An Algorithmic Model Scenario for Fragment Allocation to Handle Transaction Conflicts in Heterogeneous Distributed Databases

Dharavath Ramesh, Member IEEE, ACM and Chiranjeev Kumar, Member IEEE*
Department of Computer Science and Engineering
Indian School of Mines (ISM), Dhanbad, Jharkhand -826004, India

ABSTRACT

The design of fragmentation is the first problem that must be solved in the design of data distribution. The purpose of fragmentation design is to determine non- overlapping fragments which are logical units of allocation at different sites that are appropriate for the data allocation. For a distributed database system to function efficiently, the fragments of the database need to be located at various sites across the communication network. The problem of allocating these fragments to the most appropriate sites is a difficult one to solve, however, with most approaches available relying on heuristic techniques. A data conflict between component databases is a crucial problem in building multi-database systems. This paper presents a framework and algorithmic model for classifying these conflicts.

Keywords

Distributed databases, non-redundant allocation, reformulation, Fragment allocation, allocation model.

1. Introduction

According to the fragmentation aspect data allocation is the most important technique in distributed database design. In fact, fragmentation starts from global relations without data allocation. Distributed database design involves the following factors; (i) fragmentation according to global relation, (ii) number of copies replicated according to a particular transaction, (iii) how the fragment allocation to be done at various sites of the network and (iv) the information for fragmentation and data allocation. To simplify the above factors, we address an algorithmic model for fragmentation and allocation in order to avoid data conflicts, by assuming that global relations have been fragmented. This instance is investigated by replicated ones, in Wide Area Network (WAN) to minimize the communication cost.

For a read request by a transaction on a data item, it is quite simple to load the fragment at the issuing site, or it may be a complicated to load the target fragment from a remote site. A write request is most complicated in the case of write propagation in order to maintain consistency among all multiple copies which are spread over the network. The frequency request issued at the sites is also considered in this model. This is one of the difficult instances to implement in HDDBs. Why because a transaction may be issued by different clients at different sites on the network. So the cost factor also be considered to measure the frequencies of the requests issued by various clients.

The time complexity of horizontal fragmentation is $o((n^2)^m)$ where m is the count of simple predicate and n is the number of sites whereas the complexity case of vertical fragmentation is $o((n^m)^m)$ where m is the number of sites and n is the non-primary attributes. According to the relation model, some existing techniques for horizontal may fall into two cases: Min-term techniques [4] and affinity techniques [7].

After fragmenting the global schemas the next step is to allocate the fragments according to the global aspects. Several allocation techniques have been published towards fragmentation in distributed environment. Chang developed a network flow algorithm [6] to solve the issue of database allocation. As the allocation is NP-complete, some heuristic algorithms like knapsack solutions [8] and branch-bound techniques [5] were developed to solve this issue. Furthermore, there were other works implemented on the performance such as file redundancy [2] and allocation of database [9]. Recently a problem [3] called zero/one integer was formulated, and incorporated with concurrency mechanisms [1]. A new allocation model with genetic algorithm was developed by Rho to perform operations on nodes [11]. Even a large amount of techniques have proposed models to allocate fragments in a distributed database environment, most of them were very complicated and not well understood to solve issues related to HDDBMs. Thus, it is very difficult to use them in a real scenario. In this paper, we propose a simple model which reflects transaction behaviour in heterogeneous distributed databases.

2. The model

A. *Description and Notations*

Before beginning our methodology towards allocation, the problem must be clarified. Here we consider WAN environment because of the impact of storing copies on the sites of a LAN is not sufficient. Assume the WAN consisting of sites $S = \{S_1, S_2, ..., S_m\}$ on a set of transactions $T = \{T_1, T_2, ..., T_n\}$ is running and a set of fragments $F = \{F_1, F_2, ..., F_n\}$. To make the problem more precise, we consider that it involves not only the number of copies, but also the optimal allocation of each copy in F of S, according to the transaction T. As per the optimality definition, there are two parameters [12]: (1) the minimum cost and (2) performance.

B. *Requirements*

Before we define the formulae, some information must be shared in advance; that is (i) data about the database, (ii) the transaction instance, (iii) the site instance, and the network.

