Prof. Dr. H. A. Ali
Information System Dept. Mansoura, Egypt

Dr.Yehia A. El-Mashad
Dean of Delta Academy of science Mansoura, Egypt
ymashad@hotmail.com

ENHANCEMENT OF INFORMATION RETRIEVAL BASED-ON THE MOBILE AGENT

Abstract

As network information resources grow in size, it is often most efficient to process queries and updates at the site where the data is located. This processing can be accomplished by using a traditional client-server network interface, which ties the client to the set of queries supported by the server, or it requires the server to send all data to the client for processing. The former is inflexible; and the latter is inefficient. Transportable agents, which support movement of client computations to the location of the remote resource, have the potential to be more flexible and more efficient. Transportable agents are capable of suspending their execution, transporting themselves to another host on a network, and resuming execution from the point at which they were suspended. Transportable agents consume fewer network resources and able to support systems that do not have permanent network connections.

The purpose of this paper is to look at how the mobile agent paradigm can solve and improve the information retrieval-related process. This paper investigates two different approaches in achieving high performance information retrieval. The first approach utilizes the mobility of the agent in moving the query to the desired site where the data resides, while the second is based on reduction of the number of migrating agents. The two solutions are suite different areas for a wide range of applications. Experimental results indicate, however that the optimal performance of an agent is achieved by using agent migration.

Keywords: Mobile agent, Information retrieval, Data transmission, Distributed Data, data transfer rate

1. Introduction

Mobile agents are programs that can move through a network under their own control, migrating from host to host and interacting with other agents and resources on each site [1, 4, 12]. Such paradigms are particularly attractive in distributed information-retrieval applications. Mobile agents have several advantages in distribu-

ted information retrieval applications [2, 3, 13]. By migrating to an information resource, an agent can there invoke resource operations locally, thereby eliminating the network transfer of intermediate data and reducing end-to-end latency. Also, by migrating to the other side of an unreliable network link, an agent can continue executing even if the network link goes down, with this making mobile agents particularly attractive in mobile-computing environments [7, 8, 9, 14]. So it can be said that autonomous agents have the potential to provide a convenient, efficient and robust programming paradigm for distributed applications, particularly when partially connected computers are involved. Partially connected computers include mobile computers [4, 12]. Most importantly, an agent can choose different migration strategies depending on its task and current network conditions, and then change its strategies as network conditions change. Complex, efficient and robust behaviors can be realized with surprisingly little code.

Although each of these benefits is a reasonable argument for mobile agents, none of them are unique to mobile agents - and, in fact, any specific application can be implemented just as efficiently and robustly with more traditional techniques. Different applications require different traditional techniques, however, and many applications will need a combination of techniques. In short, the true strength of mobile agents is not that they make new distributed applications possible but, rather, that they allow a wide range of distributed applications to be implemented efficiently, robustly and easily within a single, general framework [3, 12, 15].

1.1 Migration

Mobile agents have several strengths. First, by migrating to the location of a needed resource, an agent can interact with the resource without transmitting intermediate data across the network, thereby conserving bandwidth and reducing latencies [9, 11, 15]. Similarly, by migrating to the location of the user, an agent can respond to user actions rapidly. In either case, the agent can continue its interactions with the resource or user even if network connections go down temporarily. Such features make mobile agents particularly attractive in mobile-computing applications, which often must deal with low-bandwidth, high-latency, and unreliable network links [7, 10, 14]. Second, mobile agents allow traditional clients and servers to offload work onto each other, and to change who offloads to whom according to the capabilities and current loads of client, server and network. Similarly, mobile agents allow an application to dynamically deploy its components to arbitrary network sites. Migration overhead is the time needed on the source machine in which to pack up an agent's current state and send the state to the target machine, plus the time needed on the target machine to authenticate the incoming agent, start up an appropriate execution environment and restore the state [5, 9, 11, 13].

