**DISTRIBUTED DATABASE MANAGEMENT SYSTEMS AND DATA ALLOCATION**

The efficiency and performance of Distributed Database Management Systems (DDBMS) is mainly measured by its proper design and by network communication cost between sites. Fragmentation and distribution of data are the major design issues of the DDBMS. In this paper, we propose new approach that integrates both fragmentation and data allocation in one strategy based on high performance clustering technique and transaction processing cost functions. This new approach achieves efficiently and effectively the objectives of data fragmentation, data allocation and net- work sites clustering. The approach splits the data relations into pair-wise disjoint fragments and determine whether each fragment has to be allocated or not in the network sites, where allocation benefit outweighs the cost depending on high performance clustering technique. To show the performance of the proposed approach, we performed experimental studies on real database application at different networks connectivity. The obtained results proved to achieve minimum total data transaction costs between different sites, reduced the amount of redundant data to be accessed between these sites and improved the overall DDBMS performance.

Many researches have been done in the last few years in order to improve the performance of distributed data- base management systems (DDBMS). DDBMS is a collection of logically interrelated data that are physically allocated at different locations over a computer network. Most of recent researches are concerned about keeping the performance of DDBMS high so that they focused on how to design the database such that the over- all cost is kept minimal. Keeping the cost minimal is not an easy task as there are huge amount of transactions processing that increase the complexity of distributed databases. Several techniques have been proposed in order to improve database performance which can be achieved by improving at least one of the following database management issues: database fragmentation, data allocation and replication, and the network sites clustering.

The fragmentation process divides the database into portions each of which is called a fragment. Fragmentation can be horizontal, vertical or mixed. The main advantage of fragmentation is to improve the performance of distributed database design by increasing the efficiency since data is stored only where it is needed. Fragments can be allocated at different network sites in a process called data allocation.

The fragments allocation is an NP-complete problem so that the complexity is high. In order to reduce the complexity, some heuristic algorithms have been proposed to solve the problem. In the allocation process, each fragment is assigned to a network node and sometimes to more than one node to achieve the data availability, system reliability and performance.

The clustering technique is used for grouping distributed database network sites into logical clusters. In order to reduce the communication time for data allocation, there are many algorithms that are use to find the optimal solution for grouping distributed database network sites into a disjoint clusters and making a better data distribution among them. The clustering technique aims at eliminating the extra communication costs between the net- work sites and then enhancing the DDBMS performance.

Many existing algorithms of data fragmentation and allocation in DDBMS assume some restrictions on the number of network sites so that the results of such algorithms are impractical, and reflected on the efficiency and validity of their outcomes. Moreover, some constraints on network connectivity and transactions processing time will limit the applicability of the proposed solutions to a small number of DDBMS cases.

One of the drawbacks of fragmentation and allocation solutions is the high computational complexity of their associated algorithms. In fact, when distributing a database over a network with a big number of sites and then finding an efficient, reliable and optimal solution for fragmentation and allocation are considered difficult tasks.

This paper proposed a new technique that splits the database relations into disjoint fragments. In addition, it introduces a high speed clustering technique that groups the distributed network sites into a set of clusters ac- cording to their communication cost. Also, it proposes a new intelligent technique for data allocation and redistribution based on transactions processing cost functions. Moreover, it implements a user-friendly simulation tool that performs fragmentation, clustering, allocation and replication of a database, in addition to assisting database administrators in measuring DDBMS performance.

**DDBMS (Distributed Database Management System)**

A DDBMS (distributed database management system) is a centralized application that manages a distributed database as if it were all stored on the same computer. The DDBMS synchronizes all the data periodically, and in cases where multiple users must access the same data, ensures that updates and deletes performed on the data at one location will be automatically reflected in the data stored elsewhere.

**Functions of Distributed Database System**

Distribution basically leads to increased complexity in the system design and implementation. This is to achieve the potential advantages such as:

1. Network Transparencies
2. Increased Reliability
3. Improved Performance
4. Easier Expansion

**Function of Centralized DBMS:**

1. The basic function of centralized DBMS is that it provides complete view of our data.  
   For example, we can have the query for the number of customers who are willing to buy worldwide.
2. The second basic function of Centralized DBMS is that it is easy to manage than other distributed systems.

The Distributed Database must be able to provide the following function in addition to those of a centralized DBMS’s.

