Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

An Efficient Model for Fragment Allocation in a Distributed Database System

Ezeugbor I.C[1], Ejiofor C.I[2], Onyesolu M.O[1], Okolo C.C[3]

[1] Department of Computer Science, Nnamdi Azikiwe University, Awka
[2] Department of Computer Science, Chukwuemeka Odumegwu Ojukwu University, Uli
[3] Electronic Development Institute, Fed Min of Sci and Tech, Awka Capital Territory

ABSTRACT

Various companies and establishments today are using distributed database system for daily business transactions in different domains. Some critical issues have been observed related to the complexity, maintenance, performance and communication cost of data in distributed data repository for query processing, according to the demand of end users from different locations. Distributed database helps to allocate data as fragmented, replicated and distributed over the intranet or Internet within and across an organization. In this paper, we developed an efficient model for fragment allocation in a distributed database system. These algorithms-genetic, clustering and fragmentation-were employed in allocating database fragments to individual sites and collating these distributed databases to one central store. Object oriented analysis and design methodology (OOADM) was employed in the systematic study and design of the system. Create, read, update and delete (CRUD) techniques, Bootstrap, cascading Style Sheet (CSS) and JavaScript were employed in the implementation of the system. MongoDB database engine was used to implement the database of the system. The result is a system with external fragmented database with capability of saving to a central database. This system can be used for every individual, organizations and establishments that has need to fragment database in a distributed database system for a maximum reduction of process time, for easy access of data and for complexity reduction.

Keywords: Efficient Model, Fragment, Allocation, Database System, Techniques

INTRODUCTION

The increasing complexity of information and communication technology (ICT) development in all facets of life, including the field of computer science has necessitated the urgent need for developing a model for fragment allocation in a distributed database system. The concern for this has shown a distributed database system as being imperative in achieving better efficiency and easy retrieval of information in establishments; this is particularly observable in the banking system in Nigeria. It is a known fact that every establishment has its own peculiar database, but there appears to be a necessity to develop an efficient model for fragment allocation in a distributed database system in order to be in tandem with the international best practices. Allocation is the process of assigning a function or an activity to a database. In today's financial institutions, most banks have their own database system but still need to be collated and run a fragmented central system for all banks irrespective of banking type or name. There is need for fragmentation because in a distributed database the cumbersomeness of use of data, the difficulty in accessing and manipulating data, reduction of performance speeds in accessing and the problem in data retrieval. An efficient model was developed for allocating fragments that are in a distributed database systems to one database using two techniques which are clustering technique and Genetic technique. Genetic technique comes into play as to identify the activities which are to be clustered. In order words, there are so many activities performed by banks in Nigeria but this work is limited to only transactions (credit and debit of account), and also payment of utility bills. These techniques came into use by first of all clustering the number of banks that will be in use in order to achieve this work, after which activities which the banks will perform are also clustered. Genetic technique comes into play as to identify the activities which are to be clustered. There is difficulty in accessing and manipulation of data in a distributed database, reduction of performance speed in accessing a distributed database and distributed database poses a problem in data retrieval, as such, introduction of fragmentation is very pertinent.

Review of Related works

Kahn and Hoque (2010) made a finding that saw the need for a single algorithm for both fragmentation and allocation which can be done simultaneously to be achieved. This was applied to be used for initial fragmentation problem of relational database for any distributed database system using horizontal technique. They further provided a solution which formulated a single algorithm for both fragmentation and allocation which can be done simultaneously. There was still an encountered problem where only an algorithm and design was formulated and developed, implementation was not done. There is still need for implementation by building robust software to take care of the fragmentation and allocation. Mahboubi and Darmont (2009) proposed a need for a technique that can enable horizontal fragmentation used in data warehouse. This was applied in predicate affinity for horizontal fragmentation in data warehouse. This worked on XML warehouse fragmentation where focus was on the initial horizontal fragmentation of dimensions' XML documents and exploits two more alternative algorithms. Their work ended on mathematical design of fragmentation on XML documents without any implementation. Cheng et al. (2002) used predicate affinity matrix as input by proposing genetic algorithm-based clustering approach. This approach treats horizontal fragmentation as a travelling salesman problem (TSP). The solution that was provided performed selection operation using the set of the grouped predicates, which are grouped according to the distances. There was still an encountered problem as none of the affinity-based horizontal fragmentation approaches takes into consideration data locality while clustering predicates. This is among many other works reviewed.

