During *Put* operation, if number of replication is greater than *W* then *Put* operation is successful. The same is repeated for *Get* operation, if number of *Get* is greater than N then read operation is

THE IIOAB JOURNAL ISSN: 0976-3104

successful. For high consistency, the algorithm suggest *N=W* and *R=1*. for high availability *W=1* and *R+W<N*. This step improves write latency and performance.

The system is implemented with various XML data set and requestes are generated using Microsoft web application stress tool and results are compared with MySQL and MongoDB for storage operations. The parameters Get and Put performance, load balancing, throughput, response time and number of hits have taken for evaluations. The results showed that MySTore is providing better performance than MySQL and MongoDB.

Analysis of Sharding Schemes

Having set of research work done on various sharding schemes, implementation and it analysis, in this survey, we present critical aspects and key insights of sharding performance for Bigdata applications. The essential aspects of analysis are

- The existing survey work covered only architectural and Bigdata management elements. This research survey work focus more on storage management specifically sharding and replication strategies and detailed analysis of sharding implementations
- In addition to partition the database, query can also be sharded to provide better results
- Instead of applying general sharding schemes, the knowledge of data structure is significant factor to find new sharding schemes
- The segments frequently accessed can be replicated in more than servers reduce execution time
- Implementation of sharding schemes on variety of database showed that Researchers previously started the research works on bigdata in the fields of Biological data(Genome analysis), US census Dataset, Construction industry, Health insurance, E-commerce, Social network and Scientific literature
- MongoDB autosharding, Repartitioning, FODO algorithms and heat based load balancing supports dynamic sharding techniques for load balancing online
- Mapreduce merging programming(Mapreduce) with storage(HDFS). But Sharding and Mapreduce connector assists to isolate storage from programming increases performance.
- The MongoDB with Hadoop framework supports isolation of data nodes from process nodes that increase fault tolerance percentage. MongoDB-Hadoop connector is an open source software tools supported by MongoDB
- Sharding on Cloud servers with router based sharding schemes assists the Business layer to choose partial data to move on cloud and allows confidential data to stay in local server
- The survey also identified that Random and Range based partitioning are not well suited for load balancing
- The implementation results concluded that MongoDB and Cassandra will have built-in auto sharding facilities
- Depends on the application, sharding the data with same category reduces query response time
- Greedy algorithm and Gossip protocols assist to setup fault tolerant systems
- Scalable work driven partitioning makes static(initial level) partitioning as strong and reduces dynamic migration steps
- Flexible query systems assist the system to retrieve relevant data even if exact data is not able find.
- Mostly replication factor is fixed as N = 3, replicated at N-1 nodes
- Microsoft web application stress tool, iSearch test collection, VbLabelProp, Hadoop-MongoDB connector and Graclus are the tools used by sharding schemes
- Research work can be further moved on
    - Dynamic online sharding with minimum migration steps
    - Static partitioning with proper load balance
    - Algorithms to maintain consistency in replicas
    - Identifying new sharding schemes and keys
    - Hybrid solutions like MongoDB-Hadoop connector
    - In-memory computing
    - Query sharding

Table 2 : Analysis of Sharding Schemes

Sl. No	*Title*	*Sharding Technique*	*Sharding on*	*Parameters*	*Conclusion*
1	Knowledge Driven Query Sharding	Query Sharding Explicit Semantic Analysis	MeSH	Execution Time	Query sharding in the SONCA system utilize the computing resources optimally and execution time is considerably less than traditional(without sharding) technique.
Sl. No	**Title**	**Sharding Technique**	**Sharding on**	**Parameters**	**Conclusion**
2	Clustering-based fragmentation and data replication for flexible query	Clustering Fragmentation, Derived	Biological data MeSH	Execution Time Fragmentation Count	Few fragments stored in more than one servers gives better performance in

