Workload Aware Incremental Repartitioning of NoSQL for Online Transactional Processing Applications

Anagha Bhunje, Swati Ahirrao
Departement of Computer Engineering, Symbiosis International University, India.

Article Info

Article history:

Received May 23, 2017
Revised Dec 20, 2017
Accepted Jan 10, 2018

Keyword:

Distributed transactions,
Fuzzy C-Means clustering algorithm.
Hyper graph,
NoSQL (Not Only Structured Query Language), Incremental repartitioning,
OLTP (Online Transaction Processing)

ABSTRACT

Numerous applications are deployed on the web with the increasing popularity of internet. The applications include, 1) Banking applications, 2) Gaming applications, 3) E-commerce web applications. Different applications reply on OLTP (Online Transaction Processing) systems. OLTP systems need to be scalable and require fast response. Today modern web applications generate huge amount of the data which one particular machine and Relational databases cannot handle. The E-Commerce applications are facing the challenge of improving the scalability of the system. Data partitioning technique is used to improve the scalability of the system. The data is distributed among the different machines which results in increasing number of transactions. The work-load aware incremental repartitioning approach is used to balance the load among the partitions and to reduce the number of transactions that are distributed in nature. Hyper Graph Representation technique is used to represent the entire transactional workload in graph form. In this technique, frequently used items are collected and Grouped by using Fuzzy C-means Clustering Algorithm. Tuple Classification and Migration Algorithm is used for mapping clusters to partitions and after that tuples are migrated efficiently.

Corresponding Author:

Anagha Bhunje,
Departement of Computer Engineering,
Symbiosis International University, India.
Email: Anagha927@gmail.com

1. INTRODUCTION

The amount of the data generated and stored is increasing in a tremendous way. The performance of the enterprises is fast, smart and efficient with the use of the big data. A large database is needed for storing data in order to meet heavy demands. Such data are distributed across different machines. Single machine cannot handle such huge amount of data. Relational database does not efficiently handle such data. Relational databases have fixed schema. So NoSQL data stores are used to scale easily over multiple servers.

The relational databases have some scalability issues. It is unable to handle demands of the scalable applications. The performance of the system reduces as data volume grows. NoSQL (Not Only SQL) databases are used to Scale out, as and when the data grows. D.J.Dewitt and J.Gray [1] Describes the partitioning techniques to improve the scalability of the system. There are two types of partitioning Techniques.1) Horizontal partitioning and 2) Vertical partitioning. The commonly used horizontal partitioning techniques are 1) Round-Robin, 2) Range 3) Hash partitioning. In Round Robin technique, each node is allocated with time slot and has to wait for its turn. It simply allocates the job in Round Robin fashion. In Range partitioning technique, data is partitioned into ranges based on the partitioning key. Partitioning key need to be specified as per the requirement. It does not consider the load on different machines. Horizontal partitioning techniques such as Range or Hash based techniques are unable to capture the data access pattern i.e. relation between the data items.

Accessing data tuples from geographically distributed server affects the data base scalability. Due to rapid growth in the requests, response time of server is slowed down. So scaling modern OLTP applications is the challenging task. This technique does not consider the relation between the tuples and end up with the cluster of the uncorrelated data tuples on the same partition. It results in the increased costs of transactions.

Graph structure is a way to express relationship between different objects in the form of vertices and edges. Several users are connected to each other with the help of social network. Social network contains information of different entities. It contains information about personal details, friends information of the entities. There exists at least one relationship between the entities. These entities are distributed among different servers due to which the numbers of the transactions are increased. When numbers of entities involved in the graph are increased, then graph size is increased.

In graph representation, edge can connect only one node at a time. One start node and one destination node is specified which must be on the same graph. In order to reduce the graph size, Hyper Graph Representation technique is used in this paper. In Hyper Graph Representation technique, edges can connect to more than two nodes at a time. One start node is specified and existing relationships between the attributes are found out. Start node and destination node need not be on same graph.

The difference between Graph representation and Hyper Graph Representation technique is summarized as Table 1.

Table 1. The difference between Graph Representation and Hyper Graph Representation Technique

Graph Representation	Hyper Graph Representation
1. Edge can connect to only one node at a time. One start node and one destination node is specified.	1. Edge can connect to more than two nodes. One start node is specified and relationships between the items are found out.
2. Start and destination node must be on same graph	2. Start and destination nodes need not to be on same graph. It belongs to other graph also.
3. Graph size is increased	3. Graph size is reduced
4. Number of the transactions is more.	4. Number of the transactions is less.

In order to reduce the number of distributed transactions, Hyper Graph Representation technique is used in this paper.

Existing partitioning techniques does not consider the relation among the attributes. Load balancing is not done properly in these techniques. Some amount of the data gets lost due to load imbalance on the server. So Workload Aware Partitioning Techniques are used. Workload Aware partitioning technique is used where related data items are kept on one partition. Most of the partitioning technique works on relational databases. But it handles only static data. Such techniques handle limited amount of data. It does not deal with NoSQL databases. NoSQL databases are used to handle the huge amount of the data.