1.2 Information Retrieving

Retrieving an information process from distributed database systems is an essential requisite in nowadays-distributed systems. In recent years, steady improvements in computer hardware and network technology have led to a dramatic increase in

information technology. Generally, it can be said that information retrieval and data collection are the most important requirements within information technology [12, 15]. The nature of currently available computing systems is pushing a lot towards a distributed approach, which assumes that computing resources and data are no longer located on one and the same machine instead, a migration of code and data is undertaken in order to speed up the as a whole execution process [2, 3, 5, 13]. The main objective of the most recent researches in this field is concentrated on increasing network utilization by:

1- Increasing the data transfer rate (developing high data transfer rate shared channels, enhancing network protocols to solve bottleneck and traffic problems).
2- Optimizing data transfer through the network (to minimize communication costs).

The main objective of this research is to introduce two approaches of data retrieving based on mobile agent technology. This work focuses on query processing execution using mobile agents by introducing an example of a simple distributed query, and gives with a full analysis of the different possibilities for accomplish this task. There is a discussion of the traditional approaches used for data collection and retrieving information. Then, two approaches as regards data retrieving will be proposed. The analysis of each approach is also discussed.

2 Problem Definitions

Assume that we have a homogenous distributed database system that includes database relations; these relations are fragmented vertically among different (n) sites, and each of these site issues (m) queries. For a special case, suppose that we have the following schema: TB1=Project (ID, Author, Paper); TB2=Project (ID, Publisher, Year of publishing). The database relation is distributed among the given sites as depicted in *figure 1*, where Site1: Table fragment TB; Site2: Table fragment TB2. There is then a query submitted by a user at a remote node (site3) requiring to ***"Find papers authored by 'H. A. Ali' and published by 'IEEE'"***

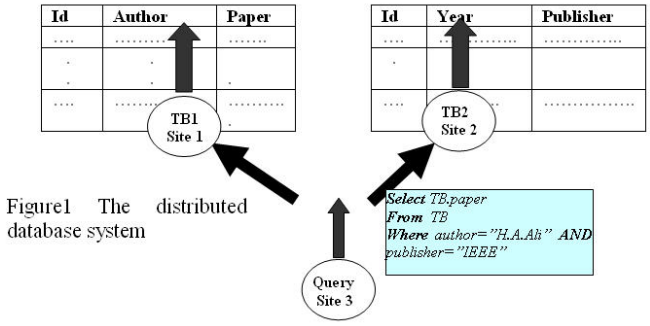

Figure1 The distributed database system

Figure 1

2.1 Traditional Approaches

Retrieving information from a distributed database stored at more than one site can be done in the traditional approach in two different ways [4, 6, 11]. First, move all the data from the sites to a central site; second, move the data with the lower size to the other. Each combination is an approach via which to retrieve information from all the sites, and each approach takes a different time compared to the other.

2.1.1 Moving data to a central site

In this approach the database tables TB1 and TB2 will be moved to a central site, then joined together at the central site, where the query will then be executed. The communication T1 time cost of the query can be calculated from the following expression [2]:

T(Communication Cost) = T(Transmit Data) + (TLogon + TLogoff) + T(Transmit results)

This in turn is given by the following expression:

$$T\,communication = \frac{(No.\ of\ TB1\ tuples + No.\ of\ TB2\ tuples) * tuple\ size}{Data\ bit\ rate} + Total\ Access\ Delay\ Time\ (seconds) + \frac{(No.\ of\ the\ final\ result\ tuples\ sent\ to\ Site3) * size\ of\ tuple}{Data\ bit\ Rate}$$

Assume that final size $S_f = 2^* S_t$, and $S_{t1} = S_{t2} = S_t$. The previous equations can thus be written as follows:

$$T_1 = \frac{(n_1 + n_2) * S_t}{f} + \frac{n_f * 2 * S_t}{f} + 2 * t_d\, Second \quad \text{.....(1)}$$

Where, n_1: Cardinality of TB1; n_2: Cardinality of TB2; S_t: Size of tuple (bytes); *f*: Data bit rate (bits/sec); t_d: Access delay of each database (seconds); and n_f: Number of final result tuples returned to site3.