MATERIALS AND METHODS

Bootstrap was used as a framework to help design sites faster and easier. It includes HTML and Cascading Style Sheet (CSS) based design templates for typography, forms, buttons, tables, navigations, modals. It also gave support for JavaScript plugins. CSS was used for describing the presentation and design of web pages including colors, fonts and layouts. It was applied in order to enable the distinction between presentation and content, including colors, layouts and fonts. JavaScript is commonly used as a client side scripting language. It is used because of its dynamic nature, and it adds special effects on pages like rollover, and many types of graphics. It can load content into a document whenever the user requires it without reloading their entire page. Nodejs is built on chrome's JavaScript runtime for building fast and scalable network applications. Jquery was used to make much easier for JavaScript to be used at the sites. Jquery is a light weight, "write less, do more", JavaScript library. MongoDB database engine was used to implement a data store that provides high performance, high availability and automatic scaling. Npm, a node package manager was used to install packages locally into this project, specifically into the Node_modules folder. Clustering algorithm (Algorithm_Clustering) combined with genetic algorithm (Algorithm_Genetic) in allocating fragments to individual sites were employed. When efficiency and solution quality is more important, genetic algorithm is considered more attractive (Ishfaq et al., 2002). This combined techniques' aim is to overcome the problems associated with previous techniques, such as restrictions on the number of network sites, complexity, inefficient solutions etc. The system used two different algorithms to achieve the result in this work. Firstly, the network sites were clustered (using Algorithm_Clustering) that is grouping sites which have comparable and similar functions together. Clustering decision value (cdv) is used to determine whether or not a site can be grouped in a specific cluster or not. After clustering, fragments are allocated to the clusters that have transactions using the fragment. Secondly, at each cluster, genetic algorithm was used. The genetic algorithm searches the solution space for possible allocations, evaluating them using a developed allocation cost function and eventually finds the best allocation in which in most cases is the optimal allocation. The allocation cost function identifies the allocation status which is computed as a logical value for the comparison between the cost of remote access of the fragment to the cluster and the cost of allocating the fragment to the cluster. The system is out to improve data accessibility, security, speed of operation and data integrity of a distributed database system.

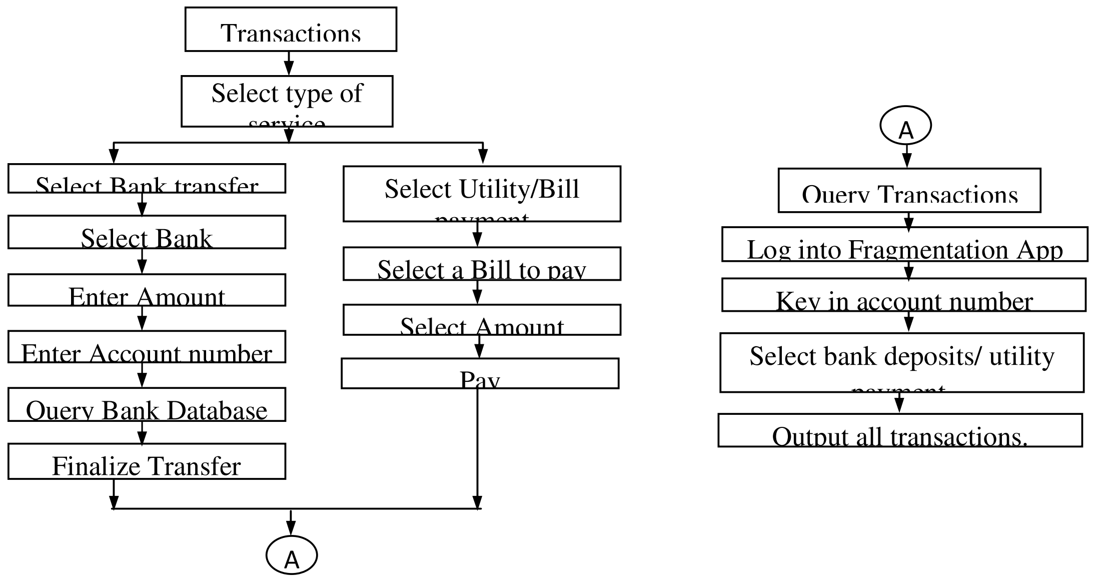


Fig 1: Information Flow Diagram Design Specification and Algorithms

To solve the problem of taking proper fragmentation decision at the initial stage of a distributed database, clustering algorithm (Algorithm_Clustering), genetic algorithm (Algorithm_Genetic and Figure 2) and create, read, update and delete (CRUD) technique were adopted as depicted in Figure 3.