Incremental Repartitioning technique is the useful technique to improve the response time of the server. In incremental Repartitioning technique, most frequently accessed items are gathered together on one server. So, Load on the server is equally balanced.

The contributions of our work is as Follows,

1. Design of the Workload Aware Incremental Repartitioning Technique in Couchdb is introduced. The workload is monitored by it and frequently accessed data items are kept on one partition in order to minimize the number of the transactions.
2. Implementation of the Workload Aware Incremental Repartitioning Technique in Couchdb.
3. Performance of this technique is evaluated by using different quality Metrics which are as Follows
    a. Response time
    b. Throughput
    c. Impact of the distributed transactions
    d. Load imbalance derivation
    e. Inter Server data migration.

Further this paper is structured as follows:

In Section 2, papers related to scalability and database partitioning is discussed. Section 3 gives brief overview of proposed system. Design of Workload Aware Incremental Repartitioning System is presented in Section 3. An Implementation details in Section 4 explains implementation and algorithms that are used for implementing this Workload Aware Incremental Repartitioning technique. Section 5 describes the results and provide conclusion.

2. RELATED WORK

Curino et al [2] Describe the workload aware approach for achieving scalability with the help of graph partitioning. The main goal of this paper is to reduce the number of the distributed transactions in order to produce the balanced partitions. The Entire transactional workload is represented with the help of the Graph Representation technique. Node represents the data items. Relationship between the data items are represented by the edges. The edges that connect the two nodes are used within the same transactions. Graph Representation technique helps in balancing weight of the partitions. The Graph Partitioning Algorithm is used to find the balanced partitions in order to reduce the number of distributed transactions. METIS tool is used to partition the graph.

The graph generated by the Schism is large. Numbers of tuples involved in the transactions are large then graph size is increased. The number of the transactions is increased. The changing workload is not monitored by this Workload Aware Algorithm. This is the Static Partitioning technique. Once the partitions are formed, those partitions do not change.

Quamar [3] addresses the problem of the Scalable Transactional Workload. The number of the transactions is increased to access the data from the several machines.

Graph size is also increased due to number of the items. The numbers of the partitions are more. In order to reduce the graph size, the new technique of Scalable Workload Aware Data Partitioning and Incremental Repartitioning, to handle the frequent changes in the workload is developed. SWORD works in 3 steps. In Data Partitioning and Placement, workload is represented with the help of the compressed Hyper Graph. Data is horizontally partitioned. This module is responsible for taking the decision of placing the data across the partitions. In order to reduce the graph size, the entire Transactional Workload is represented with Hyper Graph Representation technique. The tuples of the transactions are represented by the nodes. The edges are called as hyper edges which can connect to any number of the vertices at a time. In Graph Representation technique, one node at a time can connect to only one node. With the Hyper Graph Representation techniques, partitions formed are less in number. Hyper Graph Compression technique is used to manage the problem of the memory and computational requirements of the Hyper Graph storage which affects the performance of the partitioning and repartitioning. Hash partitioning technique uses Simple and easy method or way to compute the hash function to compress the hyper graph. Incremental repartitioning technique is used to monitor the workload changes.

Replication technique is used for placing the data across multiple machines. But Replication technique requires more nodes for placing the data. The numbers of the transactions are not reduced by using this technology.

Miguel Liroz-Gistau et al. [4] introduces new Dynamic Partitioning Algorithm Dynpart which is useful for dynamically growing databases. The software companies like Facebook, Google, and Amazon need to handle the billion of users. It deals with the huge amount of the data. The applications where data items are continually added to the database are suffered from the problem of the data management. Dynpart is the dynamic partitioning algorithm which handles the dynamically growing databases. It takes the new data items which are continually added to the applications by considering an affinity between the data items and partitions. The input of the data items to the algorithm is set. The output of the algorithm is to find the best balanced partition to place the data. It selects the partition based on the close relation between the attributes. If there are several fragments that have the highest affinity (close relation) then the smallest fragment is selected in order to keep the partitioning balanced.

Shivanjali Kanase and Swati Ahirrao [9] introduce Graph Based Workload Driven Partitioning System in NoSQL databases. Many real life applications such as E-commerce applications, banking applications generate huge amount of the data. In order to increase the scalability, partitioning technique is used. It distributes the data across many servers to balance the load. If the groups are not formed properly then it results in increasing the number of the distributed transactions.

Graph Representation technique is used for the representation for the transaction load (workload).The attributes of the transactions are represented as the nodes in the graph. The nodes are connected by the edges. As numbers of the transactions are increased then graph size is also increased. In order to reduce the graph size, following steps are followed

1) Transaction level sampling: Number of the edges representing in the graph are reduced. Only relevant transactions are shown in the graph.
2) Tuple level sampling: Number of the tuples shown in the graph is reduced.
3) Relevance Filtering: The tuples which give less information about the transaction are discarded from the graph (rarely used tuples).