2.1.2 move all tables to one of the given sites

The second approach is to move one of the two relations to the other site (according to the size of the table) and join them together, then to execute the query on the joined table. In such a case, the migrating direction can be determined by knowing the size; for the one with the smaller size will be migrated. Suppose that TB1 is the migrated table, and the communication time cost T2; we will then have:

$$T\,communication \frac{No.\ Of Moved\ Tubles * Tuple\ size}{Data\ Bit\ rate} + \frac{(No.\ of\ the\ final\ result\ tuples\ sent\ to\ site) * size\ of\ Tuple}{Data\ bit\ Rate} + Total\ Access\ Delay\ Time\ (seconds)$$

The previous equation will be:

$$T_2 = \frac{n_1}{f} * S_t + \frac{2*n_f}{f} * S_t + t_d$$

$$T_2 = \frac{n_1 + n_f}{f} * S_t + t_d \quad \ldots\ldots(2)$$

The communication time cost for the first approach is greater than the second one. The number of tuples in the two tables can be expected to be large, so the communication time will be large. Applying the mobile agent paradigm could be useful to avoid the transferring of unwanted data over the network - hence reducing communication costs, the network will also be kept free for more important data transfer.

3. First Proposed Approach

The first proposed approach is based on mobile agents for executing queries, which collect information from a number of relations located in different sites. This approach de-composes the joined operations, executed via the available relations, into a number of sub-queries. Where the generated sub-query can be encapsulated in a mobile agent, such as agent will move to the location of the data instead of trens-ferring unwanted data through the network.

3.1 Assumptions

One of the main topics of this study is mobility; i.e. it is not a goal here to develop an extravagantly distributed database system. Therefore we will need to make some simplicity assumptions which are: (1) All queries are read only (retrieving), so updating is not considered in this study. (2) Handling only select-join queries; as selects without joins are too simple to handle in a distributed network; also deletes and inserts introduce many consistency issues. (3) The query is based on a single relation, which is fragmented vertically only; and all attributes pertainig to all relations are stored on one site only. (4) There is a low system load. (5) High speed and low-loaded CPUs exists at each database system. (6) Execution times of the CPUs are very small relative to communication times. (7) Database systems on each site are relational DB type, that support standard relation language SQL.

3.2 Mobile Agents Paradigm

As in the previous example, site 3 sends a mobile agent carrying a query to site1; and the query is a sub query of the issued one. The query that is sent to site1 needs to find all papers authored by “H. A. Ali”. Its execution at site1 results in a number of tuples containing all papers are written by “H. A. Ali”. Then, at site1 the migrating mobile agent from site3 creates a number of mobile agents equal to the number of result tuples of the query executed; and, in turn, they migrate autonomously to site2. Each one containing information about a paper goes to site3 to sees if that

paper is published by “IEEE” and returns to site1 with the answer. Thus at the end, there will be information about papers written by the author “H. A. Ali” that are published by “IEEE” at site 1. Finally, the results will be sent back to site3.

$$T_m\ Communication\ = \frac{(Size\ of\ mobile\ agent) * (2+1)}{Data\ bit\ rate} + Total\ Access\ Delay\ time + \frac{(No.\ of\ the\ final\ result\ tuples\ sent\ to\ site\ 1) * size\ of\ tuple}{Data\ bit\ rate}$$

The communication time cost can be calculated via the previous expression, where:

- One agent will be moved from the initializing site (site1 in the example)
- Execution of the query encapsulated in the previous agent, resulting in a number (n_t) of tuples, which satisfy the first predicate.
- At this site, the number of mobile agents equal to n_t will be - generated equivalent to the number of resulted tuples at site 1.
- Each of the tuples will check the next site to see whether the corresponding tuple satisfys this given predicate.
- Each of these checks involves two messages (Query and response)
- The n_t mobile agents may be processed in parallel, so the time needed for n_t messages and n_t responses will be equivalent to “2”.