Algorithm_Clustering

```
CR: Clustering Range
NS: Number of sites in the distributed database system network
Output: CSM: Clusters Set Matrix
Step 1: Set 1 to i
Step 2: Do steps (3 - 12) until i > NS
Step 3: Set 1 to j
          Set 0 to k: Set 0 to Sum
          Set 0 to Average:Set 0 to clusters matrix CM
Step 4: Do steps (5 - 10) until j > NS
Step 5: If i ≠ j AND CC($S_i$, $S_j$) <= CR,
          go to step (6)
        Else,
          go to step (7)
Step 6: Set 1 to the CM($S_i$, $S_j$) and CM($S_j$, $S_i$) in the clusters matrix
        Add CC($S_i$, $S_j$) to sum Add 1 to k
        Go to step 8
Step 7: Set 0 to the CM($S_i$, $S_j$) and CM($S_j$, $S_i$) in the clusters matrix
Step 8: End IF
Step 9: Add 1 to j
Step 10: Loop
Step 11: Average = Sum/k Average(i) = Average Add 1 to i
Step12: Loop
Step 13: Set 1 to m
Step 14: Do steps (15 - 36) until m > NS
Step 15: Set 1 to q Set 0 to Minaverage Set 0 to Minrow
Step 16: Do steps (17 - 20) until q > NS or Minaverage >0
Step 17: If Average(q) > 0 Then
                Minaverage = Average(q)
         Else
                Go to Step 18
Step 18: End If
Step 19: Add 1 to q
Step 20: Loop
Step 21: If Minaverage = 0 Then
                Set site number to a new cluster
         Else
                Go to Step 22
Step 22: End If
Step 23: Set 1 to p
Step 24: Do steps (25 - 28) until p > NS
Step 25: If Average(p) > 0 AND Average(p) < Minaverage Then
                Minaverage = Average(p)
                Minrow = p
Step 26: End IF
Step 27: Add 1 to p
Step 28: Loop
Step 29: Set 1 to a
Step 30: Do steps (31 - 34) until a > NS
Step 31: If CM($S_{minrow}$, $S_a$) = 1 Then
                Set 1 to CSM($S_{minrow}$, $S_a$)
                CM($S_{minrow}$, $S_a$) = 0 Step 32:
         End IF
Step 33: Add 1 to a
Step 34: Loop
Step 35: Add 1 to m
Step 36: Loop
Step37:Stop
Input: Tmax: number of transactions issued in the database
Fmax: number of the disjoint fragments used for allocation
Cmax: number of clusters in the distributed database system
```

Algorithm_Genetic

1. Generate an initial population, repeating random strings of fixed size.
2. Do the selection, reproduction, crossover and mutation operations of the all population.
3. Replace the old population with the new one.
4. Repeat Steps (2, 3) until number of iterations is finished.
5. Display the best answer found (which has the best-fitness).

Algorithm_Fragment Allocation

```
Input: K: Number of the last fragment
Rmax: Number of database relations
Nmax: Number of fragments in each relation

Step 1: Set 0 to K
Step 2: Set 1 to R
Step 3: Do steps (4 - 21) until R > Rmax
Step 4: Set 1 to I
Step 5: Do steps (6 - 20) until I > Nmax
Step 6: Set 1 to J
Step 7: Do steps (8-18) until J > Nmax
Step 8:If I ≠ J and ∃ Si ,Sj Є SR
                goto step (9)
        Else Add 1 to J,
                go to step (18)
Step 9: If Si ∩ Sj ≠ Ø
                do steps (10)-(17)
        Else
                Add 1 to J and go to step (19)
Step 10: Add 1 to K
Step 11: Create new fragment Fk = Si ∩ Sj and add it to F
Step 12: Create new fragment Fk+1 = Si - Fk and add it to F
Step 13: Create new fragment Fk+2 = Sj - Fk and add it to F
Step 14: Delete Si
Step 15: Delete Sj
Step 16: Set Nmax + 1 to J
Step 17: End IF
Step 18: End IF
Step 19: Loop
Step 20: Add 1 to I Step 21: Loop
Step 22: Set 1 to I
Step 23 Do steps (24 - 35) until I > Nmax
Step 24: Set 1 to J
Step 25: Do steps (26 - 33) until J > Nmax
Step26:If I ≠ J and ∃ Si,Sj Є SR
                goto step(27)
        Else Add 1 to J,
                go to step (33)
Step 27: If Si ∩ Sj = Ø do steps (28)-(33)
Step 28: Add 1 to K
Step 29: Create new fragment Fk = Rj - UF
Step 30: End IF
Step31:If Fk ≠ Ø Add Fk to the set of F
Step 32: End IF
Step 33: Loop
Step 34: Add 1 to I Step 35: Loop
Step 36: Set 1 to I
Step 37: Do steps (38 - 53) until I > F
Step 38: Set 1 to J
Step 39: Do steps (40 - 51) until J > F
Step 40:If I ≠ J and ∃ Fi,Fj Є FR goto step (41) Else, Add 1 to J and go to step (50)
Step 41: If Fi ∩ Fj ≠ Ø do steps (42)-(49) Else, Add 1 to J and go to step (49)
Step 42: Add 1 to K
Step 43: Create new fragment Fk = Fi ∩ Fj and add it to F
Step 44: Create new fragment Fk+1 = Fi - Fk and add it to F
Step 45: Create new fragment Fk+2 = Fj - Fk and add it to F
Step 46: Delete Fi
Step 47: Delete Fj
Step 48: Set F + 1 to J
Step 49: End IF
Step 50: End IF
Step 51: Loop
Step 52: Add 1 to I
Step 53: Loop
Step 54: Add 1 to R
Step 55: Loop
```