Graph partitioning technique is used to find the k balanced partitions. Recursive Bisection method is used to find the k partitions. In coarsening phase, adjacency matrix is prepared from the Graph

Representation. All adjacent edges which are incident on each base node are sorted into the decreasing order according to their cost. All these nodes are stored in the queue. At each step, the first node is combined with the base node and it is marked as matched and it cannot be added with another node. This technique is called as Heavy Edge Maximal Matching. Smaller graph is partitioned into two parts such that number of nodes in each partition is equal. Refinement algorithm is used for making such partitions. Decision tree classifier is used for generating the rules. These rules are used to map the groups which are obtained from Refinement Algorithm to partitions.

In this paper, Recursive Bisection method is used for partitioning the graph. Only two partitions are formed in this method. Load is not properly balanced. This algorithm gives less accurate result.

The author Andrew Pavlo [10] introduced a new approach for automatically partitioning a database in a shared nothing, parallel Database Management System (DBMS). Horticulture is the automatic design tool which helps in selecting physical layout for DBMS. The new database design is considered the amount of the data and transactions assigned to the single partition. Horticulture analyses a database schema, the structure of the applications stored procedures, and a sample transaction workload, and then automatically generates partitioning strategies that minimizes distribution overhead while balancing access skew. Horticulture makes use of Large Neighbourhood Search (LNS). LNS compares potential solutions with a cost model that analyses the DBMS will perform using a particular design for the sample workload trace without needing to actually deploy the database.

Reduction in the number of the distributed transactions in shared nothing distributed database is difficult task for the transactional workloads. Nowadays, there is a tremendous growth in the data volumes.

Rajkumar Buyya [12] introduces Workload Aware Incremental Repartitioning technique for cloud applications. The proposed idea is implemented on the relational databases. Workload Aware Incremental Repartitioning technique is used to reduce the number of the transactions and to improve the response time of the server. The entire workload is represented as Hyper Graph or Graph. K-way min cut graph clustering algorithm is used to balance the load among the partitions and then clusters are placed across the set the physical servers.

In k-way, data point must exclusively belong to one cluster. It is less accurate.

3. PROPOSED SYSTEM OVERVIEW

The input to the Workload Aware Incremental Repartitioning System is transaction loads (number of the transactions) and the output is the number of the equally balanced partitions, which minimizes the number of the distributed transactions. As shown in Figure 1.

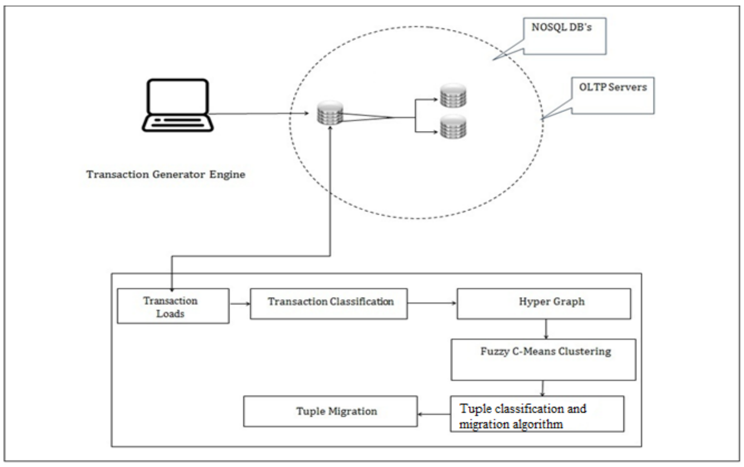
NOSQL DB's
OLTP Servers
Transaction Generator Engine
Transaction Loads
Transaction Classification
Hyper Graph
Fuzzy C-Means Clustering
Tuple classification and migration algorithm
Tuple Migration

Figure 1. The input to the Workload Aware Incremental Repartitioning System is transaction loads

The basic process is stated in the following steps:

a. Transaction Loads: The system takes the input as number of the transactions. In this section, the design of Couchdb has been modelled from TPC-C schema. Mapping of these nine tables (warehouse, customer, district, history, new order, item, order, order line and stock) into documents of Couch DB is performed.

b. Transaction Classification: Transaction data is classified on the basis of warehouse id. .Warehouse ids are found out while executing these transactions. Relationship between the items is considered. From the given transactional data, system tries to find out distinct warehouse id.When new order is placed then based on the warehouse id, transaction data is classified.

c. Hyper Graph Representation: Transaction workload is represented with the help of Hyper Graph Representation technique. As compared to the Graph Representation technique, Hyper Graph Representation technique is most useful technique. Hyper Graph is a graph in which edge can connect to any number of the nodes. Here edge indicates the relationship. It takes the output of the transaction classification i.e. Unique Warehouse ids and Unique Customer ids.

d. Fuzzy C-means Clustering Algorithm: It is a process of grouping which allows one piece of the data to belong to two or more clusters. The Fuzzy C means clustering algorithm is applied on the created Hyper Graph.

