Addis Ababa University College of Natural and Computational Sciences Department of Computer Science

CoSc6201– Distributed Database Systems

Vertical Fragmentation in Distributed Database System: A Comparative Study of Clustering Algorithms

By Yoseph Mulugeta
GSE/2388/14

To: Ayalew Belay (PhD.)

June 2022

Acronyms

DDBS	Distributed Database System
VF	Vertical Fragmentation
HF	Horizontal Fragmentation
BEA	Bond Energy Algorithm
MBEA	Modified Bond Energy Algorithm
GBVPA	Graph Based Vertical Partitioning Algorithm

CONTENTS

Introduction

A distributed database is a single logical database that is spread physically across computers in multiple locations that are connected by a data communications network [1]. Major task of Distributed database system is fragmentation that permits a divide and conquer management technique of giant database to small portion of it which are called fragments. These fragments are logical units of data stored in various sites. The aim is to promote operations in efficient and effective way during execution takes place in database. Fragments kept in different sites and communicated thought computer network. In addition, all attributes or relations of access table may not be needed so the distributed database design must eliminate these attributes during the execution. We use fragmentation because unit of distribution using fragments is better than unit of distribution using relations [2].

DDBS is a single logical database system which is placed in different sites which are geographically separated but communicates to one another by a computer network. A powerful distributed database system should be remained in systematic, resilient, and structured way. The performance and efficiency of a distributed database design largely depend on the fragmentation of global relations. In a Distributed Database Design, we have three types of fragmentations Vertical Fragmentation (VF), Horizontal Fragmentation (HF) and Hybrid Fragmentation (combination of VF and HF). HF is designed to partition relations into different tuples or instances whereas, VF is designed in a way in which attributes or columns of the database are fragmented and grouped into small relations. Vertical fragments are created by dividing a global relation R on its attributes by applying the project operator:

$$Rj = \Pi \{Aj\}, \text{key } R, \text{ where } 1 \leq j \leq m \ [3]$$

VF is the process of subdividing the attributes of a relation to generate fragments [3] during query. Query is a standard technique in which users of the application request information and retrieve the necessary data by binding the correspondent attributes distributed across many tables. Database Fragmentation in VF is implemented by splitting large table into smaller tables, queries can retrieve specific attributes as small fractions of data in faster way results less data to scan, parse and validate.

A centralized database system can't be accessed equally to all terminals in a distributed system and in another term, if the data is stored in logically fragmented lightweight version near to the usage site, it increases the execution process. Fragmentation primes easy to implement the security and privacy of the database. Another advantage of fragmentation is it controls correctness of partitioning using three metrics completeness, reconstruction (for HV Union operation is used, For VF Join operation is used) and Disjoint-ness.The process of database partitioning will be performed using a vertical fragmentation method based on the calculation results of Bond Energy algorithm (BEA) and Graph-Based Vertical Partitioning algorithm (GBVP) [4].

In this proposal, GBVP and BEA algorithms are involved and a performance comparison study will take place between these two most known popular algorithms that assess the performance of three algorithms using Bond Energy Algorithm (BEA) , Modified Bond Energy Algorithm (MBEA) and Graph Based Vertical Partitioning Algorithm (GBVPA).

A Taxpayer database is selected for this study, the necessary stages such as query selection, attribute selection, affinity calculation process will take place. BEA, MBEA and GBVPA will take identical affinity matrix as input clustered affinity matrix CA which is a permutation of AA will be our output for each of algorithms. The results of these algorithms compare to one another in terms two metrics, query *execution time* and *access data cost*.

Problem Statement

The database system can be either centralized or distributed database. It is more complicated than HF because it can be implemented in many alternative perspectives. VF relation R produces fragments R1, R2, R3, …. Rn, with mandatory options that all fragmented relations are subsets of R and primary key of R.

let’s assume B = Bell Number (by Niamir), m = member attribute and ’m’ is not primary key in relation R, we can calculate and find the total number of fragments and it is equal to B(m). Considering these in mind creates two approaches of fragmentation splitting, from perspectives of attribute and grouping, from perspectives of relation R.