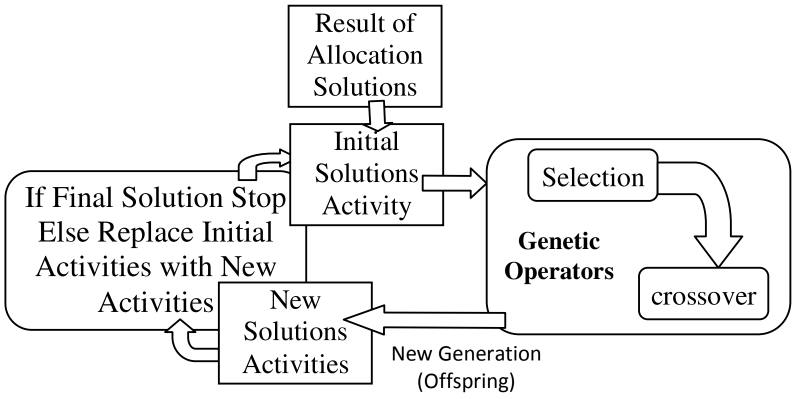

Result of Allocation Solutions
Initial Solutions Activity
Selection
Genetic Operators
crossover
If Final Solution Stop Else Replace Initial Activities with New Activities
New Solutions Activities
New Generation (Offspring)

Fig 2: The Basic Genetic Steps

RESULTS AND DISCUSSION

The results of this work are shown in Figure 2 to figure 8. The system is designed to display all the transactions made by an individual within a specified period of time by bringing the transactions closer for easy access (Figure 4). It also shows the beneficiary bank, transaction mode, date and time, amount, sender and receiver's account number on the fragmented database. The use of genetic technique as a model supports and promotes efficiency of the system in the sense that the data are properly arranged on a row, also displaying the transactions made according to the order of transfer. The benefits of this model to the users are reduction in communication cost by reducing the number of servers, easy and fast retrieval of data and closeness of data to the user. This work was achieved by clustering three banks, allocating activities to the banks i.e., transaction of funds and payment of utility bills.

