New Algorithms for Fragmentation, Allocation and Redistribution in DDBSs

خوارزميات جديده لتجزئة العلاقات وتوزيعها واعادة توزيعها في قواعد البيانات الموزعه

BY

Ali Abdullah Amer
ID No: 429108105

Supervisor

Dr. Hassan Ismail Abdalla

A thesis proposal submitted in partial fulfillment of the requirements for the degree of Master in Computer Science

Submitted To

Department of Computer Science, College of Computer & Information Sciences, King saud University

OCT 2011

الملخص:

تصميم قواعد البيانات الموزعة يعالج في الأساس مشاكل تجزئة العلاقات وتوزيعها وإمكانية استخراج نسخ مكرره منها. والهدف الاساسي من ذالك التوزيع هو البحث عن طريقة تزيد من موثوقية نظام قاعدة البيانات الموزعة وتعمل على تطوير أدائه. و يعتبر هذان العاملان أساسيان لتصميم قواعد البيانات الموزعة ويبرران الأهمية الكبيرة التي تحظى بها قواعد البيانات الموزعة في البحث العلمي اليوم.

في هذا العمل تم دراسة قواعد البيانات الموزعه في اطار قواعد البيانات العلائقية . العديد من طرق التجزئة والتوزيع واعادة التوزيع المختلفة تم طرحها للوصول للحل الامثل او قريب من الامثل . في الحقيقة تم استخدام تقنية لتقسيم العلاقات وتوزيع تلك الاجزاء في ان واحد . يمكن تطبيق تلك التقنية في مراحل التقسيم الاولى او المتاخرة من تصميم قواعد البيانات الموزعة، وعلاوة على ذالك تم بحث مشكلة إعادة التوزيع وطرح حلول لها بطرق ديناميكية تتناسب مع البيئات المتغيرة لقواعد البيانات الموزعة.

تم طرح خوارزميات جديده للتجزئة الافقية والعمودية والتوزيع في هذا البحث وايضا تم اقتراح خوارزميات جديده لاعادة توزيع اجزاء العلاقات في قواعد البيانات الموزعه بحيث تمكن كل موقع من الاستقلالية في اتخاذ القرار لاعادة التوزيع للاجزاء المخزونه فيه عن طريق سجل عمليات التحديث والقراءة المتواصلة لتلك الاجزاء مما يجعلها اكثر تناسبا لاستخدامها في البيئات المتغيرة.

ABSTRACT

Distributed database design addresses the problems of fragmentation, allocation and replication. The basic goal of distribution is to find effective way to increase database system reliability and enhancing system performance. These are considered as one of the key research issues today and justifies why the distributed database system (DDBSs) has become widely used and researched.

In this work, the distributed database system has been investigated in the context of relational databases. Different fragmentation, allocation and reallocation scenarios are considered to reach an optimal solutions or near optimal solutions. In the proposed work, fragmentation techniques that can be applied at the initial stage as well as in later stages of a distributed database design for partitioning, and allocating the fragments are performed simultaneously. Moreover, reallocation problem is investigated to obtain optimal dynamic reallocation solutions to be used in a dynamic DDBS environments.

New horizontal and vertical fragmentation and allocation algorithms are presented and new heuristic algorithms to redistribute DDBSs fragments are proposed so that each site decides on its own fragments. The decisions are made on-the-fly regarding the allocation based on current operations and recent history of local read and write, so it is more suitable to use in a dynamic DDBS environments.

Table of Contents

RESEARCH OBJECTIVES

The main objectives of this research are;

- Exploring the shortcoming of some existing fragmentation and allocation techniques.
- Proposing techniques that brings robustness and enhancement to the DDBSs performance and minimize fragment unit transfer cost.
- Implementing the proposed algorithms.
- Comparing our results with that of some existing algorithms.