The necessary information in fragmentation and allocation should be fulfilled, database Information, application Information, communication network Information and computer System Information should all be relevant. If one of these information is affected it requires strong methods, techniques, and algorithms to solve. Since VF is produced by assigning related attributes into one fragment, these attributes will be operated together. Attributes that are grouped in the same fragmentation should cooperate with one another and the study of this cooperation is measured by affinity attribute (AA).

As we discussed, one of the required information in fragmentation is applications information to access the required information. Applications usually ask information by using queries from different sites interims of queries in both centralized and distributed database systems. Vertical fragmentation techniques use the Attribute Usage Matrix (AUM) [5].

> Let Q = query, and Q = {q1, q2, q3, . . . , qn}
> Let R = relation, let = Attribute, R (A1, A2, A3, . . .,AN) and for each of the query(qi) and attribute(Aj) the attribute usage of value donated by use(qi,Aj) is defined
> Use (qi, Aj) = 1 if attribute Aj is referenced by query qi otherwise use(qi, Aj) = 0 [5].

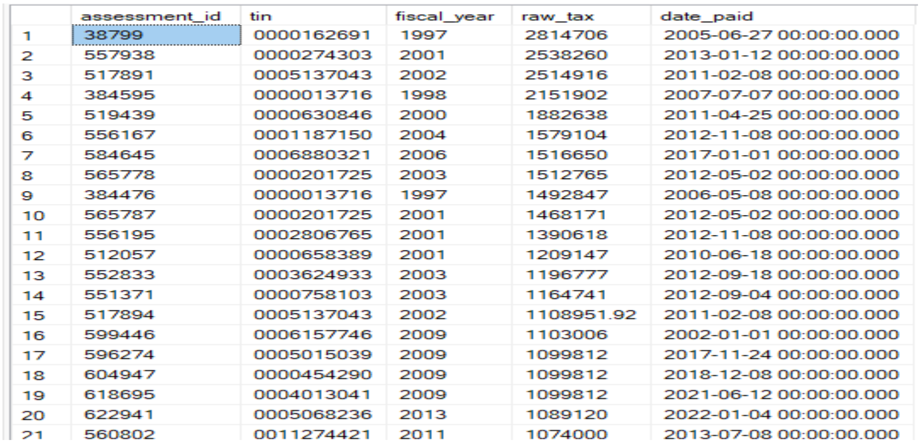
VF is applicable based on defined subset of attributes of the original relation and it’s functional to improve the business performance, keep track of operational records, secure sensitive information, to store data, etc. When the data queried from different physically distributed system the access speed will be low and we need to adjust latency. Reconstruction is one of the metrics in fragmentation, however if the fragmentation is recursive this metrics acquires a well-designed method, algorithm, and techniques. Hardware or software failure is another issue to lose the required data and it leads ineffectiveness of the database system.

Experimentation and Evaluation

Example: The following example shows a Taxpayer database that contains the following five attributes that can be queried by different applications.

```
SELECT assessment_id, tin, fiscal_year, raw_tax, date_paid
 FROM [TaxPayer].[dbo].[Assessment]
```

The above SQL Script retrieves information for 189,000 records in Taxpayer Database from Assessment relation that can be quired depending on applications VF specification which is applicable.