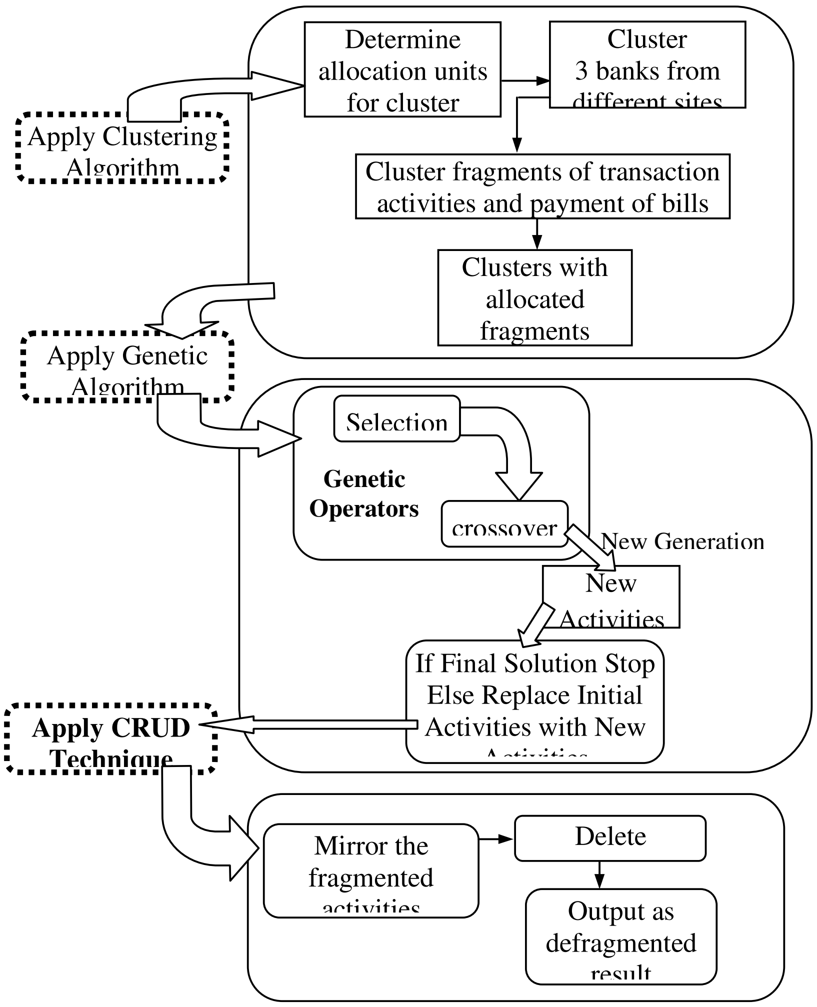

Determine allocation units for cluster
Cluster 3 banks from different sites
Apply Clustering Algorithm
Cluster fragments of transaction activities and payment of bills
Clusters with allocated fragments
Apply Genetic Algorithm
Selection
Genetic Operators
crossover
New Generation
New Activities
If Final Solution Stop Else Replace Initial Activities with New Activities
Apply CRUD Technique
Mirror the fragmented activities
Delete
Output as defragmented result

Fig 3: Fragment Allocation Techniques

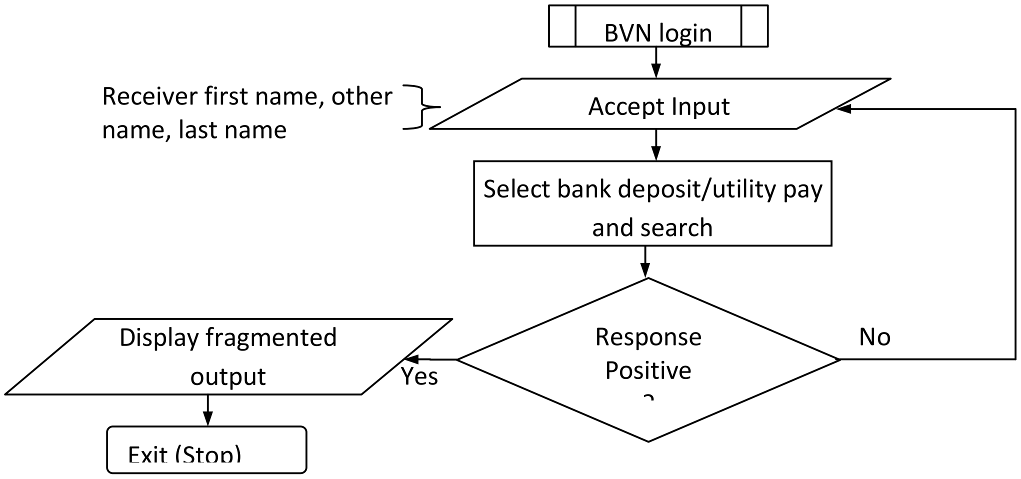

BVN login
Receiver first name, other name, last name
Accept Input
Select bank deposit/utility pay and search
Response Positive ?
No
Yes
Display fragmented output
Exit (Stop)

Fig 4: Flowchart on fragmented system

Admin Menu

This is made up of Open Account, Search Account and Log Out (Figure 5). Open Account provides the admin with an interface that enables the admin open an account for a new customer taking the customers information from the keyboard (Figure 6) and saving to the database for future use and after which a unique account number is generated automatically by the system. Search Account is very necessary to navigate from one account to another considering the fact that in an average bank, a million plus customer accounts may all be linked to the same database, a search with name or account number allows for ease of transaction after the customers detail and statement is retrieved from the database. There are other functions provided on this level, update of an existing customers account, view and delete accounts in the event "the customer chooses to close their account". Logout the admin logs out after the admin is through with whatever job for the day.- Every admin in has a unique login ID, with a login ID the system keeps record of user's input and how long a user was active. This also makes it possible for users to be held account of any theft or wrong input entered into the system, although there is a counter security measure which in this case is a session timer which times out if system is idle for a minute.