Database intent: The size of the fragment, called $(FRAG_j)$, must be defined. Why because, it plays a major role when the communication cost is calculated.

Transaction intent: In this model, we consider two matrices, Retrieval and Update according to transactions. The access frequency elements of Retrieval and Update are r_{ij} and u_{ij} with respect to transaction Ti. All the fragments have not been accessed by a transaction.

Retrieval matrix: (R)

TABLE1: Data Description for Retrieval

	F_1	F_2	F_3	F_4	F_5
T_1	2	3	0	0	0
T_2	2	0	0	1	0
T_3	0	0	3	0	0
T_4	3	0	2	0	0

Update matrix: (U)

TABLE2: Data Description for Update

	F_1	F_2	F_3	F_4	F_5
T_1	0	0	0	1	2
T_2	0	3	0	0	0
T_3	2	1	0	1	0
T_4	3	2	0	0	3

In the matrices R and U, transaction T_4 retrieves F_1 three times and updates F_1 three times, F2 twice, and F4 zero for each run. The number of tuples will not be retrieved of updated at all times as the same for all transactions. In order to know about how many fragments to be accessed by a transaction Ti, here we define a SELC matrix which indicates the percentage of accessed transactions. In this work a synchronized fragmentation and allocation method have been proposed to perform fragmentation using a cost model, which would be depicted as the linear functionality.

SELC matrix: (%)

TABLE3: Data Description for Selection

	F_1	F_2	F_3	F_4	F_5
T_1	0.1	0.1	0	0.3	0.2
T_2	0.1	0.3	0	1	0
T_3	2	5	0.1	0.5	0
T_4	0.5	0	10	0	3

Furthermore, we also define a matrix FREQN which indicates the execution frequency of all the transactions issued at each site.

FREQN matrix:

TABLE4: Data Description for Frequency

	S_1	S_2	S_3	S_4
T_1	0	3	1	2
T_2	2	0	0	1
T_3	3	0	2	0
T_4	0	0	3	1

Transaction T_2 runs twice at site S_1 and once at site S_4.

Site intent: It is about storage and processing capacity.

Network intent: Here we consider WAN environment to perform the fragmentation allocation with respect to the sites. Here, we also consider the cost (C cost) of initiating a packet with size p while the cost of transmitting ($TRANSM_{ij}$) a unit data with size m. It can be assumed to be a linear functional aspect as follows:

$$\mathrm{C}_{\mathrm{Cost}}(TRANSM_{ij}, m) = C_{Cost} * \frac{m}{p} + TRANSM_{ij} * m \tag{1}$$

C. *The Cost Model*

Here we originate cost equations with respect to communication to predict a transaction's behaviors in a WAN environment. As we mentioned in section B, we try to find an optimal allocation which can minimize the communication cost. Where, the communication consists of two workings: The cost for loading $CommC_{ld}$ and the cost for transaction processing $CommCT_{proc}$.

$$\min(CommC_{ld} + CommCT_{proc}) \tag{2}$$

The eq. (2) can be expressed further as:

$$CommC_{ld} = \sum_{j=1}^{n}\sum_{k=1}^{m} AT_{j,k} * C_{Cost}(TRANSM_{SITE}, k, size(FRAG_j)) \tag{3}$$

Where *AT* is the allocation table for the fragments $AT_{j,k}$ is 1 if a fragment $FRAG_j$ is allocated to a particular site k; otherwise, it is 0. Then $CommCT_{proc}$ consists of two workings: one is transaction retrieval and other one is transaction update. It can be expressed as:

$$CommCT_{proc} = \sum_{k=1}^{m}\sum_{i=1}^{q} EXECFreq_{i,k} * (TranRet_i + TranUpd_i) \tag{4}$$

Where $EXECFreq_{i,k}$ is the frequency of the transaction Ti at site Sk. The costs for transaction update and retrieval is shown as:

$$TranUpd_i = \sum_{j=1}^{n} u_{i,j} *(\sum_{j=1}^{n} TA_{j,l} * \mathrm{C}_{\mathrm{Cost}}(TRANSM_{k,l}, \frac{SELi, j}{100} * size(FRAG_j))) \qquad (5)$$

$$Tran\,\mathrm{Re}t_i = \sum_{j=1}^{n} r_{i,j} * \min(\mathrm{C}_{\mathrm{Cost}}(TRANSM_{k,s}\,where AT_{j,s} = 1, \frac{SELi, j}{100} * size(FRAG_j))) \qquad (6)$$

The retrieval cost in eq. (6) depicts that all the sites with the fragment copies $FRAG_j$, only the site yields minimum transmission cost. The update cost in eq. (5) depicts all the transmission costs for remote sites.