**Functions of Distributed database system:**

* **Keeping track of data**  
  The basic function of DDBMS is to keep track of the data distribution, fragmentation and replication by expanding the DDBMS catalog.
* **Distributed Query Processing**  
  The basic function of DDBMS is basically its ability to access remote sites and to transmits queries and data among the various sites via a communication network.
* **Replicated Data Management**   
  The basic function of DDBMS is basically to decide which copy of a replicated data item to access and to maintain the consistency of copies of replicated data items.
* **Distributed Database Recovery**   
  The ability to recover from the individual site crashes and from new types of failures such as failure of communication links.
* **Security**   
  The basic function of DDBMS is to execute Distributed Transaction with proper management of the security of the data and the authorization/access privilege of users.
* **Distributed Directory Management**   
  A directory basically contains information about data in the database. The directory may be global for the entire DDB, or local for each site. The placement and distribution of the directory may have design and policy issues.
* **Distributed Transaction Management**   
  The basic function of DDBMS is its ability to devise execution strategies for queries and transaction that access data from more than one site and to synchronize the access to distributed data and basically to maintain the integrity of the complete database.

**Database Partitioning**

Various methods already exist describe data fragmentation in distributed DDBMS. Naturally, there are benefits and drawbacks to all schemes. Some methods need to incorporate performance evaluation ways, may not minimize the transactions response time and cannot guarantee the ability to process a given portion of a given transaction in all sites.

In our proposed approach, the database will be partitioned into pair-wise disjoint fragments by using a horizontal partitioning technique, in which the records of a relation split into disjoint fragments; this strategy guarantees the ability to processing all portions of a given transaction and distributes it precisely over the DDBMS sites.

Generating data fragments accomplished by performing the following processes respectively: defining trans- actions, creating segments and extracting disjoint fragments. Figure 1 below describes the architecture of the fragmentation method that supports the use of knowledge extraction and helps to achieve the effective use of small data packets.