Comparison between K-way partitioning algorithm and Fuzzy C-means clustering algorithm is shown in Table 2.

Table 2.Comparison between K-way Partitioning Algorithm and Fuzzy C-means Clustering Algorithm

K-way partitioning algorithm	Fuzzy C-means clustering algorithm
1. In k-way, data point must exclusively belong to one cluster	1. In Fuzzy C-means algorithm, data point is assigned membership to each cluster centre as a result of which data point may belong to more than one cluster centre.
2. It is less accurate.	2. It is more accurate.

e. Tuple Classification techniques: Classification technique is used for mapping partitions to clusters obtained by Fuzzy C-means algorithm. Total five clusters are formed for each customer. Five levels which are as follows:
   1) Very High,
   2) High,
   3) Medium,
   4) Low and
   5) Very Low.

The high level cluster values only are considered for selection. It is considered as the frequently accessed items. Other two cluster data is considered as rarely accessed. So it is not considered.

4. IMPLEMENTATION DETAILS

4.1. TPC-C benchmark

TPC-C benchmark is an OLTP workload. The benchmark represents a wholesale provider with the geographically distributed warehouses and districts. It measures the performance of the Online Transaction Processing System. The benchmark consists of the five different Transactions.
1) Entering and Delivering orders,
2) Recording payments,
3) Checking the status of orders, and
4) Monitoring the level of stock at the warehouses.

4.1.1. New order

New order transaction is the combination of read and writes transactions. It creates a new order for the customer and places an order according to the customer need.

4.1.2. Payment

Payment transaction is also the combination of read and writes transaction. When a payment transaction is executed it updates the balance of customer.

4.1.3. Order status

Order status transaction is read only transaction. It tracks the status of the customer that is customer’s last order.

4.1.4. Delivery

Delivery transaction is also a read and writes transaction. It consists of group of 10 new orders that is orders not yet delivered to the customer.

4.1.5. Stock level

Stock level transaction is read only transaction. It decides the quantity of recently sold things which have stock below the threshold.

Usually NEW ORDER transactions are 45%, PAYMENT transactions are 43% and ORDER STATUS, STOCK and DELIVERY transactions are 4%.

4.2. Workload Aware Incremental Repartitioning

4.2.1. Problem Definition

Let S= { } be as system for Incremental Repartitioning for OLTP transaction. Let Input as T= { t_1 , t_2 , t_3 ,……………t_n} Where t_i= Transaction tuples. S= {T}. In Incremental Repartitioning technique, only unique transactions are considered. Let the transactional workload is represented by H_g= {$H_{g0,}$ H_{g1} , H_{g2}……………H_{gm}}. H_{gi} = set of the unique transactions in T. The set of the distributed and non distributed transactions are represented as T_d and $T_{d'}$.$T = T_d \cap T_{d'} = \phi$. Transactions are classified based on the warehouse ids. The distributed or non distributed transactions that repeat multiple times within new order transaction are considered. Such transactions are collected together and kept on one partition.

The proposed system is as Follows

1) In this step all the transactions are fed to the system so that the database schema of the web application can be generated at the Server end.
2) As the transactions arrive at the server all the transactions are classified by using the Hyper Graph based on the number of warehouses for the respective user. This is achieved by identifying the unique user as nodes in the hyper graph coarsening technique.
3) Fuzzy C means Clustering is used to cluster the number of users based on the warehouse id. This is accomplished using matrix evaluation of the user occurrences based on the fuzzy crisp values like very low, low, medium, high and very high.
4) Then the system uses the decision tree to classify the user belongs to high and very high clusters are considered to be as partitioned deserving entities.

4.3. Hyper Graph Representation:

The transactional workload is represented with the help of the hyper graph. Neo4j is a high-performance, NoSQL graph database is used for the storage and representation of the graph database in this paper. The number of the unique warehouses and numbers of customers are represented as nodes in the graph, which are connected by the edges. The edges represent the relationship. The W_ID is considered as the base node. The edge cost indicates the total number of the transaction which co-accesses this pair of tuples. For Example is shown in Table 3.

Table 3. The Example of Edge Cost Indicates the Total Number of the Transaction which Co-Accesses this Pair of Tuples

	Customer name	Item	Quantity
1)	Anagha	Laptop	1
2)	Aboli	Mobile	1
3)	Janhavi	Camera	1
4)	Sneha	Laptop	1
5)	Jay	Shoes	1
6)	Hema	Mobile	1

Hyper Graph Representation as Shown in Figure 2.