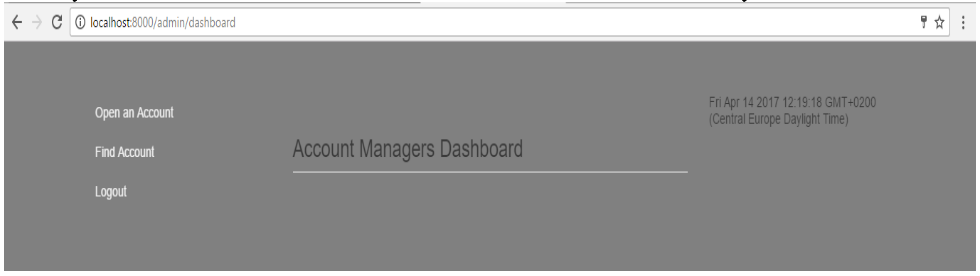

localhost:8000/admin/dashboard
Open an Account
Find Account
Logout
Account Managers Dashboard
Fri Apr 14 2017 12:19:18 GMT+0200 (Central Europe Daylight Time)

Fig 5: Admin Main Menu

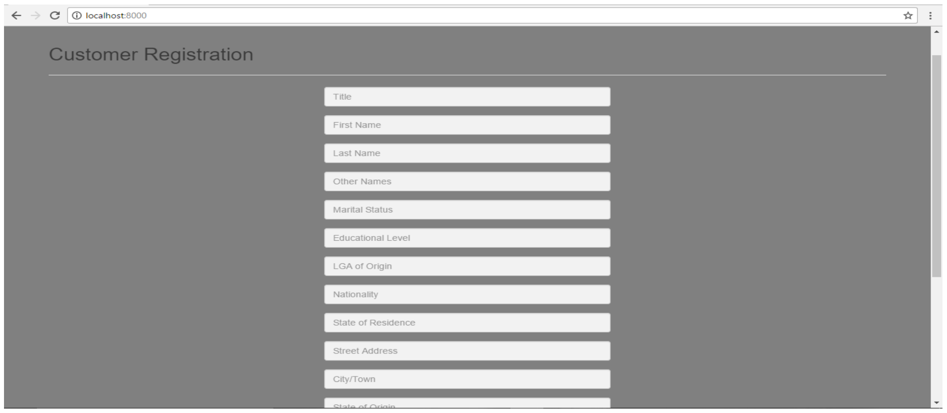


Fig 6: Customer Registration Interface

User Menu

User menu is made up of Account Balance, Account Activity, Account Statement, Utility/Bills Payment and Bank Transfer (Figure 7). Account Balance provides a user with the interface that contains details of all accounts connected by BVN. The user available balance in all connected accounts will be viewed without access to printing. Account Activity enables a user see the last five activities or transactions. Account Statement reflects all transaction on the account during a stipulated period of time, the range in period is often decided by the user of the account. The user has an option of printing this statement or downloading and saving on their personal device for viewing without network access at their leisure. Utility/Bills Payment (Figure 8) - A user is expected to make other transactions aside view of statement, checking of account balance and so on. This interface enables a user the opportunity to pay for light bill, water bill and purchase of data or recharge cards with ease from any of the bank accounts connected with the help of BVN fragmentation. Bank Transfer allows a user the opportunity to transfer funds from any of the user's his registered bank accounts to other accounts either to business associates or family.