3. Algorithmic Approach

This paper introduces a simple cost model which collects retrieval and update information at all the sites where queries can be executed. Then it examines the cost of allocating the fragments across heterogeneous sites. Here we choose the model of heuristic to store the fragment by sustaining the highest query cost. Finding allocation in this model is NP-complete since n fragments and m different sites. So there will be $(2^m - 1)^n$ formulations. Thus, we look for heuristic approach to solve this problem precisely.

A. Algorithm

The proposed algorithm is modeled in three ways; first, we attain allocation table AT by only considering the retrievals issued by transactions. Second, we consider update scenario. In the last, if a particular fragment has not been allocated to a site, and been updated by some transactions, we find its sites according to the update matrix. The time complexity of the proposed algorithm is within the bound $O(nm^2 p)$, where ***n*** represents number of distinct fragments, ***m*** is the number of sites, and ***p*** represents the transactions. A remote retrieval may not consume much communication cost, depending upon request access. The algorithm is depicted as follows:

```
INPUT:

Retrieval matrix (transaction, fragment);
Update matrix (transaction, fragment);
Selection matrix (transaction, fragment);
Frequency matrix ( transaction, site);

OUTPUT:
Allocation Table(fragment, site);

Function:
Cost (fragment, site);

Begin
$\cos t = \sum_{S_k \in S} \sum_{T_i \in T} FREQ(Ti, Sk) * Upd(Ti, frag) * C\cos t(\frac{SEL(Ti, grag)}{100} * size(frag));$
For Sk in S do

    begin
      Let n be the site which has network delay from Sk
      When Sk retrieves frag;
       If(n= site)

    begin
$T_1 = \sum_{T_i \in T} FREQ(Ti, Sk) * \mathrm{Re}t(Ti, frag) * C\cos t(\frac{SEL(Ti, grag)}{100} * size(frag));$
        Let $n_1$ be the site which has network delay from Sk
        When Sk retrieves frag;
$T_2 = \sum_{T_i \in T} FREQ(Ti, Sk) * \mathrm{Re}t(Ti, frag) * C\cos t(\frac{SEL(Ti, grag)}{100} * size(frag));$
      Cost = Cost + (T2- T1);

      end
   end

return cost;
End

Begin     /* Solution for retrieval */

For $T_i$ in T, $FRAG_j$ in F, $S_k$ in S do

if(RetMat(Ti, FRAGj)*FREQ(Ti,Sk) > 0)

 AT($FRAG_j$, $S_k$) = 1;

              /* Fragment removal */

For $FRAG_j$ in F do


 While(NumCopy(FRAGj) > 1)
       begin
          Let Sk be the site with AT(FRAGj, Sk)=1
       If ( Cost ( $FRAG_j$, $S_k$) > 0)
          AT($FRAG_j$,$S_k$) = 0
else
          break;
end

 /* If a frag has not yet allocated but is has updated by some
    transactions, select a least-delay site where the fragment
    is to be placed */

For $FRAG_j$ in F do

     If (Numcopy($FRAG_j$) = 0 and Updmat ($T_i$, $FRAG_j$) > 0)
     begin
          $S_k$= delay($FRAG_j$);
          AT ($FRAG_j$, $S_k$) =1;
        end
end
```

Figure 1. Psuedo code for the proposed Algorithm

4. Implementation and experimental results

To asses our model we implemented it on a bank employee relation which is depicted in table 5.And for simplicity we have considered only four sites of the distributed allocation as depicted in Figure 2.