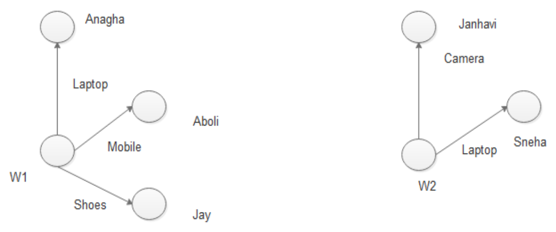

Figure 2: Hyper Graph Representationg

Transaction data is classified on the basis of number of warehouses. Depending on the count of the warehouses, numbers of the Hyper Graphs are formed. As there are two unique warehouses, then two separate small graphs are formed. Graph size is reduced. When the new transaction occurs, it first checks the presence of the node for the particular attribute value in the graph. If the node is absent thenthe new node for that attribute value is added to the graph and respective cost on edges is also updated.

```
Input:          Set S = {$W_i$, $C_n$, $I_t$}
 Where
 $W_i$- is the Warehouse ID
 $C_n$ – Customer id
 $I_t$- Item
Output:   Hyper Graph G ($W_i$, $C_n$ ,$I_t$)
Algorithm:
Start
Get the Set S for the new order transaction as input
 FOR i=0 to Size of S
 Separate Wi, Cn and $I_t$ into List $L_w$,$L_c$
 END FOR
 Get unique elements form $L_w$and $L_c$
 Nw=Size of $L_w$ (Number of nodes for warehouse id)
 $N_c$=Size of $L_c$ (Number of nodes for Customer id)
 Identify the relational Edges E
 Form Graph G
 return G
Stop
```

4.4. Fuzzy C-means Clustering Algorithm:

```
Input:
 Number of the unique warehouse ids,
 Number of the unique customer ids.
Output:
 Number of the balanced clusters.
Algorithm:
 Start
 Read Data Set
 Pre-Processing Selection of Relevant Attributes From Dataset
Unique Attributes and Removal of duplicates
 Membership Matrix computation using Attributes
Get MinMax for Rule Generation
 Generate five Rule very Low, Low, Medium, High, and Very High
 R {r1, r2, r3, r4, r5}
Perform Clustering based on fuzzy rules.
Clusters {c1, c2, c3, c4, c5}
Stop
```

4.5. Tuple classification and Migration Algorithm

```
Input :
 Fuzzy clusters $F_c$ = {$F_{c1}$,$F_{C2}$…….$F_{Cn}$}
 Transaction data D={ $d_1$,$d_2$……$d_n$}
Output : Classification Labels
Algorithm:
 Start
FOR each of D
 Get di and Identify warehouse as $W_1$
 FOR each of Fc
        Identify warehouse as $W_2$
        IF $W_1$=$W_2$
        Add into vector V
        End Inner FOR
        End outer FOR
         FOR each sub vector of V
        Get $\mathbf{V_{ij}}$
 Check for high Fuzzy Crisp values
 Check for Warehouse
 Add into classification Label L={ L1, L2}
 End For
 Return L
 Stop
```

5. RESULT

In this section, the performance of the number of the transactions and number of the warehouses are evaluated on the basis of the following quality metric.

a. Response time
b. Throughput
c. Impact of the distributed transactions
d. Load imbalance derivation
e. Inter Server data migration.

The goal of this experiment is to improve the response time of the server and minimize the number of the transactions. Figures (3-6) Shows the response time required for executing the transactions in single DB and Multi DB.Along with x-axis, there is number of the users and along the y-axis; there is response time (in Milli seconds). It is observed that response time of the server in single DB is more and response time of the server in Multi DB is less. Figure (10-13) shows the throughput of the system in both the cases. The throughput for the numbers of the warehouses has been measured. Each time throughput is more in case of Multi DB. Along with x-axis, there is number of the transactions and along the y-axis; there is Throughput. Figure 7 shows the impact of the DT (in percentage). Along with x-axis, there is number of the transactions and along the y-axis; impact of the transaction. Figure 8 represents load imbalance derivation in Multi DB. Along with x-axis, there is number of the transactions and along the y-axis; there is derivation. This value denotes the equally balanced value. Due to this value both the partitions are having same number of the transactions for execution. Figure 9 shows the inter server data migration. This is mean value for the partitions to execute the number of the transactions. Figures (10-13) shows the throughput required for executing the transactions in single DB and Multi DB.Along with x-axis, there is number of the users and along the y-axis; there is throughput. Throughput is large in case of Multi DB as compared to Single DB.

This clearly represents that the proposed method is efficiently incorporated and improves the response time of the server and minimizes the number of the transactions with improved scalability.