1. INTRODUCTION

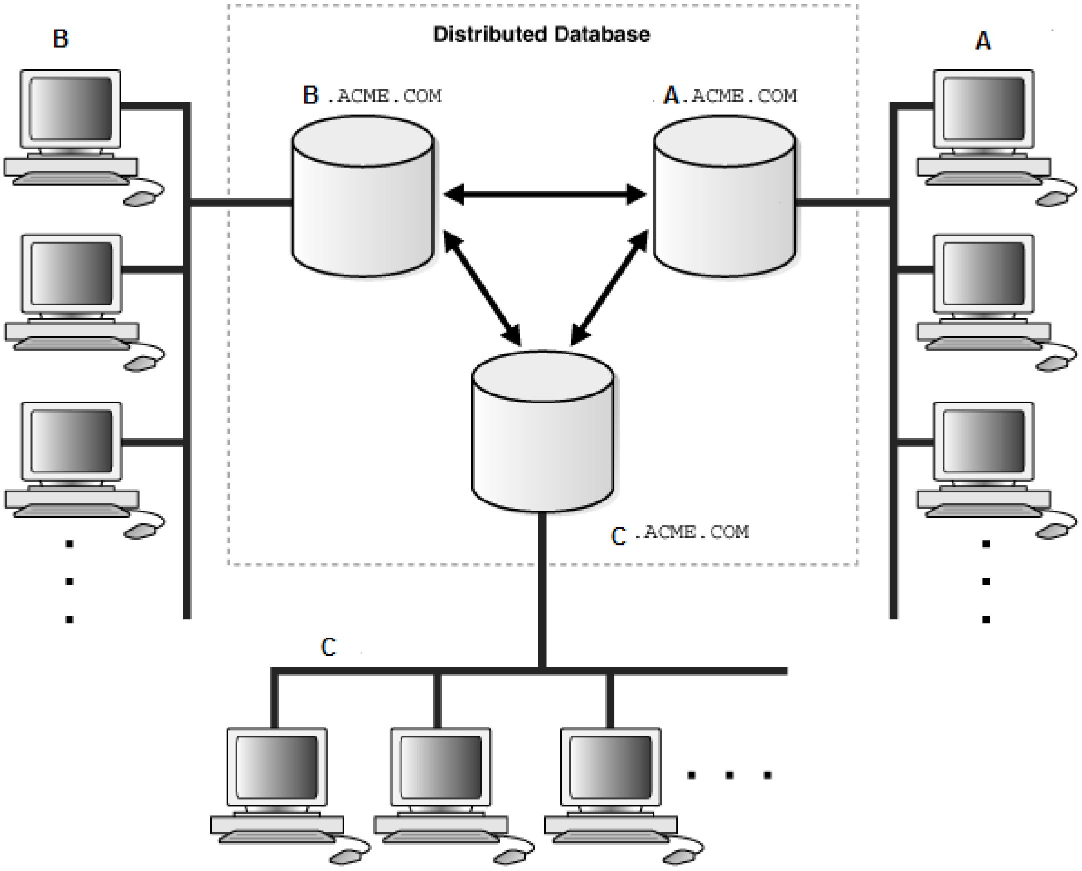
A distributed database system allows applications to access data from local and remote sites. It is a data collection which logically belongs to the same system and satisfies the following assumptions [28]:
(1) Resides on more than one site with computational power; (2) Sites are connected by a communication network; (3) Benefits from distributed database management system (DDBMS) that provides the management of the distributed database system and makes the distribution transparent to the users [11,28].

The distributed database sites can have the same network address or they can exist in the same room but the communication between them is done over a network instead of shared memory. Thus, it is not necessary that database system have to be geographically distributed. DDBSs consider either a heterogeneous distribution where at least one of the databases is not the same type, i.e. different sites may use different system and software, or a homogenous distribution where there is a network of two or more databases that reside on one or more machines, i.e. all sites have the same system and software so that it appears to user as a single system.

Figure 1, illustrates a distributed system that connects three databases: A, B, and C. An application can simultaneously access or modify data in several databases in a single distributed environment. For example, a single query from client B on local database B can retrieve joined data from the *product* table on the local database and the *dept* table on the remote database A.

Moreover, the distributed processing on database management systems (DBMS) is an optimal method of enhancing performance of transactions that manipulate large volumes of data.
This may be done by removing irrelevant data accessed during the execution of queries and by reducing the data exchange among sites, which are the two main goals of the design of distributed databases [25].

Figure 1: *Homogeneous Distributed Database*



Fragmentation, allocation and replication are database distribution design techniques that aim at improving the system performance. Usually two fragmentation techniques are considered vertical fragmentation and horizontal fragmentation. Vertical fragmentation is often considered more complex than horizontal fragmentation because of the large number of alternatives available which makes it nearly impossible to get an optimal solution to the vertical fragmentation problem. Therefore, making proper fragmentation of relations in the context of relational or object oriented database is the basic objective of distributed database system design. Another important objective is the allocation and replication of the fragments in different sites of the distributed system, and local optimization in each site [28, 25]. Actually, horizontal, vertical or mixed fragmentation is a design technique to divide a single relation of a database into two or more partitions such that the union of the partitions forms the original database without any loss of information [28]. This reduces the amount of irrelevant data accessed by the transactions of the database, thus reducing the number of disk accesses. In other words, fragmentation technique aiming at determining fragments to locate at different sites so as to minimize the total data transfer cost incurred in executing a set of queries.