Assume that all mobile agents autonomously migrate to site3 at the same time, neglecting their time of creation at site2; thus, the total time of their transfer is equal to one mobile agent time of migrating and returning. Let the average mobile agent size be S_m, the previous equation can then be written as follows:

$$T_m = \frac{3*S_m}{f} + (n_t + 1)*t_d + \frac{2*n_f*S_t}{f} \quad \text{.....(3)}$$

Where T_m is the total communication time using mobile agents, n_t is the number of tuples coming from table TB1; S_m is the size of a mobile agent in bytes

$$\Delta T = T_1 - T_m$$

$$\Delta T = \frac{(n_1 + n_2)*S_t}{f} + \frac{2*n_f*S_t}{f} + 2*t_d - \frac{3*S_m}{f} - (n_t+1)*t_d - \frac{2*n_f*S_t}{f}$$

$$\Delta T = \frac{(n_1 + n_2)*S_t}{f} + 2*t_d - \frac{3*S_m}{f} - (n_t+1)*t_d$$

t_d can be neglected with respect to the term $n_t * t_d$

$$So \qquad \Delta T \cong \frac{(n_1 + n_2)*S_t}{f} - n_t*t_d - 3*\frac{S_m}{f}$$

$$\Delta T \cong \frac{(n_1 + n_2)*S_t}{f} - (n_t*t_d + 3*\frac{S_m}{f}) \quad \text{.....(4)}$$

2.2.1 A Comparison between the Proposed and Traditional approaches

In the next section we will study the benefits of using mobile a agent, which can be done by individually comparing traditional approaches with the mobile agent approach.

ΔT is the time difference between the two approaches, and from equation 4 it is clear that it is affected by the following: the number of tuples of relation 1 and 2, size of tuple, size of mobile agent, number of selected tuples from relation1, access delay time for each database, and the data bit rate. In order to achieve maximum benefits from the mobile agent approach, the time difference should be kept greater than zero - or the following inequality must be valid. Where

$$\Delta T \equiv \frac{(n_1 + n_2) * S_t}{f} - (n_t * t_d + 3 * \frac{S_m}{f}) > 0$$

$$\text{So,} \quad (n_1 + n_2) * S_t > (n_t * t_d * f + 3 * S_m)$$

The term $3 * S_m$ can be neglected with respect to the $n_t * t_d * f$

Let $DB = (n^1 + n^2) * S^t$

Then $DB > n^t * f * t^d$ .....(5)

If DB represents the database size it is clear from the previous inequality that, for large database sizes, mobile agent strategy is the best choice. Let's take a numerical example, as follows: $(n_1+n_2)_t$= 1,000,000 , f = 50,000 bit/sec, S_t = 400 bits, S_m = 250*8=2000 bit, t_d = 0.2 second, and n_t = 100. From equation 4 we can compute the time difference asΔT=7959 (sec), ΔT=2.21 hrs

3. An analysis of the proposed approach

Table 1 shows the communication Time Differences (CTD) between first approach and the mobile agent paradigm gives different database sizes

Table 1: CTD VS size of database

Database size $(n_1+n_2)*S_t$ (Kbyte)	CTD (sec)	
10	-38.4	
100	-24.12	
1000	120 (2 mins)	
5000	760	(12.67 mins)
10,000	1560	(26 mins)
50,000	7960	(2.21 hrs)
100,000	15960	(4.43 hrs)
500,000	79960	(22.21 hrs)
1,000,000	159960	(44.43 hrs)

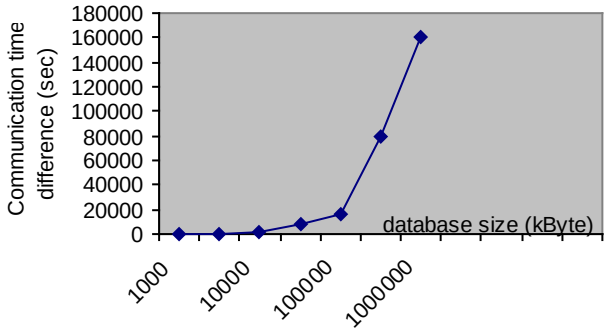

Figure 2: Communication time difference vs size of database

From *table 1*, we should to notice that: the first two rows contain negative values; this is because the database size is small (10-100kB). So using a mobile agent causes an „overhead" mobile agents of transferring themselves, in addition to the database access delay, though as the database size increases the mobile agent strategy results in a rapid decrease in communication time.