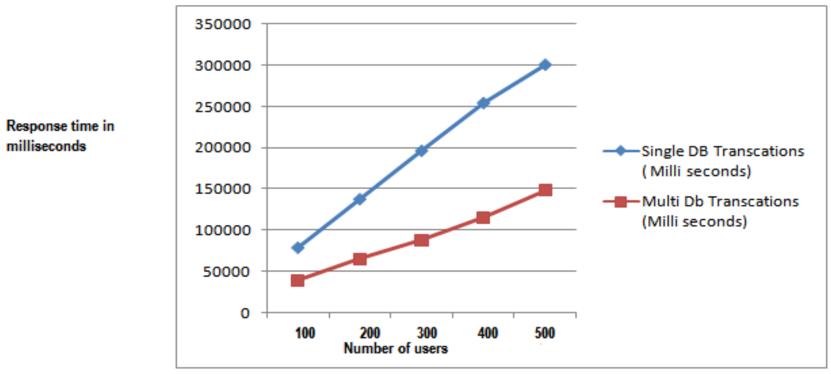

Figure 3 Response time for Single DB transactions and Multi DB transactions (in Milli Seconds) (Warehouse=5)

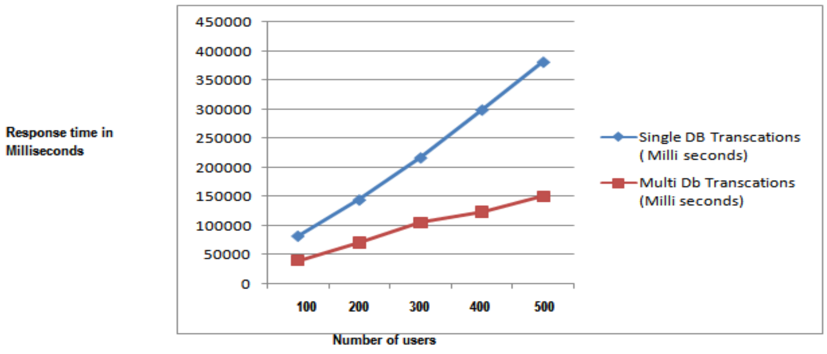

Figure 4. Response time for Single DB transactions and Multi DB transactions (in Milli Seconds) (Warehouse=10)

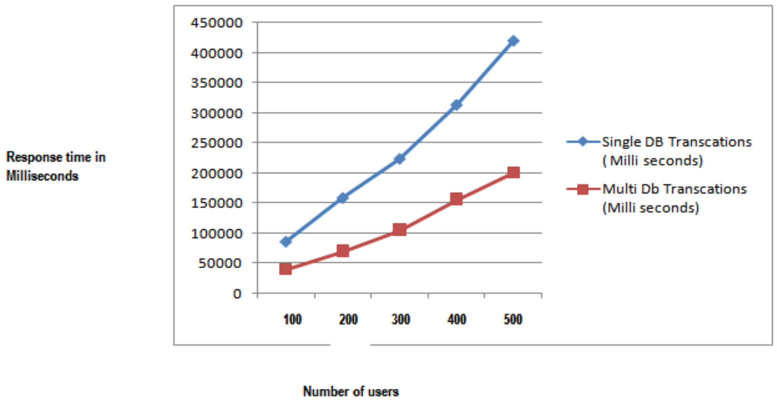

Figure 5. Response time for Single DB transactions and Multi DB transactions (in Milli Seconds) (Warehouse=15)

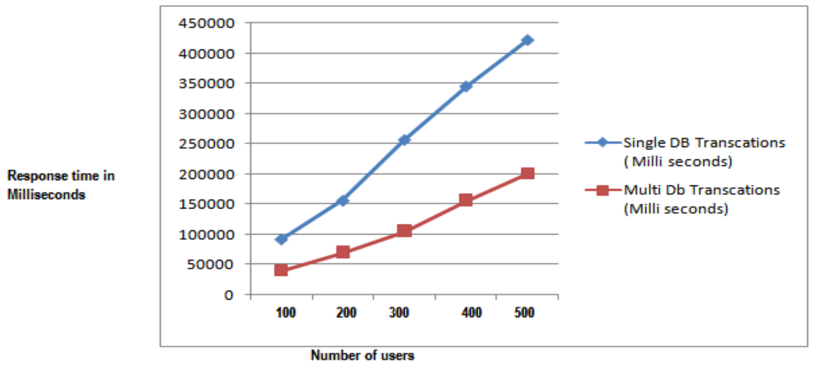

Figure 6. Response time for Single DB transactions and Multi DB transactions (in Milli Seconds) (Warehouse=20)