TABLE5: Bank Employee Relation

Name	DOB	Job-Id	Salary	Location	Dept-Id
Rams	2/7/1982	Ap-12	2000	dhn	Cse-05
Chiru	14/6/1984	Acp-3	1750	dhn	Cse-05
Jana	12/1/1960	Prf-2	2750	klkta	Cse-05
Naick	4/2/1980	Ap-11	1400	rnch	Eee-03
Sash	2/2/1982	Ap-17	2000	wrgl	Min-17
Matho	22/4/1978	Acp-4	1500	rou	Ece-10
Korras	2/6/1984	Ap-19	1200	skl	Ece-14
Bankam	25/8/1979	Ap-5	1500	dhn	Cse-05

A. Matrix Construction

Suppose that we update attribute matrix (U), retrieval attribute matrix (R) with the predicates for attributes DOB (Date of birth) (P1: DOB <82, P2: DOB>82, P3: DOB = 82), salary (P1: salary > 1500, P2: salary < 1500, P3: salary =1500) and location (P1: location = “dhn”, P2: location =”klkta”, P3: location =”wrgl”) the cost is calculated in distance cost matrix (table 6) by eq. (5) and (6).

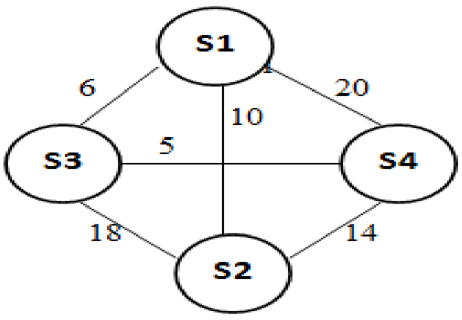

Figure 2. Network Sites

TABLE6: Distances with Cost Matrix

Sites	S1	S2	S3	S4
S1	0	10	6	11
S2	10	0	16	14
S3	6	16	0	5
S4	11	14	5	0

Applying our algorithmic model with eq. (4) in section II we obtain distance cost matrix (Table 6).

We consider the environment as hypothetical, which consists three transactions, three fragments, and four sites and the other consists three transactions, five fragments, and four sites. Though the environment is simple, the possible allocation combinations were 3931 in the former and 740099 in the later. Seven forms in which data were generated randomly for each environment were considered. The costs of both environments are depicted in Table 12 and Table 13, respectively. From the tables, we can see that cost of our algorithmic model close to that of the optimal allocation. Furthermore, our model performed better than Lin [10]. The rank percentage of our algorithm is also given in the tables. The rank of the algorithm could be easily obtained by calculating the costs of all possible combinations. It is interesting that Lin’s algorithm was not as good as what they claimed.