	assessment_id	tin	fiscal_year	raw_tax	date_paid
1	38799	0000162691	1997	2814706	2005-06-27 00:00:00.000
2	557938	0000274303	2001	2538260	2013-01-12 00:00:00.000
3	517891	0005137043	2002	2514916	2011-02-08 00:00:00.000
4	384595	0000013716	1998	2151902	2007-07-07 00:00:00.000
5	519439	0000630846	2000	1882638	2011-04-25 00:00:00.000
6	556167	0001187150	2004	1579104	2012-11-08 00:00:00.000
7	584645	0006880321	2006	1516650	2017-01-01 00:00:00.000
8	565778	0000201725	2003	1512765	2012-05-02 00:00:00.000
9	384476	0000013716	1997	1492847	2006-05-08 00:00:00.000
10	565787	0000201725	2001	1468171	2012-05-02 00:00:00.000
11	556195	0002806765	2001	1390618	2012-11-08 00:00:00.000
12	512057	0000658389	2001	1209147	2010-06-18 00:00:00.000
13	552833	0003624933	2003	1196777	2012-09-18 00:00:00.000
14	551371	0000758103	2003	1164741	2012-09-04 00:00:00.000
15	517894	0005137043	2002	1108951.92	2011-02-08 00:00:00.000
16	599446	0006157746	2009	1103006	2002-01-01 00:00:00.000
17	596274	0005015039	2009	1099812	2017-11-24 00:00:00.000
18	604947	0000454290	2009	1099812	2018-12-08 00:00:00.000
19	618695	0004013041	2009	1099812	2021-06-12 00:00:00.000
20	622941	0005068236	2013	1089120	2022-01-04 00:00:00.000
21	560802	0011274421	2011	1074000	2013-07-08 00:00:00.000

Figure 1: Assessment information of taxpayer from assessment relation in Taxpayer Database

The following example shows information about Registered Taxpayers a revenues Bureau and let's consider our applications are defined to run on this relation.

```
q1:     SELECT raw_tax
        FROM [TaxPayer].[dbo].[Assessment]
        where tin is not null
q2:     SELECT fiscal_year, raw_tax
        FROM [TaxPayer].[dbo].[Assessment]
q3:     SELECT assessment_id
        FROM  [TaxPayer].[dbo].[Assessment]
        WHERE  raw_tax > 0
q3:     SELECT fiscal_year
        FROM [TaxPayer].[dbo].[Assessment]
        WHERE  raw_tax > 0
```

VF techniques use the Attribute Usage Matrix as we discussed earlier so let's start from it then and finally lets calculate attribute Affinity Matrix.

Let A1 = raw_tax, A2 = tin, A3 = fiscal_year, A4 = assessment_id,

	A1	**A2**	**A3**	**A4**
q1	1	1	0	0
q2	1	0	1	0
q3	1	0	0	1
q4	0	0	1	1

Attribute Usage Matrix

	S1	**S2**	**S3**
q1	7	11	0
q2	12	16	19
q3	6	0	0
q4	0	3	21

application frequencies

	A1	**A2**	**A3**	**A4**
A1	71	18	47	6
A2	18	18	0	0
A3	47	0	71	24
A4	6	0	24	30

Attribute Affinity Matrix

Designing VF’s primary aim is to identify interrelated attributes and group these attributes depending on attribute affinity (AA) values. Therefore, as we calculate the affinity values in the above example, the next step is comparing the Clustering algorithms used are to solve these issues and we will review three types of VF ***clustering algorithms.***

Bond Energy Algorithm:

In bond energy algorithm, similarity between two attributes is calculated using affinity measure [6]. The algorithm has input of attribute affinity matrix, and the result is clustered affinity matrix.

```
Input: AA: attribute affinity matrix
Output: CA: clustered affinity matrix
begin
{initialize; remember that AA is an n×n matrix}
CA(•,1) ← AA(•,1) ;
CA(•,2) ← AA(•,2) ;
index ← 3 ;
while index ≤ n do {choose the “best” location for attribute AAindex}
for i from 1 to index−1 by 1 do calculate cont(Ai−1,Aindex,Ai) ;
calculate cont(Aindex−1,Aindex,Aindex+1) ; {boundary condition}
loc ← placement given by maximum cont value ;
for j from index to loc by −1 do
CA(•, j) ← CA(•, j −1) {shuffle the two matrices}
CA(•,loc) ← AA(•,index) ;
index ← index+1
order the rows according to the relative ordering of columns
end
```

Bond Energy Algorithm will determine the positions of every entity by using global affinity measure.

$$Cont(Ai ,Ak , Aj) = 2bond(Ai\ Ak)+2bond(Ak ,Aj)-2bond(Ai , Aj)$$

	A1	**A2**	**A3**	**A4**
A1	71	18	47	6
A2	18	18	0	0
A3	47	0	71	24
A4	6	0	24	30

	A1	**A2**	**A3**	**A4**
A1	71	18	47	6
A2	18	18	0	0
A3	47	0	71	24
A4	6	0	24	30

	A1	**A2**	**A3**	**A4**
A1	71	18	47	6
A2	18	18	0	0
A3	47	0	71	24
A4	6	0	24	30

Calculation of the Clustered Affinity (CA) Matrix

Modified Bond Energy Algorithm:

In modified version of MBEA, two attributes have concurrent occurrence in a query then its similarity is considered and for the same query concurrent absence of the attributes could be considered as a weighted measure of similarity. Attribute Affinity Measure is modified and is denoted as Sij.

$$Sij = n11 + n00/n11 + n00 + w1(n01 + n10)$$

This formula gives the match between two attributes directly. Here n00 and n11 show number of simultaneous presence and absence of attributes. In denominator, n10 and n11 represent that only one attribute is accessed by the query at a time. Weights are assigned to dissimilar pairs based on their contribution to the similarity. In this approach less similar attributes are separated correctly.

Graphical Base Partitioning Algorithm:

Graphs are ubiquitous [7] in engineering sciences because they prove to be a flexible model in the modeling of various complex phenomena. This algorithm takes a parameter, affinity matrix as input however it is represented by edge to form a linear span tree. The affinity attributes are vertices with edge model and the interaction or relation is laid between affinity attributes.

General Objective

The overall objective of this study is to propose and determine a performance comparative of vertical fragmentation in three clustering algorithms BEA, MBEA and GBVPA.

Specific Objective

The specific objectives of this study are:

- Review the approaches, methods and techniques used in distributed system clustering algorithms.
- A Taxpayer database will be used and five(5) relations of the out of forty seven (47) relations are selected for our study. Customers, Business, Assessment, Payment, and Clearance classes are the selected five classes.
- 20 to 25 script queries will be constructed in purpose of VF.
- Design and develop a prototype of the proposed model.
- Evaluate the comparative study of the three clustering algorithms

Methodology

The methodology applied to achieve the general and specific objectives of this study are presented as follows

Literature Review

We will review literature, including conference papers, journal articles, and books about reducing access latency of the distributed system.

Testing and Evaluation

The proposed protocol will be evaluated based on different parameters. The results of these algorithms compare to one another in terms two metrics, *query execution time* and *access data cost*.

Significance

The importance of the proposed system is to perform evaluation for BEA, MBEA and GBVPA then determine the best clustering algorithms in distributed database system.

References

[1] Asma H. Al-Sanhani, Amira Hamdan, Ali B. Al-Thaher, Ali Al-Dahoud **“**A comparative Analysis of Data Fragmentation in Distributed Databas**”** ,2017 8th International Conference on Information Technology (ICIT)

[2] Dr. Ayalew Belay, Distiributed Database Sysytem Lecture class in **“oral discussion”** ,chapter 3: Distributed Database Design, May 04 2022.

[3] L. Bellatreche, K. Karlapalem, and A. Simonet, “Horizontal class partitioning in object-oriented databases,” in proceedings of the 8th International Conference on Database and Expert Systems Applications (DEXA’97), September 1997, pp. 58–67, Lecture Notes in Computer Science 1308.

[4] H. Rahimi, F. Parand, and D. Riahi, “Hierarchical simultaneous vertical fragmentation and allocation using modified Bond Energy Algorithm in distributed databases,” Appl. Comput. INFORMATICS, 2015.

[5] Farhi Marir; Yahiya Najjar; Mahmoud Y. AlFaress; Hassan I. Abdalla, “ An enhanced grouping algorithm for vertical partitioning problem in DDBs”, IEEE Xplore: 25 February 2008

[6] Hossein Rahimi, Fereshteh-Azadi Parand, Davoud Riahi, (2016) “Hierarchical Simultaneous Vertical Fragmentation and Allocation using Modified Bond Energy Algorithm in Distributed Databases”, Applied Computing and Informatics, Saudi Computer Society, King Saud University, pp. 1-7.

[7] Danai K, Christos F. Individual and collective graph mining: principles, algorithms, and applications. Synth Lect Data Mining Knowl Discov. 2017;9:2.