THE IIOAB JOURNAL ISSN: 0976-3104

	answering in distributed databases	fragmentation query re-writing Bin Packing Problem			query answering
3	Social BIMCloud: a distributed cloud-based BIM platform for object-based lifecycle information exchange	Horizontal Partitioning, Automatic Replication, Dynamic instance creation	AEC Industry	Not implemented	Performance depends on Bandwidth of Internet and reliability of service providers
4	The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data	Shared memory parallelization and distributed clusters(shards)	Sub region of Genome, SNP and SAM format of genome	Processor count and Elapsed time	Reduces elapsed time for end user analysis Parallelize analysis tools
5	Research on The Improvement of MongoDB Auto-Sharding in Cloud Environment	FODO algorithm	Random data with four fields int, long, string and double	Concurrent write and query operation	Concurrent number of operations increased
6	Migrating Application Data to the Cloud using Cloud Data Patterns	Router based sharding, critical data filter and Pseudonymizer	Health Insurance Company and External Auditing company	Literature Research	The authors guiding researchers to analyze various mapping of scenarios to data patterns and choose one among to implement as a single framework.
7	Point collection partitioning in MongoDB Cluster	RPS SPS KPS	3 sets - random and uniform distribution range	Response time	KPS provided better results than SPS and RPS techniques.
8	Data management in cloud environments: NoSQL and NewSQL data stores	Study on partitioning, Replication, consistency and concurrency	Study paper	-	Range and Hash based partitioning schemes are compared
9	An Overview of Newly Open-Source Cloud Storage Platforms	Comparison	MongoDB Cassandra ABIcloud	Selection of best storage mechnism	MongoDB has been selected as storage mechanism for their own experiments.
10	High dimensional biological data retrieval optimization with NoSQL technology	Microarray Key -value data model on HBase	transcriptomic data set from NCBI GEO	Data retrieval time	HBase retrieval rate is higher than MongoDB and RDBMS
11	Sharding for Literature Search via Cutting Citation Graphs	Cutting the citation and co-citation graphs, iSearch test collection, Graclus - Graph clustering software	Scientific Literature	Retrieval of relevant documents	co-citation graph performance is better than citation graph
Sl. No	**Title**	**Sharding Technique**	**Sharding on**	**Parameters**	**Conclusion**
12	Morphus: Supporting Online Reconfigurations in Sharded NoSQL Systems	Online reconfiguration, Greedy Algorithm,	Amazon reviews	Read and Write Latency Scalability Network	Morphus scales well with increasing replica set

THE IIOAB JOURNAL ISSN: 0976-3104

		Bipartite matching		optimization	
13	Research on Improvement of Dynamic Load Balancing in MongoDB	Heat based load balancing system	Simulation Xenserver	Resource utilization Migration cost	Resource utilization is controlled and reduced migration cost.
14	Scalable transactions in cloud data stores	Workload driven data partitioning	E-commerce Transactions	throughput, response time, number of distributed transactions	Less number of migrations and distributed transactions
15	Sharding Social Networks	VbLabelProp BlockShard	LiveJournal and Twitter	Load balance No of shard access	Network aware sharding with METIS and VBLabelProp outperform Random and Geo based sharding techniques
16	Performance Evaluation of a MongoDB and Hadoop Platform for Scientific Data Analysis	Hadoop-MongoDB connector	US Census Data set	Read and write performance	The MongoDB with Hadoop framework supports isolation of data nodes from process nodes that increase fault tolerance percentage
17	Using Paxos to Build a Scalable, Consistent, and Highly Available Datastore	Spinnaker's Paxos Zookeeper	Spinnaker	Read and Write latency	Read operation latency in significantly reduced compared to Cassandra. Write operation results are 5% to 10% worse than Cassandra
18	SWORD: Scalable Workload-Aware Data Placement for Transactional Workloads	hypergraph compression technique, Incremental data repartitioning, SWORD, fine grained quorum	TCP-C Benchmark	Throughput Query response time End-to-end transaction time	Improved query routing time, overall end-to-end time and throughput
19	A novel clustered MongoDB-based storage system for unstructured data with high availability	Consistent Hashing algorithm with virtaul nodes	XML data set	Get and Put performance Through put Response time Load balancing	MySTore is providing better performance than MySQL and MongoDB