Horizontal fragmentation (HF) allows relation *R* to be partitioned into disjoint tuples or instances. While, the vertical fragmentation (VF) allows relation *R* to be partitioned into disjoint sets of columns or attributes except the primary key. Combination of horizontal and vertical fragmentations to mixed or hybrid fragmentations (MF) are also proposed [25]. The previous approaches of *HF*, *VF* or *MF* have one problem in common: Most of them concentrated only on fragmentation problem and overlooked allocation problem to reduce complexity.

Allocation is the process of assigning the database fragments to DDBSs network sites. When data are allocated, it might either be replicated or maintained as a single copy. The replication of fragments improves reliability and efficiency of read-only queries but increase update query costs [28]. In fact, the fragment allocation operation play key role in DDBSs application performance. Database queries access the sites application on DDBSs and should be performed efficiently. Some of DDBSs fragments has been accessed for updating purposes and other for retrieval purposes so the underlying accessed fragments have needed to be allocated in optimal way to reduce the communication update or retrieval cost or even to minimize the transfer cost during the application execution in some of network sites of DDBSs.

In summary, in a distributed database system if a relation is stored locally, it can be divided in pieces called fragments for physical storage purposes. Data fragments can be stored at different sites, on the same or different machines, where they are more frequently used such that lower network traffic achieved and performance increased. Data fragmentation can be done horizontal, vertical or mixed. To illustrate this , Let $R[A_1 , ..., A_n]$ be relation and A_i are the relation attributes. $\forall i, 1 <= i <= n$.

A horizontal fragment can be obtained by applying the "*Selection*" operation as follows:

$R_i = \sigma_{cond_i} (R)$, where $cond_i$ is the selection condition.

Such that the original relation can be rebuilt back by using the union operation as follows:
$R=R_1$ U R_2 U R_3 …U R_k. However, vertical fragment is obtained by applying the *"projection"* operation as follows:

$R_i = \Pi (A_{x1},..A_{xi}) (R)$, $\forall$ A_{xi}, i= 1,…, P are the relation attributes and *p* is the relation attributes number.

Such that the initial relation can be reconstructed by joining the fragments:

R=R_1⍰R_2⍰…, ⍰R_p.

The resulting fragments should be allocated in effective way or reallocated dynamically when needed.

2. LITERATURE REVIEW

An optimal distributed database design must trade off the performance and cost through the retrieval and update activities at the various sites.

The relation fragmentation and allocation to DDBSs sites is a critical issue because they are key factors in deciding on the application success to support data access and sharing at different sites. Most of the previous work in this domain separate the fragmentation problem from the allocation problem. Some of researches assumes that since the relation fragmentation has been done, then they need to only find the optimal allocation methods regardless of how the fragmentation has been done. In contrast, others try to find the best solution to fragment the relation without thinking on how the allocation will be performed. Only few who started from the fragmentation and proceeded to the allocation problem.

Data fragmentation and allocation problem were introduced in [14]. Studies on vertical fragmentation [26, 12], horizontal fragmentation [10] and mixed fragmentation [37] were investigated, in addition to the allocation and reallocation issues.

Ceri *et al.* [9], proposed a work for *HF* using min-term predicate while Navathe *et al*. used attribute usage matrix (AUM) and Bond Energy Algorithm to produce vertical fragments [26]. He also proposed *VF* algorithm using a graphical technique [27]. Also, *VF* is presented in [2], which is considered as enhancement for the work proposed in [4] by proposing vertical partitioning algorithm using grouping approach that starts from the attribute affinity matrix and generates initial groups based on the affinity values between attributes. In [29] a graph based algorithm for *HF* is presented in which predicates are clustered based on the predicate affinities. Partition evaluator to measure the goodness of *VF* is presented by Chakravarthy *et al.* [8]. Navathe *et al.* [25] proposed *MF* technique where the proposed procedure receives comprises a predicate affinity table and an attribute affinity table. Abuelyaman [5] provided a solution for initial fragmentation of relations by proposing a randomly generated reflexivity matrix, a symmetry matrix and a transitivity module that has been used to produce vertical fragments of the relations, without giving evidence on his assumption that the produced fragments are good. In [24], the concurrency control and constraints of capacity, local process power and capacity on the links of the network have been considered.