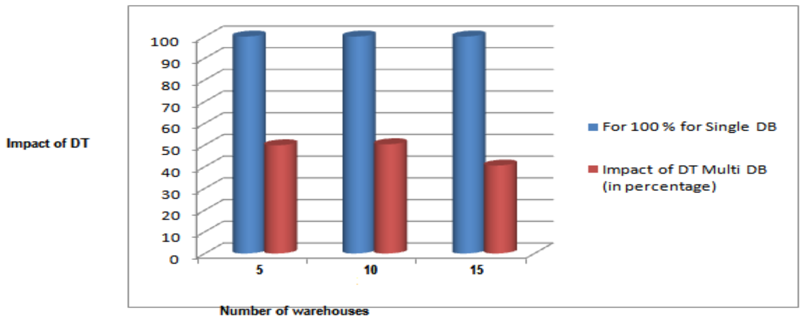

Figure 7. Impact of the DT (in percentage) for Single DB and Multi DB

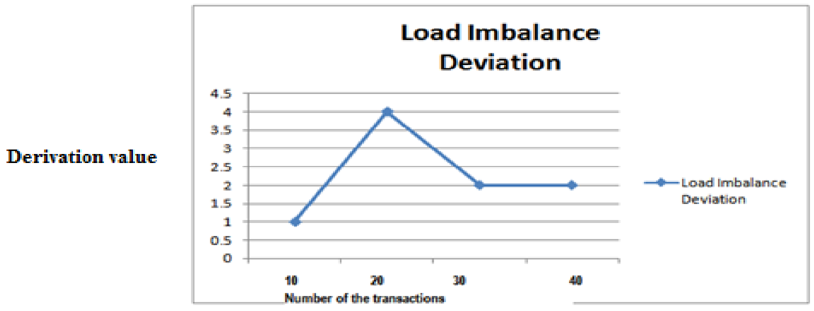

Figure 8. Load Imbalance Derivation in Multi DB

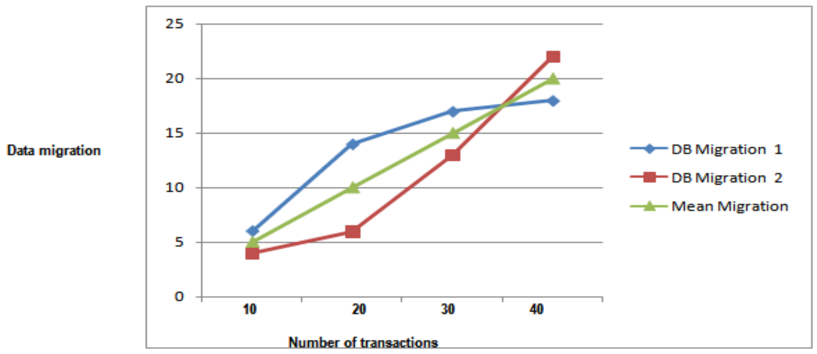

Figure 9. Inter server data migration

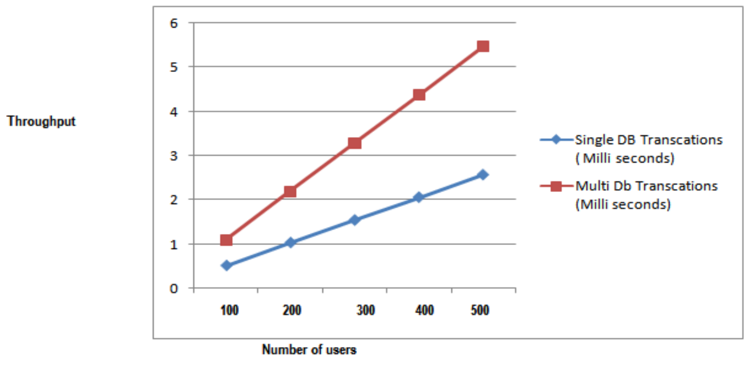

Figure 10. Throughput for Single DB and Multi DB (Warehouse=5)

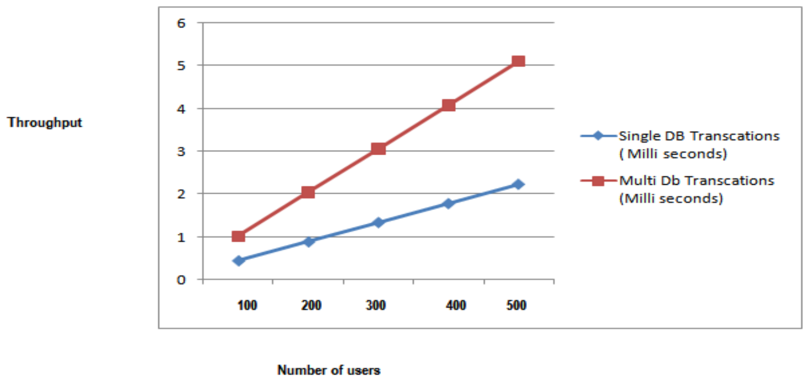

Figure 11. Throughput for Single DB and Multi DB (Warehouse=10)

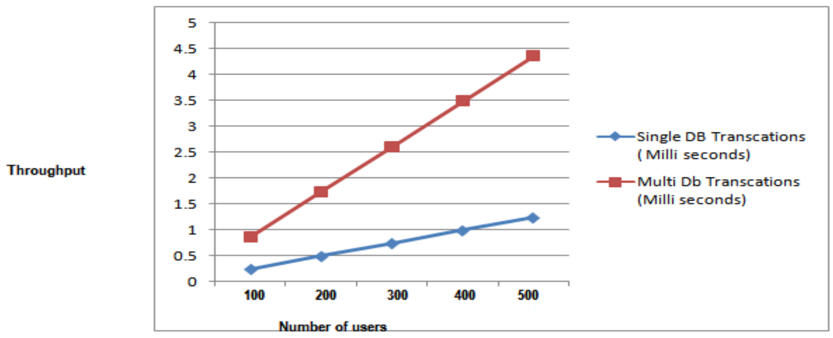

Figure 12. Throughput for Single DB and Multi DB (Warehouse=15)