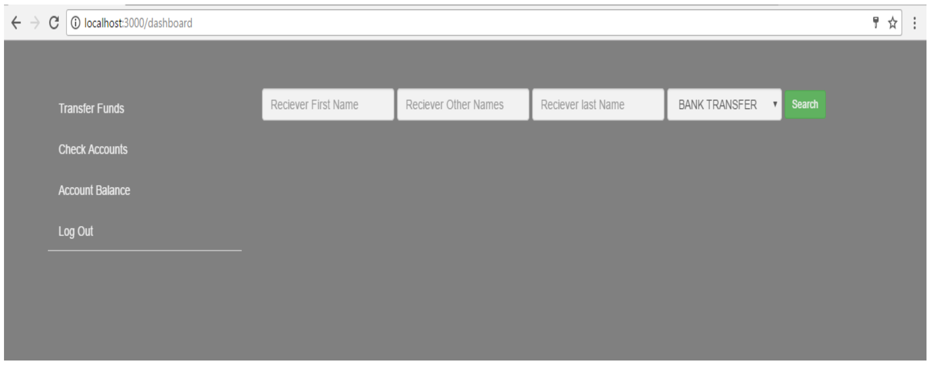
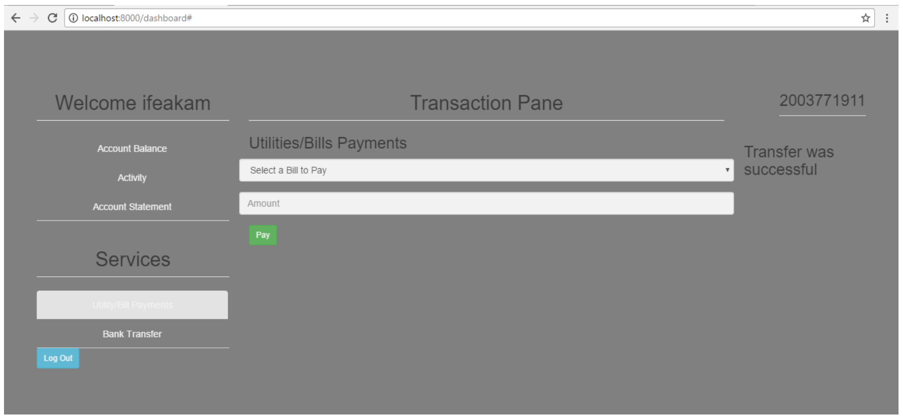
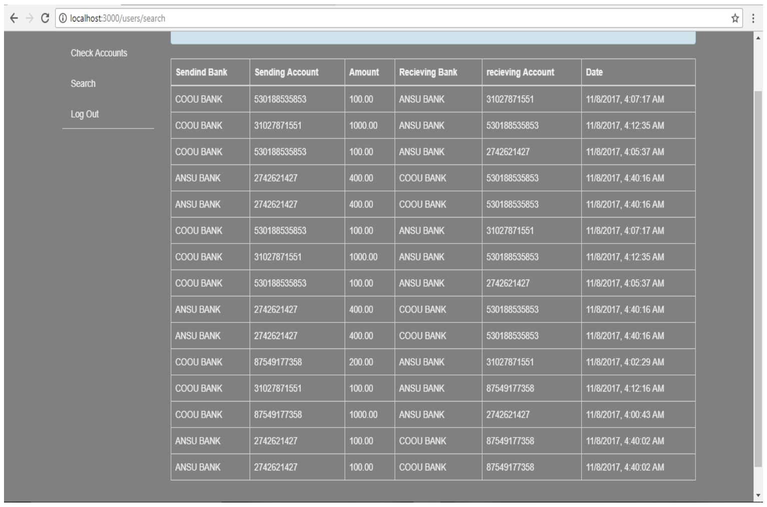


Fig 7: User Main Menu



Transfer was successful

Fig 8: Utility/Bills Payment Interface



Sendind Bank	Sending Account	Amount	Recieving Bank	recieving Account	Date
COOU BANK	530188535853	100.00	ANSU BANK	31027871551	11/8/2017, 4:07:17 AM
COOU BANK	31027871551	1000.00	ANSU BANK	530188535853	11/8/2017, 4:12:35 AM
COOU BANK	530188535853	100.00	ANSU BANK	2742621427	11/8/2017, 4:05:37 AM
ANSU BANK	2742621427	400.00	COOU BANK	530188535853	11/8/2017, 4:40:16 AM
ANSU BANK	2742621427	400.00	COOU BANK	530188535853	11/8/2017, 4:40:16 AM
COOU BANK	530188535853	100.00	ANSU BANK	31027871551	11/8/2017, 4:07:17 AM
COOU BANK	31027871551	1000.00	ANSU BANK	530188535853	11/8/2017, 4:12:35 AM
COOU BANK	530188535853	100.00	ANSU BANK	2742621427	11/8/2017, 4:05:37 AM
ANSU BANK	2742621427	400.00	COOU BANK	530188535853	11/8/2017, 4:40:16 AM
ANSU BANK	2742621427	400.00	COOU BANK	530188535853	11/8/2017, 4:40:16 AM
COOU BANK	87549177358	200.00	ANSU BANK	31027871551	11/8/2017, 4:02:29 AM
COOU BANK	31027871551	100.00	ANSU BANK	87549177358	11/8/2017, 4:12:16 AM
COOU BANK	87549177358	1000.00	ANSU BANK	2742621427	11/8/2017, 4:00:43 AM
ANSU BANK	2742621427	100.00	COOU BANK	87549177358	11/8/2017, 4:40:02 AM
ANSU BANK	2742621427	100.00	COOU BANK	87549177358	11/8/2017, 4:40:02 AM