Regarding fragments allocation, the earlier investigated researches on dynamic data allocation give a framework for data redistribution [35] and specify how to perform the redistribution process in minimum possible time [30]. In [6], a dynamic data allocation algorithm for non-replicated database systems is proposed, but no modeling is done to analyze the proposed algorithm. [36] Provides a dynamic data allocation model for data redistribution and reallocates in accordance with changing data access pattern.

In [21] a dynamic object allocation and replication algorithm with centralized control has given. While, [20] Presents an optimal algorithm for non-replicated database systems, [33] proposed a threshold algorithm for non-replicated distributed databases where the fragments are continuously reallocated according to the changing data access patterns.

In [16], the authors developed a method for clustering the sites of a network based on the communication cost between them in order to determine the fragment allocation to a group of sites instead of allocating the fragments site by site. Thus, reducing the communication cost will be reduced during the application execution in those sites. The proposed greedy approach in [19], presents a mathematical modeling for data allocation in DDBSs that considers a network communication, local processing and data storage costs to allocate fragment site by site. This methods supports the best location of data fragmentation in DDBSs based on fragment access pattern and cost of moving fragments from one site to another with site constraints maintained as site capacity (C) and the fragment limit (FL). A simple and comprehensive model that shows the transaction behavior in DDBSs for fragment allocation is developed in [1] where two heuristic algorithms were developed to find a near-optimal allocation such that the total communication cost is minimized.

In [18], site-dependent fragmentation graph representation is developed to model the dependencies among the accessed fragments. In this work, the allocation problem is tackled based on both query strategies to address the non-redundant data allocation of fragments in DDBSs. Thus, the query driven data allocation integrated the query execution strategies with the fragmentation. In [13], the data allocation model is proposed to solve the DDBSs problems of the pre allocation and post allocation of data. [15] Presented solutions for data allocation and optimization in distributed database systems that performed by means of mobile agents having attached learning capacities completed by collaboration and coordination functionalities. However, a query processing cost model to evaluate the performance of the system is presented in the context of complex value databases [30], in which a heuristic approach is proposed for fragmentation and fragments allocation using the cost model and is aimed at globally minimizing these costs.

A decentralized approach for dynamic table fragmentation and allocation in distributed database systems based on observation and monitoring of the sites access patterns to tables is proposed in [17] which performs fragmentation, replication, and reallocation based on recent access history aiming at maximizing the number of local accesses compared to accesses from remote sites. [22] Presented a new integer programming formulations for the non-redundant version of the fragment allocation problem which is extended to address problems which have both storage and processing capacity constraints. [3] Addressed the data allocation problem in terms of minimizing two different types of data transmission across the network such that a new heuristic algorithm, that based on the ant colony optimization algorithm, is presented. Taking into account the applied strategies for query optimization and integrity enforcement.

3. PROBLEM STATEMENT

In distributed databases, the communication costs can be reduced by partitioning database tables horizontally or vertically into fragments, and allocating or reallocating these fragments to the network sites where they are most frequently accessed. The basic objective is to make most data accesses local, and avoid remote access as much as possible. Obviously, important challenges in fragmentation and allocation are how to fragment, how to allocate the fragments and to decide on their replication.

Because of dynamic workloads, static fragmentation and allocation may not always be optimal. Thus, the fragmentation and allocation or reallocation management should be dynamic and completely automatic, i.e., changing access patterns should result in re-fragmentation and reallocation of fragments when beneficial, as well as in the creation or removal of fragment replicas.

In this work, several fragmentation, allocation and reallocation scenarios are investigated to reach optimal solutions. Each site decides over its own fragments, and decisions are made on-the-fly regarding the allocation based on current operations and recent history of local reads and writes. Some techniques are proposed to increase the DDBS performance and minimize the unit transfer cost. In addition, new methods are presented to redistribute DDBS fragments. Actually, this work addresses horizontal, vertical fragmentation and allocation problems simultaneously and the reallocation scenarios in the context of the relational model.

In summary, first, this work will propose a technique to fragment a relation horizontally and vertically to effectively find an optimal fragmentation and allocation solution. Then, two heuristic algorithms for fragment redistribution will be presented.

4. REASEARCH METHODOLOGY

Optimal or near optimal fragmentation of relations and the allocation of the resulting fragments is a major research area in distributed databases. Many methods have been presented by the researchers community using empirical knowledge of data access and query frequencies. While optimal fragmentation and allocation at the initial stage of a distributed database design has been rarely addressed.