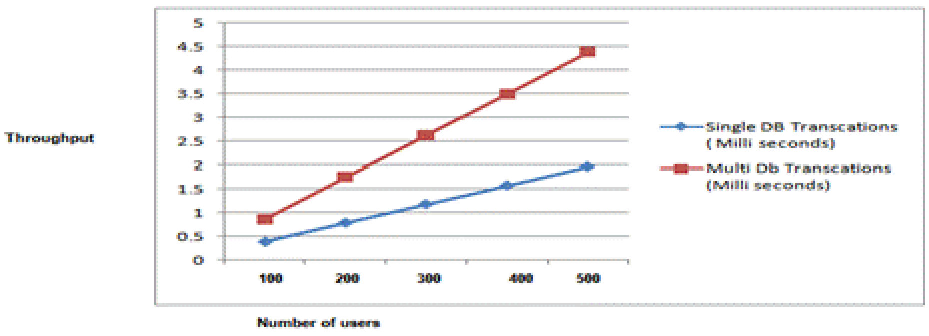

Figure 13. Throughput for Single DB and Multi DB (Warehouse=20)

6. CONCLUSION

The Incremental Repartitioning Technique for scalable OLTP applications is implemented in Couchdb to improve the scalability of the OLTP applications. It helps in gathering most frequently accessed items together. The industry benchmark TPC-C is used to stimulate the OLTP workload. Hyper Graph representation technique helps in reducing the graph size. Transactions are classified on the basis of the Warehouse id.Fuzzy C means clustering algorithm is used for making clusters based on the output of the Hyper Graph. Fuzzy rules are applied to get the balanced clusters. Frequently accessed items are kept together on one partition. So the number of the transactions is reduced.

The main goal of the Incremental Repartitioning technique is to convert the distributed transactions into the non-distributed transaction is obtained here. The response time in single DB is large as compared to response time in DT. The Time required for executing the transactions in DT is less. It is also observed that Incremental Repartitioning technique reduces the number of distributed transactions.

REFERENCES

[1] D.J.Dewitt, J.Gray, "Parallel Database systems: The future of high performance database systems", *ACM* (1992).

[2] C.Curino, E.Jones, Y.Zhang, S.Madden,"Schism: a workload-driven approach to database replication and partitioning", *Proc.VLDB Endow*.3 (1-2) (2010).

[3] A.Quamar, K.A.Kumar, A.Deshpande,"SWORD: Scalable Workload-aware data placement for transactional workloads", *Proceedings of the 16th International Conference on Extending Database Technology, ACM* (2013).

[4] Miguel Liroz- Gistau , Reza Akbarinia, Esther Pacitti, Fabio Porto, Patrickvalduriez, "Dynamic workload-based partitioning for large scale databases", *lirmm-00748549*,version-1-5 (Nov 2012).

[5] Xiaoyan Wang, Xu Fan, Jinchuan Chen,Xiaoyong Du,"Automatic Data Distribution in Large-scale OLTP applications", *International Journal Of Databases theory and Applications*,volume.7,No.4(2014).

[6] Alexandru Turcu,Roberto Palmieri,Binoy Ravindra,Sachin Hirve,"Automated Data Partitioning for Highly scalable and strongly consistent transactions", *IEEE transactions on parallel and distributed systems*,Volume-27,January (2016).

[7] Francisco Cruz, Francisco Maia, Rui Oliveira, Ricardo Vilaca, "Workload aware table splitting for NoSQL", *ACM,14 March* (2014).

[8] Swati Ahirrao, Rajesh Ingle, "Dynamic Workload Aware Partitioning in OLTP cloud data store", *JATIT LLS*,Volume.60 No.1,(Feb 2014).

[9] Shivanjali Kanase and Swati Ahirrao, " Graph Based Workload Driven Partitioning System for NoSQL Databases", *JATIT LLS*, Volume.88.No.1, (July 2016).

[10] Andrew Pavlo, Carlo Curino, and Stanley Zdonik, "Skew-aware automatic database partitioning in shared-nothing, parallel OLTP systems", *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. ACM*, No.149, 2012.

[11] Joarder Mohammad Mustafa kamal, Manzur Murshed, Mohamed Medhat Gaber,"Progressive Data stream mining and transaction classification for workload aware incremental database repartitioning" ,*IEEE*,(2014).

[12] Joarder Mohammad Mustafa kamal, Manzur Murshed, Rajkumar Buyya, "Workload aware Incremental Repartitioning Of Shared nothing Distributed databases for scalable cloud applications ",*IEEE*,(2014).