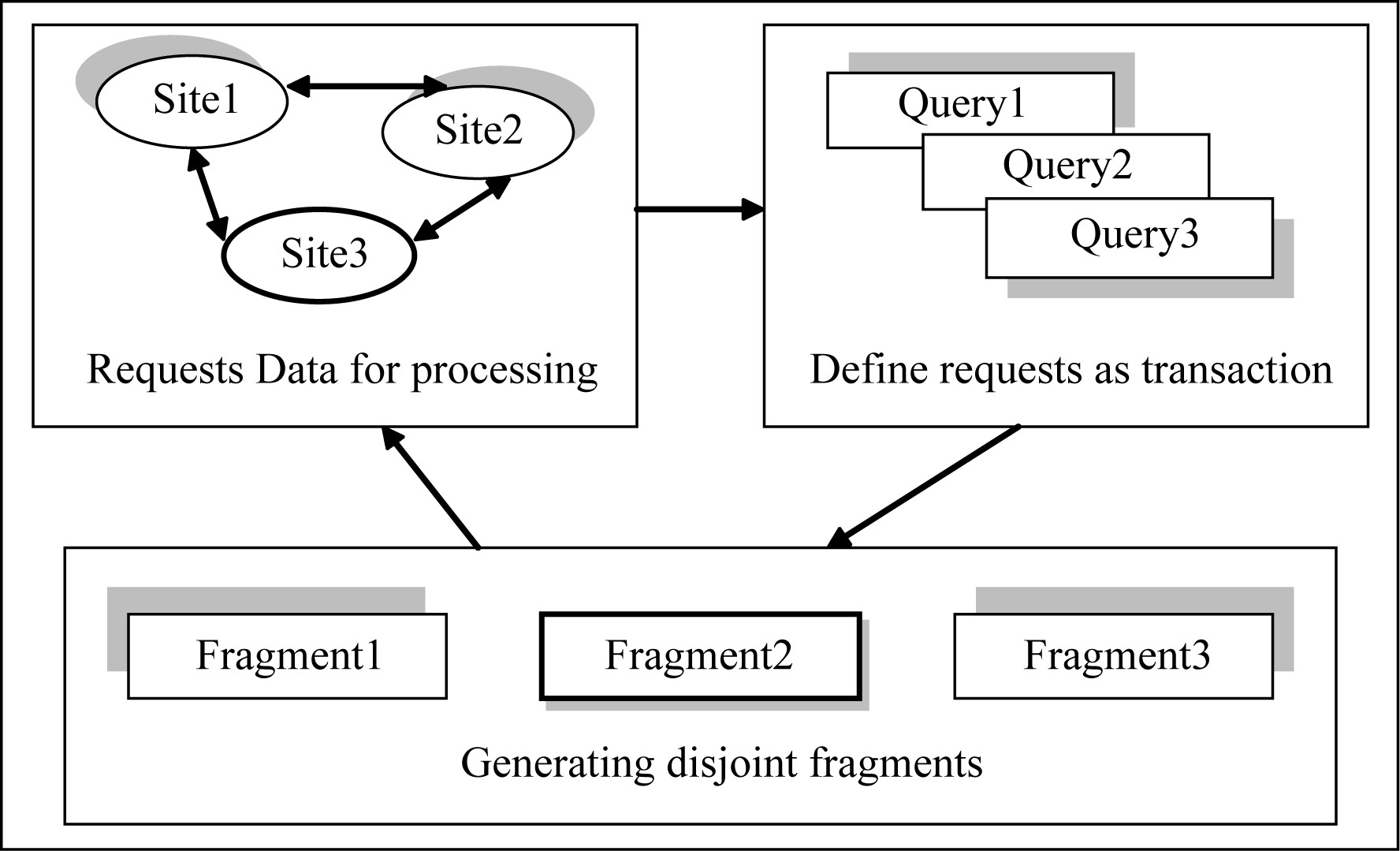


Figure 1 Generating disjoint fragments

As shown in Figure 1, the data request is initiated from the DDBMS sites (Site1, Site2, and Site3) and defines transactions as queries (Query1, Query2, and Query3) and then these transactions (queries) are processed into disjoint fragments (Fragment1, Fragment2, and Fragment3). The database transaction could be associated with more than one relation, in this case, the transaction should be divided into a number of sub-transactions equal to the number of relations used that transaction. The process of generating fragments is described as follows:

1. Database fragmentation starts with any two fragments having intersection records between them. If there is an intersection, then three disjoint fragments will be generated as follows:

* The common records in the two intersected fragments,
* The records in the first fragment but not in the second segment, and
* The records in the second fragment but not in the first segment.

1. The intersecting fragments are then removed from the fragments list. This process continues until removing all the intersecting fragments.
2. The new derived fragments and the non-overlapped ones that do not intersect with any other fragments from the new list of totally disjoint fragments.

**Network Sites Clustering**

The benefit of generating database disjoint fragments can’t be completed unless it enhances the performance of the distributed database system. As the number of database sites becomes too large, a problem of supporting high system performance with consistency and availability constraints becomes crucial. Different techniques could be developed for this purpose; one of them consists of clustering distributed networking sites. Clustering database sites is a technique in which the sites that have similar physical property (e.g., having comparable communication costs) are logically grouped together in order to increase the performance of the distributed database system. However, grouping sites into clusters is still an open problem and it is proven that the optimal solution to this problem is NP-Complete since it is transformed to a cheapest path problem. Therefore, near-optimal solution for grouping database sites into clusters helps to eliminate the extra communication costs between the sites during the process of data allocation and improves the system performance. Performing sites grouping after database fragmentation, will speed up the process of data allocation by distributing the fragments over clusters of sites rather than site by site. Thus, the communication costs are minimized and the distributed database system performance is improved.

**Clustering Algorithms**

Since the task of clustering is subjective, the means that can be used for achieving this goal are plenty. Every methodology follows a different set of rules for defining the ‘*similarity’* among data points. In fact, there are more than 100 clustering algorithms known. But few of the algorithms are used popularly, let’s look at them in detail:

* **Connectivity models:** As the name suggests, these models are based on the notion that the data points closer in data space exhibit more similarity to each other than the data points lying farther away. These models can follow two approaches. In the first approach, they start with classifying all data points into separate clusters & then aggregating them as the distance decreases. In the second approach, all data points are classified as a single cluster and then partitioned as the distance increases. Also, the choice of distance function is subjective. These models are very easy to interpret but lacks scalability for handling big datasets. Examples of these models are hierarchical clustering algorithm and its variants.
* **Centroid models:** These are iterative clustering algorithms in which the notion of similarity is derived by the closeness of a data point to the centroid of the clusters. K-Means clustering algorithm is a popular algorithm that falls into this category. In these models, the no. of clusters required at the end have to be mentioned beforehand, which makes it important to have prior knowledge of the dataset. These models run iteratively to find the local optima.
* **Distribution models:** These clustering models are based on the notion of how probable is it that all data points in the cluster belong to the same distribution (For example: Normal, Gaussian). These models often suffer from overfitting. A popular example of these models is Expectation-maximization algorithm which uses multivariate normal distributions.
* **Density Models:** These models search the data space for areas of varied density of data points in the data space. It isolates various different density regions and assign the data points within these regions in the same cluster. Popular examples of density models are DBSCAN and OPTICS.