In the proposed methodology, a horizontal and vertical fragmentation techniques to partition relations of a distributed databases will be presented along with allocation at the initial stage of design.

The proposed research methodology will be performed in two main steps:

1. Horizontal and vertical fragmentation and allocation.
2. Heuristic algorithms for fragments redistribution in DDBSs.

4.1. Horizontal and Vertical Fragmentation and Allocation

4.1.1. Horizontal Fragmentation (HF)

Horizontal fragmentation is performed by dividing a global relation *R* on its tuples by using of the *"selection"* operator as follows;

$$R_j = \sigma_{Pj} (R), \forall j, 1 <= j <= m$$

where P_j is the selection condition as a simple predicate and *m* is the maximum number of fragments. It should satisfy the completeness rule which is satisfied if the selection predicates are complete. In this part, a method to fragment relation R horizontally is presented. The fragmentation technique can be applied at the initial stage of distributed database design as well as in later stages of the design. This method also allocate the fragments properly among the sites of DDBSs at the initial stage.

The proposed technique is based on a simple predicates Pr $[P_1,\ldots, P_n]$ where *n* is the number of relation attributes. Those predicates have been assigned to the relation attributes $[A_1,\ldots.,A_n]$, where every attribute *A* has a particular *Pr*. Therefore, every attribute predicate *Pr* has retrieval frequency (RF) value and update frequency (UF) value achieved by the sites transactions or queries $[Q_1,\ldots,Q_k]$ executed by a site query strategy where *k* is the number of site's queries that are assumed to be the most frequently executed queries in DDBSs and the values of *RF* and *UF* are assigned at the design stage of a distributed database by DDBSs system designer. Generally update incurs more cost than retrieval from the database. So, *UF* values might seem to be greater than *RF* values according to given requirements that extracted from particular DDBS by designer. The attribute predicate is represented as following;

A_i.value ¢ v RF_i .value1 UF_i .value1 represent attribute predicate.

Where ¢ $\in$ [<.>.=], 0 < value1<100 and value v $\in$ attribute domain (A_i.D) , $\forall i$, 1<= i <= n.

For every attribute *A*, there are three predicate states and their *RF* and *UF* values. Therefore, *HF* divides a single relation *R* according to the Retrieval and Update Priority of its Attributes (RUPA).

RUPA is a table which is constructed by computing the retrieval frequency and update frequency for relation attributes individually. The calculation is performed with help of Frequency of Retrieval and Update Attributes (FRUA) matrix and cost function, given that the following requirements are satisfied:

There is a Relation R (A_1, A_2,…., A_n) and predicates Pr(Pr_1, Pr_2,…., Pr_n), where, Pr_i is in one of three states either ($Pr_i > V$), ($Pr_i < V$) or ($Pr_i = V$) and for each Pr_i there is $Pr_i.RF_i$ value and $Pr_i.UF_i$ value representing the predicate retrieval frequency and predicate update frequency respectively for the corresponding attribute issued by executed queries among all DDBSs sites. In addition, DDBSs network consists of many sites S(S_1, S_2, …………,S_m), where each site has capacity C(C_1, C_2, …………..,C_m) and fragment limit FL(FL_1,FL_2…………,FL_m) which indicates how many fragments the site S_j can handle. For each site, there are the most frequently issued queries by this site accounting for say more than 75% of the processing in the distributed database system, Q(Q_1 , Q_2 ……,Q_k) where *k* is the number of site queries. Each query Q_k can be executed from any site with a certain frequency (QF). The execution frequencies of *k* queries at *m* sites can be represented by a $m \times k$ matrix, QF_{km}.

And let RF (A) be the retrieval frequency matrix for corresponding relation attributes predicates and UF(A) be the update frequency matrix for corresponding relation attributes predicates. Both, RF(A) and UF(A) are assumed to be achieved by sites queries.

Based on the requirements above, the horizontal fragmentation will be performed according to particular mechanism using the predicates P_i as selection predicate. This mechanism will be explained in details later in this work.

Based on the resulting fragments, the allocation operation will be performed. Every fragment (F_i) will be allocated in that corresponding site (S_j) that has max value for its related predicate unless the underlying site constraints has been violated.

This technique will be performed in the following scenarios:

1. Without maintaining site constraints of capacity (C) and fragment limit (FL).
2. With maintaining the above sites constraints.

In the last step, the obtained results from the above two scenarios will be compared to show which method is better than other.