THE IIOAB JOURNAL
ISSN: 0976-3104

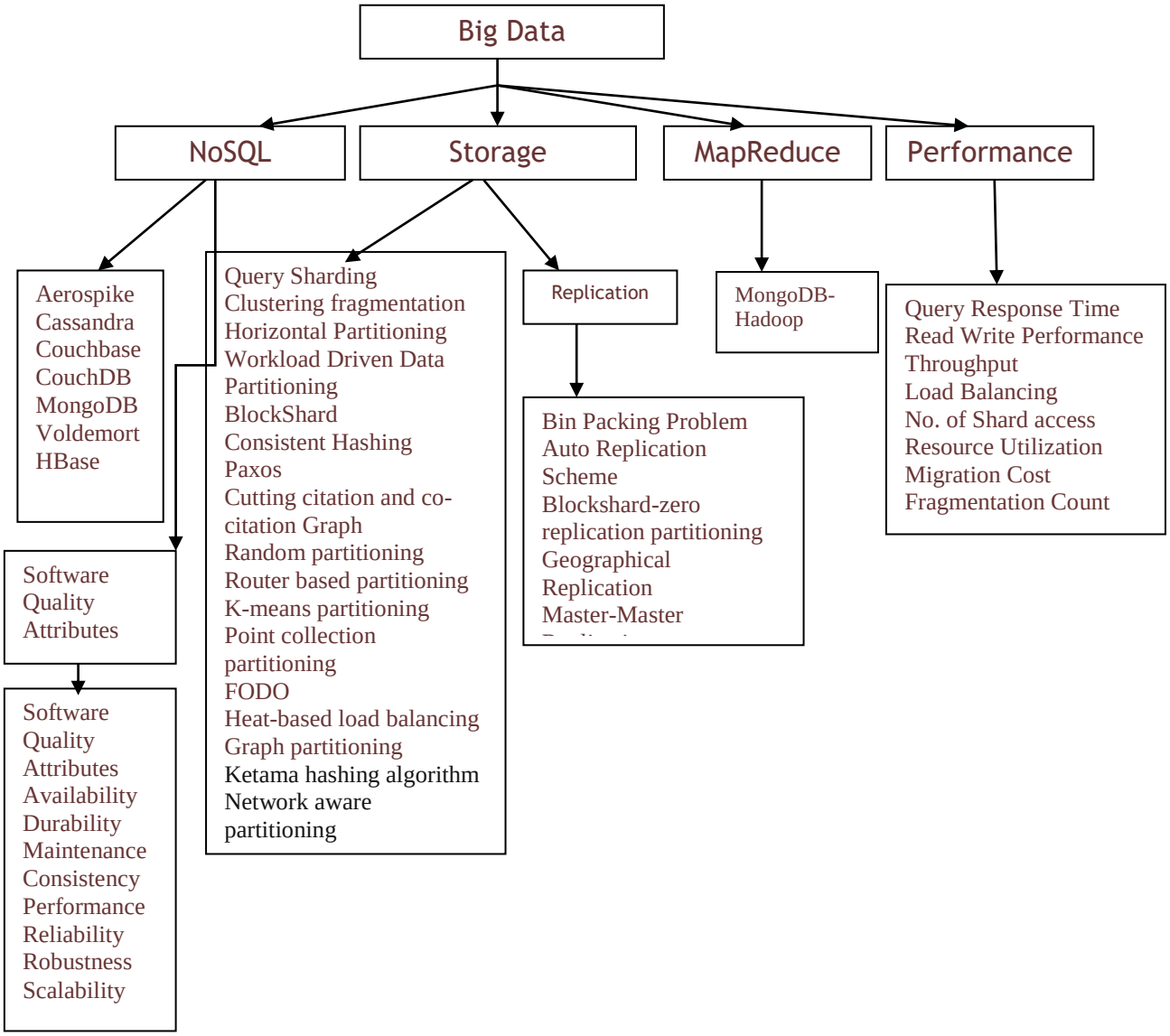


Fig 1 Technologies and Parameters Listed in this Survey

The Table 2 shows various sharding techniques and its concluding remarks. Fig 1 listed various sharding schemes, replication strategies, NoSQL DBs, Software quality parameters and performance parameters taken for survey in this research review paper. The above detailed analysis, technologies mentioned in Fig 1 and the concluding points in Table 2 help out the researchers to move further in sharding schemes.