**Fragments Allocation**

Data allocation and replication technique places and distributes the database fragments on the clusters and their respective sites. Initially fragments are allocated to the clusters that have transactions using that fragments. Al- locating fragments to a small number of clusters instead of large number of sites will reduce the number of communications and therefore enhance the system performance. Figure 2 illustrates the structure of data alloca- tion and replication technique.

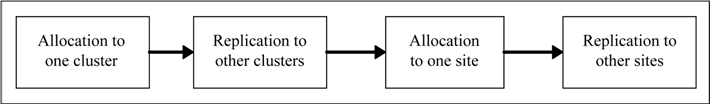


Figure Data Allocation and Replication Technique

A heuristic fragment allocation and replication technique will be introduced to perform the processes of frag- ments allocation in the distributed database system. Initially, all fragments are subject for allocating to all clus- ters having transactions using these fragments at their sites. If the fragment shows positive allocation decision value (i.e. allocation benefit greater than or equal to zero) for a specific cluster, then the fragment is subject for allocating at each site in this cluster, otherwise the fragment is not allocated (cancelled) from this cluster. This step is repeated for each cluster in the distributed database system.

As a result of the previous step, the fragment that shows positive allocation decision value at any cluster is a candidate for allocating at all sites of this cluster. If the fragment shows positive allocation decision value at a site of cluster that already shows positive allocation decision value, then the fragment is allocated at this site, otherwise, the fragment is not allocated to this site. This step is repeated for each site at this cluster.

To ensure data availability in the distributed database system, each fragment should be allocated to at least one cluster and one site. In case a fragment shows negative allocation decision value at all clusters, the fragment is allocated to the cluster that holds the least average communication cost and then to the site that has the least communication cost in this cluster.

**Allocation Cost Functions**: The allocation cost functions identifies the allocation status which is computed as a logical value for the com- parison between the cost of remote access the fragment to the cluster/site and the cost of allocating the fragment to the cluster/site. If the cost of remote access the fragment to the cluster/site is greater than or equals to the cost of allocating the fragment to the cluster/site, then the allocation status is positive and the fragment is allocated to the cluster/site. On the other hand, if the cost of remote access is less than the cost of allocating, then the alloca- tion status is negative and the fragment is cancelled from the cluster/site.

**Performance Evaluation**

We believe that the performance of allocation and network load are the major performance issues; hence, we will focused our performance evaluation on these two issues as they will maximize the overall system through- put at each network site in the distributed database environment, and minimize the cost of data that has been transferred and processed.

**Fragment Allocation Performance Evaluation**

The technique of fragment allocation and replication at clusters is evaluated according to the performance generated by reducing the size of fragments that allocated finally at the clusters. The closest methods in the literature to the proposed technique of fragment allocation and replication are those proposed above. The main differences between these two methods are described as follows.

Compared to the allocation techniques in the literature, our allocation and replication technique considers different communication costs, namely, the cost of the real network communication between cluster sites, the up- date and retrieval costs (*i.e*., it mostly represents the cost of writing operation that takes place during the execution time). Moreover, with our clustering technique, database sites are grouped according to a clustering range and not only to a specific communication cost. Our clusters can communicate with each other instead of prefer- ring to have all fragments in their sites. This communication is cheaper than allocating fragments in all sites. In addition to the gain in term of communication cost, the independency of our clusters makes our DDBMS more reliable and more functional.

**Network Performance Evaluation**

The DDBMS network workload involves the queries from a number of database users who access the same data- base simultaneously. The amount of data needed per second and the time in which the queries should be processed depend on performance of the database applications running under DDBMS network.

* **Server Load** The server load determines how much the speed of the server in terms of bits per seconds. The following figures depict the network server loads in the proposed DDBMS.
* **Network Delay** The network delay is the delay caused by the transactions traffic on the server of the distributed database system. The maximum time required for the network system to reach the steady state is defined as network delay, it’s measured in millisecond.