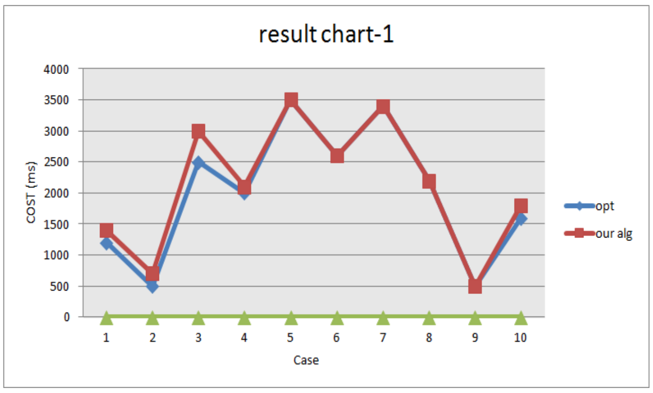

Figure 3. Three Fragments, Three Transactions, and Four Sites

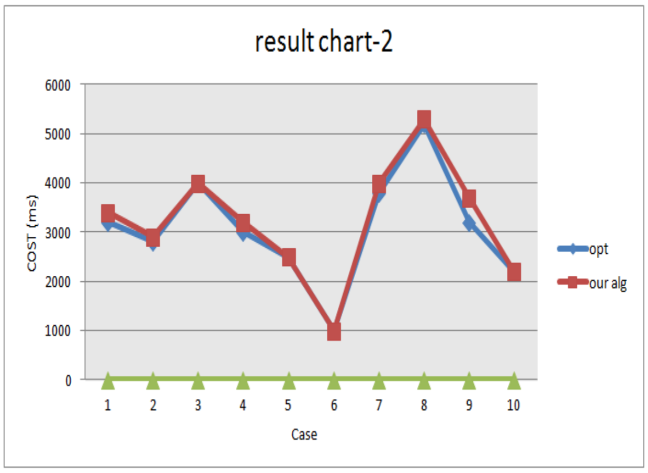

Figure 4. Five Fragments, Three Transactions, and Four Sites

Our algorithm also checks the data of a fragment copy will be updated at a particular site where a transaction is issued to complete its activities; that is, the more data in a fragment copy is updated accordingly, the earlier the fragment copy is removed from a site.

B. Experimental Analysis (Fragment Allocation)

According to the matrix construction of section IV (A), we obtain attribute frequencies from FREQN matrix as shown in table 7.

Predicate set was produced for the salary attribute with the highest cost in the Emp. Relation. P = { P1: salary > 1500, P2: salary < 1500, P3: salary =1500 }. According to this pridicate set P, the relation was gragented and allocated to all the corrosponding sites as depicted in table 8,9 and 10.

Predicate1: For this predicate the Retrieval cost < Updadate cost, then this fragment is allocated to site4, because it has the maximum update cost value according to this fragment (depicted in Table 8).

Predicate2: Retrieval cost > Update cost, then this fragment is allocated to all sites (depicted in table 9).

Predicate3: Retrieval cost > Update cost then this fragment is allocated to all sites (depicted in table 10).

TABLE 7: Attribute Frequency Table

Sites	DOB	Salary	Location
S1	34	60	44
S2	30	56	43
S3	85	72	64
S4	80	69	67

TABLE 8: Attribute Frequency Table

Name	DOB	Job-Id	Salary	Location	Dept-Id
Rams	2/7/1982	Ap-12	2000	dhn	Cse-05
Chiru	14/6/1984	Acp-3	1750	dhn	Cse-05
Jana	12/1/1960	Prf-2	2750	klkta	Cse-05
Sash	2/2/1982	Ap-17	2000	wrgl	Min-17

TABLE 9: Attribute Frequency Table

Name	DOB	Job-Id	Salary	Location	Dept-Id
Naick	4/2/1980	Ap-11	1400	rnch	Eee-03
Korras	2/6/1984	Ap-19	1200	skl	Ece-14

TABLE10: Attribute Frequency Table

Name	DOB	Job-Id	Salary	Location	Dept-Id
Matho	22/4/1978	Acp-4	1500	rou	Ece-10
Bankam	25/8/1979	Ap-5	1500	dhn	Cse-05

We state some experiments that were examined the cost formulas which truly reflect the communication cost in the real scenario. Due to lack of a WAN environment and the temperament of the delay by routers, we used an Ethernet-based local area network (LAN) to predict transaction mode on a WAN. This network consisted of 45 nodes. To reduce the execution time in the experiments, we only studied one case for each environment mentioned in the preceding subsection. The first experiment investigated case 4 in Table 12, and the second experiment investigated case 8 in Table 13. This result shows that the cost formulas utilized in our algorithmic model according to sites can truly reflect the cost in the real world. By using this we have shown the final allocation (Table 11) for the Employee relation which we have considered for various operational tasks.

TABLE 11: Fragment Allocation according to Sites

Fragment/Site	S1	S2	S3	S4
F1	0	1	1	0
F2	1	0	1	1
F3	1	1	1	1

We compare our algorithmic approach with a method of Lin. In every aspect of cost and rank , our approach is very near to the optimal and better out standing out comes compared to the method of Lin. All the factors have been stated in Table 12 and Table 13.