4.1.2. Vertical Fragmentation (VF)

Vertical fragmentation is performed by projecting a global relation *R* on its attributes by applying the "*Project"* operator as follows;

$$R_j = \prod \{A_j\}_{,key} \ R, \forall j, 1 <= j <= m$$

Where, $\{A_j\}$ is a set of attributes not containing the primary key, upon which the vertical fragmentation is defined and *m* is the maximum number of fragments. In this methodology, a new heuristic method for vertical fragmentation of the relations in a distributed database is presented. This method is capable of taking suitable fragmentation decision at the initial stage by using the information gathered during requirement analysis and DDBSs design phase. It can also allocate the attributes properly to the different sites of DDBSs.

In this part, heuristic approach is developed to fragment the relation *R* vertically. The vertical fragmentation will be performed based on the following requirements:

- The relation attributes vector: R [A_1, A_2 ………… A_n].
- Available network sites in DDBSs environment: S [S_1, S_2, ………… S_m].

- Available network sites capacities C [C1, C2, ………., Cm], this is restricted by the constraint that no two sites should have the same capacity.
- The most executed queries for all DDBSs sites: Q [Q_1, Q_2 ………… Q_m].
- The query frequencies matrix in their sites, *QF*. This is restricted by the constraint that if the same query exists in several sites, then it will be treated as a different query and will have different frequency value in the relevant site. This way the attribute duplication probability will be reduced to the minimum level. Using the site capacity, a formula will be made to avoid attributes overlapping when distributed over network sites at the allocation stage.

Also, let RF(A) be the retrieval frequency matrix for corresponding relation attributes and UF(A) be the update frequency matrix for corresponding relation attributes. Both, RF(A) and UF(A) are assumed to be achieved by sites queries.

Based on the requirements above, the relation attributes will be distributed over the network sites. The retrieval frequency and update frequency values for the relation attributes, which are assumed to be obtained during the DDBSs design phase by DDBSs designer play key role in performing the vertical fragmentation and allocation using the proposed cost model at the initial stage.

In summary, the allocation of attribute A_i to site S_j will be done for A_i that has the highest number of retrieval and update frequency multiplied by the query frequency at the same site QF_k. This method guarantees that no attribute replication over network sites will take place except for the primary key attribute.

In the last part of this method, an evaluation will be performed and the obtained results will be compared with other vertical fragmentation technique.

4.2. Heuristic Algorithms for DDBS Fragments Redistribution

In this part, a redistributed dynamic data allocation model with two heuristic algorithms that maintain site constraints will be described.

4.2.1. Optimal Algorithm Enhancement (OAE)

It is a new dynamic data allocation algorithm for non-replicated distributed database system, which exploits the concepts of the algorithms presented in [6, 31,33,34]. Optimal Algorithm Enhancement (OAE), is defined as a combination of the optimal algorithm, A threshold Algorithm and Threshold Time Constraint Algorithm (TTCA). It assumes that DDBSs fragments has been distributed previously using static allocation method. The proposed algorithm reallocates data fragments with respect to the constraints of site capacity and site fragment limit, changing data access pattern and access time. Therefore, when the migration decision has been determined through specific time, migrating the intended fragment will be performed using the shortest path from the source (old place) to the destination (new place) which helps in reducing the transfer cost. This method is suitable for DDBSs where data access pattern changes dynamically.

Actually, this algorithm will decrease the movement of data over the network and also improve the overall performance of the system. The most important point is that the fragment migration strategy uses the shortest path algorithm which help to minimize the fragment transfer cost in a fully connected network. Moreover, if there are many sites requesting the same fragment, then, the optimal location can be calculated by choosing the location that supports the minimum cost for placing that fragment at that site. In the last step, an evaluation will be performed by comparing the proposed algorithm with the previous ones [6,34,31].

4.2.2. Proposed Heuristic Redistribution Algorithm (HRA)

The objective of this approach is to find the optimal fragments redistribution over network sites. This approach assumes that fragments in DDBSs has been distributed in some way such that the initial distribution has been completed based on a given retrieval matrix (RM) of all fragments in DDBSs and query frequencies (QF) for the most queries executed in DDBSs that assigned by the designer at the design stage. Based on a given update fragments matrix (UM), query frequencies matrix (QF) and communication cost matrix between network sites, the reallocation of fragments will be performed and fragments duplication will be removed if there are any.

The allocation will be performed based on the site which has the minimal update cost to select the fragment satisfying the migration conditions to be the locally stored fragment in that site. In other words, the costs of reallocating fragment to a network site is evaluated and then the decision is made by selecting a site that has the least update query cost.

In the last step of this approach, the performance evaluation will be performed.