CONCLUSION

In this survey, we studied, investigated, categorized, and analyzed critical aspects of sharding schemes for Bigdata management systems. The survey on sharding schemes implemented from the year 2010 have been done and the results are analyzed. The existing survey papers also direct the researchers to implement the usecase with different architecture and NoSQL DBs assists to compare and find the optimal solution in Bigdata management. This research review paper also identified that dynamic load balancing, sharding keys and algorithms to find minimum migration steps, in-memory computing, maintaining consistency in replicas and query sharding are the major research areas in this direction.

THE IIOAB JOURNAL
ISSN: 0976-3104

CONFLICT OF INTEREST

Authors declare no conflict of interest.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the family member and collegues for given support.

FINANCIAL DISCLOSURE

No financial support was received to carry out this project.

REFERENCES

[1] Min Chen, Shiwen Mao, Yunhao Liu. " Big Data: A Survey" *Journal of Mobile Network Applications* 19 (2014) :171–209.

[2] Laney D 3-d data management: controlling data volume, velocity and variety. META Group Research Note, 6 February 2001

[3] Cattell R Scalable sql and nosql data stores. ACM SIGMOD Record 39(4) (2011):12–27

[4] Wu, Lengdong, Liyan Yuan, and Jiahuai You. "Survey of large-scale data management systems for big data applications." *Journal of computer science and technology* 30.1 (2015): 163-183.

[5] Campbell D, Kakivaya G, Ellis N. Extreme scale with full SQL language support in Microsoft SQL Azure. In *Proc. the 2010 ACM SIGMOD International Conference on Management of Data*, June 2010, pp.1021-1024.

[6] DeCandia G, Hastorun D, Jampani M, Kakulapati G, Lakshman A, Pilchin A, Sivasubramanian S, Vosshall P, Vogels W. Dynamo: Amazon's highly available key-value store. In *Proc. the 21st ACM SIGOPS Symposium on Operating Systems Principles*, October 2007, pp.205-220.

[7] Cooper BF, Ramakrishnan R, Srivastava U *et al.* PNUTS: Yahoo!'s hosted data serving platform. In *Proc. the 34th International Conference on Very Large Data Bases*, August 2008, pp.1277-1288. Lakshman A, Malik P. Cassandra: A decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 2010, 44(2): 35-40.

[8] Joshi A, Sam H, Charles L. Oracle NoSQL database scalable, transactional key-value store. In *Proc. the 2nd International Conference on Advances in Information Mining and Management*, October 2012, pp.75-78.

[9] Ghemawat S, Gobioff H, Leung S T. The Google file system In *Proc. the 19th ACM Symposium on Operating Systems Principles*, December 2003, pp.29-43.

[10] Chang F, Dean J, Ghemawat S *et al.* Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems*, 2008, 26(2): Article No.4.

[11] Baker J, Bond C, Corbett J *et al.* Megastore: Providing scalable, highly available storage for interactive services. In *Proc. the 5th Biennial Conference on Innovative Data Systems Research*, January 2011, pp.223-234

[12] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 2008, 51(1): 107-113

[13] Lourenço, João Ricardo, et al. "Choosing the right NoSQL database for the job: a quality attribute evaluation." *Journal of Big Data* 2.1 (2015): 1-26.