TABLE 12: Three Fragments, Three Transaction, and Four Sites

	Optimal	Our Algorithm		Lin Method	
Case	Cost(ms)	Cost	Rank (%)	Cost	Rank (%)
1	1235075	1378210	1.18	1695853	10.89
2	634722	648852	0.33	1313352	14.79
3	1247490	1327198	1.87	1589547	7.86
4	1147698	1873420	2.57	3064459	9.44
5	1594654	1549564	1.01	3373447	16.58
6	2557942	2452343	1.85	4243706	24.53
7	3625830	3635380	0.54	5288175	14.24

TABLE 13: Five Fragments, Three Transaction, and Four Sites

	Optimal	Our Algorithm		Lin Method	
Case	Cost(ms)	Cost	Rank (%)	Cost	Rank (%)
1	3349686	3371014	1.84	3512314	3.77
2	276045	2864441	2.98	3808433	7.00
3	4050135	4051074	1.48	5494376	13.65
4	2874733	3080416	2.73	4070848	7.24
5	2457967	2457976	2.54	5074256	16.86
6	1398089	1398870	4.04	4849245	28.61
7	3393788	3651177	5.22	6836547	35.84

5. CONCLUSIONS

In distributed database environment, relations are frequently distributed according to transactional needs, and allocated over HDDBs sites in order to minimize the network costs. Allocation of fragments to the sites is the challenging task. In this paper, for a fragment allocation for transactions, we have proposed a simple and widespread model that can reflect transaction behaviour in heterogeneous distributed databases. Based on the model and transaction description, our algorithmic model has been developed to find an optimal allocation with the minimized cost. Using our algorithmic model no additional complexity is considered in order to provide the effective communication. The results which are shown in this are very near to optimal one and are better than Lin. Some experimental tasks were also performed to verify that the cost formulas which can reflect the costs in the real world scenario. However, some constraints were not taken in the account here. At the first, an "efficient" algorithm to find an optimal allocation was not stated here. Second, the fragments geared by a transaction are not all independent to real world scenario, an issue we did not address. These issues are worth scrutinizing in the future.

6. References

[1] A. M. Tamhankar and S. Ram, "Database fragmentation and allocation: an integrated methodology and case study," *IEEE Transactions on Systems, Man, and Cybernetics- Part A: Systems and Humans*, Vol. 28, No. 3, 1998, pp. 288-305.

[2] M. S. Muro, T. Ibaraki, H. Miyajima, and T. Hasegawa, "Evaluation of file redundancy in distributed database systems," *IEEE Transactions on Software Engineering*, Vol. 11, No. 2, 1985, pp. 199-205.

[3] G. M. Chiu and C. S. Raghavendra, "A model for optimal database allocation in distributed computing systems," in *Proceedings of IEEE INFOCOM '90*, 1990, pp.

[4] 827-833.Bellatreche,L.,Karlapalem,K.,and Simonet, M. vertical fragmentation in distributed object database system with complex attributes and methods. (DEXA)(1996).

[5] M. K. Fisher and D. S. Hochbaum, "Database allocation in computer networks," *Journal ACM*, Vol. 27, No. 4, 1980, pp. 718-735.

[6] S. K. Chang and A. C. Liu, "File allocation in a distributed database," *International Journal of Computer Information Sciences*, Vol. 11, No. 5, 1982, pp. 325-340.

[7] M. T. ¨Ozsu and P. Valduriez, Principles of Distributed Database Systems. New Jersey: Alan Apt, 1999.

[8] S. Ceri, G. Martella, and G. Pelagatti, "Optimal file allocation in a computer network: a solution method based on the knapsack problem," *Computer Network*, Vol. 6, No. 5, 1982, pp. 345-357.

[9] M. Yoshida, K. Mizumachi, A. Wakino, I. Oyake, and Y. Matsushita, "Time and cost evaluation schemes of multiple copies of data in distributed database systems," *IEEE Transactions on Software Engineering*, Vol. 11, No. 9, 1985, pp. 954-958.

[10] X. Lin, M. Orlowska, and Y. Zhang, "On data allocation with the minimum overall communication costs in distributed database design," in *Proceedings of the Fifth International Conference on Computing and Information*, 1993, pp. 539-544.

[11] S. T. March and S. Rho, "Allocating data and operations to nodes in distributed database design," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 7, No. 2, 1995, pp. 305-317.

[12] L. W. Dowdy and D. V. Foster, "Comparative models of the file assignment problem," *ACM Computing Survey*, Vol. 14, No. 2, 1982, pp. 287-313.