Fig 9: Output Format of The Fragmented System

SUMMARY

A novel model for fragment allocation in a distributed database system was developed and implemented. The new system was developed to create a central financial technology solution that allows users carry out financial transactions from all their bank accounts using database fragmentation to separate the banks and their operations. It tackled the problem of delay in processing data at the banks. It provided a unified system for all the banks for easy access. The model was created to add fragments to DDBMS for easy access.

CONCLUSION

A new approach to improve fragment allocation in distributed database system was proposed in this work. This approach is integrating of two techniques namely database fragmentation and fragment allocation. These techniques are developed to avoid drawbacks of database fragmentation and data allocation like data redundancy and complexity of data redistribution problem. In addition, a novelty of our approach is to satisfy a certain level of data availability and consistency. The result obtained in this work showed that our approach significantly improved performance requirement satisfaction in distributed systems.The strength of this system include reduced data transfer between sites, increased security, event logging is easier as it improves availability, it can control transactions hence lead to safer transactions, efficiency in performing transactions, it is not greatly influenced by errors in assumptions about the distribution of sample errors, data that are not required by local applications are not stored locally and it is easy to implement in any organization. Keeping record of all transactions made from this window for accountability. The system must be informative, robust, responsive, user-friendly and secure. System should be designed to allow possible future expansion.

REFERENCES

Cheng, J.M. et al. (1984). IBM database 2 performance: Design, Implementation and tuning". IBM systems J., 23(2): 189-210.503.

Cheng, C. H., Lee, W. K. and Zaverucha, G. (2002). A genetic design algorithm-based clustering approach for database partitioning, IEEE Transactions on systems, man and cybernetics,(32) 215-230, 2002.

Cheng, C. H., Lee, W. K. and Wong, K. F. (2002). A genetic algorithm-based clustering approach for database partitioning". IEEE Transactions on systems, man and cybernetics-part C: Applications and Reviews, (32) 3.

Chen, J. (2012). Distributed Database: Fragmentation and Allocation, Journal of Data Miningand Knowledge Discovery ISSN: (3)2229-6662 & ISSN 2229-6670, 2012.

Daudpota, N.H, (1998). Five steps to construct a model of data allocation for distributed database systems. Journal of Intelligent Information Systems: Integrating Artificial Intelligence and Database Technologies. (11) 2, 153-68.

Huang, Y. F. and Chen, J. (2001). Fragment allocation in distributed database design. Journal of Information of Science and Engineering, (17) 491-506.

Kahn, S.I. and Hoque, A.S.M.L. (2010). A new technique for database fragmentation in distributed systems, International Journal of Computer Applications", (5)9, 20-24 http://dx.doi.org/10.5120/940-1318

Mahboubi, H. and Darmont, J. (2009). Fragmenting very large XML data warehouse via K-means clustering algorithm, International Journal of Business Intelligence and Data Mining. (4), 3-4.

Ozsu, M.T. and Valduriez, P. (1997). Distributed and parallel database systems, In tucker, A., editor, handbook of Computer science and Engineering, 1093-1111. CRC press. 38.

Ozsu, M.T. and Valduriez, P. (1999). Principles of distributed database systems: Third edition, DOI 10.1007/978-1-4419-8834-8, c springer science + Buisness media.

Ozsu, M.T. and Valduriez, P. (1999). Database fragmentation and allocation: An integrated methodology and case study. Article in IEEE Transactions on systems man and cybernetics- A part A system and humans 28(3):288-305.

Peddemors, A.J.H and Hertzberger, L.O. (1999). A high performance distributed database system for enhanced internet services, Future-Generation-Computer-Systems.(15) 3; 407-15.