The proposed heuristic algorithm formulated in the following steps;

1) Perform the initial allocation for all fragments over network sites based on the RM and QF matrix.

2) Take the most frequently used accounting for say more than 8% of the processing in the distributed database system.

3) Optimize all the queries and construct the fragments update frequency matrix (FUFM) for each database relation type *R* based on the queries based on *FM* and *QF* matrix.

4) Calculate the update cost at each site for each fragment to construct the cumulative update table for all fragments over all sites.

5) Calculate the pay value at each site for each fragment which represents the costs of total update value that achieved by all executed queries in every site individually.

6) Distribute all fragments to the site which has the least pay value.

5. PLAN

5.1. Thesis Outlines

i. Abstract

ii. Introduction

iii. Related Work

iv. Proposed Technique

v. Implementation

vi. Evaluation

vii. Conclusion

viii. References

5.2. Proposed Schedule

The following table contain of the expected times for the thesis:

Task	Time (months)											
	1	2	3	4	5	6	7	8	9	10	11	12
Literature Review	🟨	🟨	🟨									
Studying Different Fragmentation and Allocation Techniques		🟨	🟨	🟨	🟨	🟨						
Studying Several Reallocation Techniques				🟨	🟨	🟨						
Implementation							🟨	🟨	🟨	🟨		
Techniques Evaluation									🟨	🟨	🟨	
Thesis writing						🟨	🟨	🟨	🟨	🟨	🟨	🟨

5.3. Proposed Budget

The following table contents the estimated budget which may be 8 thousand SR.

Requirement	Estimated Price
Books	1000
Laptop & utilities	6000
Printer	1000

6. REFERENCES

[1] H. Abdalla, et al., “Using a Greedy-Based Approach for Solving Data Allocation Problem in a Distributed Environment”, to appear in the Proceedings of the 2008 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'08).

[2] H. Abdalla and M. AlFares, “An Enhanced Grouping Algorithm for Vertical Partitioning Problem in DDBs”, ISCIS'07. 22nd International Symposium on Computer and Information Sciences, November 7-9, 2007, Ankara, Turkey.

[3] R. Karimi Adl, S. M. T. Rouhani Rankoohi, A new ant colony optimization based algorithm for data allocation problem in distributed databases, Knowl Inf Syst (2009) 20:349–373, 23 January 2009.

[4] M. AlFares, H. Abdalla and F. Marir, “Vertical Partitioning for Database Design: A Grouping Algorithm”, 16th International Conference on Software Engineering and Data Engineering July 9-11, 2007, Nevada USA, Las Vegas, July 2007.

[5] E. S. Abuelyaman, “An optimized scheme for vertical partitioning of a distributed database,” Int. Journal of Computer Science & Network Security, Vol. 8, No. 1, 2008.

[6] A. Brunstroml, S.T. Leutenegger and R. Simhal,“Experimental Evaluation of Dynamic Data Allocation Strategies in a Distributed Database with changing Workload”, ACM Trans. Database Systems,1995.

[7] R., BLANKINSHIP, HEVNER, A.R., AND YAO, S.B. An iterative method for distributed database design. In proceedings of the 17th International Conference on Very Large Data Base (1991), G.M. Lohman, A. Sernadas, And R. Camps, Eds., Morgan Kaufmann, pp. 389-400.

[8] S. Chakravarthy, J. Muthuraj, R. Varadarajan, and S. B. Navathe, “An objective function for vertically partitioning relations in distributed databases and its analysis,” Distributed and Parallel Databases, Springer, Vol. 2, No. 2,pp. 183–207, 1994.

[9] S. Ceri, M. Negri, and G. Pelagatti, “Horizontal data partitioning in database design,” in Proc. ACM SIGMOD,1982, pp. 128–136.

[10] S.Ceri, S.B. Navathe and G. Wiederhold, Distribution Design of Logical Database Schemas, IEEE Transactions on Software Engineering, 9:487-503. 1983.

[11] S. Ceri and G. Pelagatti, Distributed Databases Principles and System, 1st ed., New York: McGraw-Hill, 1984.

[12] S.Ceri, B.Pernici and G. Wiederhold, Optimization Problems and Solution Methods in the Design of Data Distribution, Information Systems, 14: 261-272. 1989.

[13] NH, Daudpota, Five steps to construct a model of data allocation for distributed database systems.Journal of Intelligent Information Systems: Integrating Artificial Intelligence and Database Technologies. 1998 vol.11, no.2; Sept.-Oct. p.153-68.