[14] Kuznetsov S, Poskonin A (2014) Nosql data management systems. Program Comput Softw 40(6):323–332

[15] Haughian G (2014) Benchmarking replication in nosql data stores. Dissertation, Imperial College London

[16] Gudivada VN, Rao D, Raghavan VV (2014) Nosql systems for big data management. In: Services (SERVICES), 2014 IEEE World Congress On. IEEE, Anchorage, AK, USA. pp 190–197

[17] Krasuski, Adam, and Marcin S. Szczuka. "Knowledge Driven Query Sharding." *CS&P*. Vol. 225. 2012

[18] Wiese, Lena. "Clustering-based fragmentation and data replication for flexible query answering in distributed databases." *Journal of Cloud Computing* 3.1 (2014): 1-15.

[19] Das, Moumita, Jack CP Cheng, and Srinath S. Kumar. "Social BIMCloud: a distributed cloud-based BIM platform for object-based lifecycle information exchange." *Visualization in Engineering* 3.1 (2015): 1-20

[20] Liu, Yimeng, Yizhi Wang, and Yi Jin. "Research on the improvement of MongoDB Auto-Sharding in cloud environment." *Computer Science & Education (ICCSE), 2012 7th International Conference on*. IEEE, 2012.

[21] Strauch, Steve, Vasilios Andrikopoulos, and Thomas Bachmann. "Migrating application data to the cloud using cloud data." *3rd International Conference on Cloud Computing and Service Science,(CLOSER)*. 2013.

[22] Zhang, Shuai, et al. "Point collection partitioning in MongoDB Cluster."

[23] Grolinger, Katarina, et al. "Data management in cloud environments: NoSQL and NewSQL data stores." *Journal of Cloud Computing: Advances, Systems and Applications* 2.1 (2013): 22

THE IIOAB JOURNAL
ISSN: 0976-3104

[24] Gu, Genqiang, et al. "An overview of newly open-source cloud storage platforms." *Granular Computing (GrC), 2012 IEEE International Conference on*. IEEE, 2012.

[25] Wang, Shicai, et al. "High dimensional biological data retrieval optimization with NoSQL technology." *BMC genomics* 15.Suppl 8 (2014):

[26] Zhao, Haozhen. "Sharding for literature search via cutting citation graphs."*Big Data (Big Data), 2014 IEEE International Conference on*. IEEE, 2014.

[27] Ghosh, Mainak, et al. "Morphus: Supporting Online Reconfigurations in Sharded NoSQL Systems." (2014).

[28] Wang, Xiaolin, Haopeng Chen, and Zhenhua Wang. "Research on Improvement of Dynamic Load Balancing in MongoDB." *Dependable, Autonomic and Secure Computing (DASC), 2013 IEEE 11th International Conference on*. IEEE, 2013

[29] Ahirrao, Swati, and Rajesh Ingle. "Scalable transactions in Cloud Data Stores." *Advance Computing Conference (IACC), 2013 IEEE 3rd International*. IEEE, 2013.

[30] Duong, Quang, et al. "Sharding social networks." *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 2013.

[31] ede, Elif, et al. "Performance evaluation of a mongodb and hadoop platform for scientific data analysis." *Proceedings of the 4th ACM workshop on Scientific cloud computing*. ACM, 2013.

[32] Rao, Jun, Eugene J. Shekita, and Sandeep Tata. "Using Paxos to build a scalable, consistent, and highly available datastore." *Proceedings of the VLDB Endowment* 4.4 (2011): 243-254

[33] H. Hsiao and D. J. Dewitt. Chained Declustering: A New Availability Strategy for Multiprocessor Database Machines. In *ICDE*, pages 227–254, 1990

[34] Quamar, Abdul, K. Ashwin Kumar, and Amol Deshpande. "SWORD: scalable workload-aware data placement for transactional workloads." *Proceedings of the 16th International Conference on Extending Database Technology*. ACM, 2013.

[35] McKenna, Aaron, et al. "The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data." *Genome research* 20.9 (2010): 1297-1303.

[36] Wenbin Jiang, Lei Zhang et al. " A novel clustered MongoDB-based storage system for unstructured data with high availability, Journal of Computing (2014) 96:455–478

[37] Petrovic J (2008) Using memcached for data distribution in industrial environment. In: Proceedings of 2008 3rd international conference on systems (ICONS'08). IEEE, Piscataway, pp 368–372