[14] K.P Eswaran, Placement of Records in a File and File Allocation in a Computer Network, in Proceedings of IFIP Congress on Information Processing, Stockholm, Sweden, pp: 304-307. 1974.

[15] H. Grebla, G. Moldovan, S. A. Darabant, A. Câmpan, Data Allocation In Distributed Database Systems Performed By Mobile Intelligent Agents, Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics - ICTAMI 2004, Thessaloniki, Greece.

[16] I. Hababeh. A Method for Fragment Allocation Design in the Distributed Database Systems. The Sixth Annual U.A.E. University Research Conference.

[17] J. O. Hauglid , N. H. Ryeng, DYFRAM: dynamic fragmentation and replica management in distributed database systems, Distrib Parallel Databases (2010) 28: 157–185 ,8 September 2010.

[18] Yin-Fu Huang, Jyh-Her Chen, Fragment Allocation in Distributed Database Design. Journal of Information Science and Engineering. 2001, 17 P. 491-506.

[19] Ahmad. I., KARLAPALEM. K, KWOK. Y, Evolutionary algorithms for allocating data in distributed database systems. Distrib.Parallel Databases 11(1), 5–32 (2002).

[20] L.S. John, “A Generic Algorithm for Fragment Allocation in Distributed Database System”, ACM 1994.

[21] W.J. Lin and B. Veeravalli, “A Dynamic Object Allocation and Replication Algorithm for Distributed System with Centralized Control,” International Journal of Computer and Application, Vol. 28, no. 1,pp. 26-34, 2006.

[22] S. Menon, Allocating Fragments in Distributed Databases, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 16, NO. 7, JULY 2005.

[23] H. Ma, K. Schewe, and Q. Wang, "A heuristic approach to cost-efficient fragmentation and allocation of complex value databases", in Proc. ADC, 2006, pp.183-192.

[24] March and Rho, S. (1995). Allocating Data and Operations to Nodes in Distributed Database Design, IEEE Trans.on Knowledge and Data Engineering, 7(2).

[25] S. Navathe, K. Karlapalem, and M. Ra, “A mixed fragmentation methodology for initial distributed database design,” Journal of Computer and Software Engineering Vol. 3, No. 4 pp 395–426, 1995.

[26] S. B. Navathe, S. Ceri, G. Wiederhold, and J. Dour,“Vertical partitioning algorithms for database design,”ACM Transactions on Database Systems (TODS), Vol. 9,No. 4, pp, 680–710, 1984.

[27] S. B. Navathe, and M. Ra, “Vertical partitioning for database design: A graphical Algorithm,” ACM SIGMOD Record, Vol. 14, No. 4, pp. 440-450, 1989.

[28] M. T. Ozsu and P. Valduriez, Principles of Distributed Database Systems, 2nd ed., New Jersey: Prentice-Hall,1999.

[29] M. Ra, “Horizontal partitioning for distributed database design,” In Advances in Database Research, World Scientific Publishing, pp. 101–120, 1993.

[30] P.I., Rivera-Vega, R. Varadarajan and S.B. Navathe, 1990. Scheduling Data Redistribution in Distributed Databases, in IEEE Proceedings of the Sixth International Conference on Data Engineering, pp: 166-173.

[31] Arjan Singh and K.S. Kahlon, "Non-replicated Dynamic Data Allocation in Distributed Database Systems", IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.9, pp. 176-180, September 2009.

[32] Toadere, Teodor . (2002). Graphs. Theory, Algorithms and Applications,Editura Albastra, Cluj-Napoca, ROMANIA).

[33] T. Ulus and M. Uysal, “A Threshold Based Dynamic Data Allocation Algorithm- A Markove Chain Model Approach”, Journal of Applied Science 7(2), pp 165-174, 2007.

[34] T. Ulus and M. Uysal, “Heuristic Approach to Dynamic Data Allocation in Distributed Database Systems”, Pakistan Journal of Information and Technology, 2(3): pp. 231-239, 2003.

[35] Wilson, B. and S.B. Navathe, 1986. An Analytical Framework for the Redesign of Distributed Databases, in Proceedings of the 6th Advanced Database Symposium, Tokyo, Japan, pp: 77-83.

[36] O. Wolfson, S. Jajodia, and Y. Huang, “An Adaptive Data Replication Algorithm,” ACM Trans. Database Systems, vol. 22, no. 2, pp. 255-314, 1997.

[37] Y.Zhang, and M.E. Orlowska, On Fragmentation Approaches for Distributed Database Design, Information Sci., 1: 117-132. 1994.