4. A Mobile Agent Paradigm for N sites

The previous approach can be generalized for a number of N sites, with each site containing a fragmented relation obtained from the vertical fragmentation of the global relation - and also the query, which will be executed as depicted in *figure 3*. The communication time for N sites for the first approach T_{1n}, can becalculated from the following equation:

$$T_{1n} = \frac{(\sum_{\substack{j=2 \\ j \neq i}}^{N} n_j) * S_t}{f} + \frac{n_f * S_t}{f} + (N-1) * t_d \text{second} \quad \text{.....6)}$$

Similarly, we are able to compute the communication time T_{mn} in the case of using mobile agents from the following equation:

$$T_{mn} = \frac{3 * S_m}{f} + ((N-2) * n_t + 1) * t_d + \frac{n_f * S_t}{f} \text{.....(7)}$$

ΔT can then be computed as follows:

$$\Delta T = T_{1n} - T_{mn}$$

$$\Delta T = \frac{(\sum_{\substack{j=2 \\ j \neq i}}^{N} n_j) * S_t}{f} + \frac{n_f * S_t}{f} + (N-1) * t_d - \frac{3 * S_m}{f} - (N-2) * n_t * t_d - \frac{n_f * S_t}{f} - td$$

$$\Delta T = \frac{(\sum_{\substack{j=2 \\ j \neq i}}^{N} n_j) * S_t}{f} + (N-2) * t_d - \frac{3 * S_m}{f} - (N-2) * n_t * t_d$$

$$Then \quad \Delta T \cong \frac{(\sum_{\substack{j=2 \\ j \neq i}}^{N} n_j) * S_t - 3 * S_m}{f} - (N-2) * n_t * t_d$$

$$let \quad DB = (\sum_{\substack{j=2 \\ j \neq i}}^{N} n_j) * S_t$$

$$So, \qquad \Delta T \cong \frac{(DB - 3 * S_m)}{f} + (N-2) * n_t * t_d \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (8)$$

For studying the effect of the database size, the number of sites "N", the data bit rate and communication network reduction on the time difference between two approaches, we will assume constant values for the following parameters (f = 50,000 bit/sec, N = 20, S_t = 400 bits, S_m =250*8=2000 bit, t_d = 0.2 second, and n_t = 100)

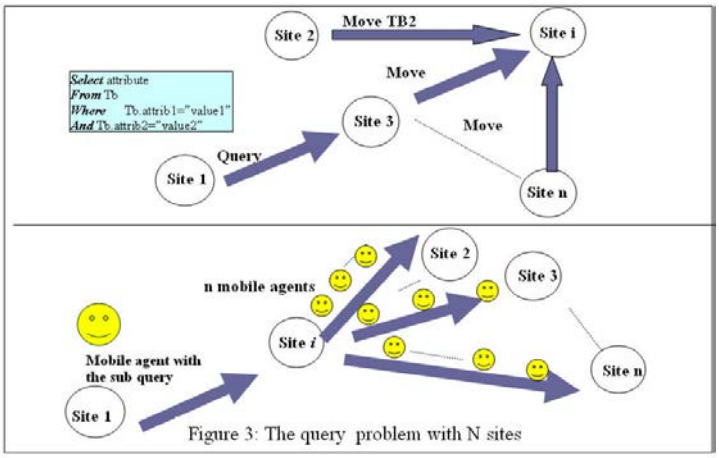

Figure 3: The query problem with N sites

4.1 Distributed Database Size Effect

The total number of tuples of the looked at relations – except that of the site initializing the query – can be an indication of the distributed database size. It is clear from equation 8 that the greater the number of tuples the greater the communication time difference – which means that use of the mobile agent approach will have a great benefit over the first type in a reduction of communication cost. So equation 8 can be represented as follows:

$$\Delta T \cong A * \sum_{\substack{j=2 \\ j \neq i}}^{N} n_j - B \quad \text{.....(9)}$$

where,

$$A = \frac{S_t}{f} \qquad B = 3 * \frac{S_m}{f} + (N-2) * n_t * t_d$$

This for large database sizes:

For the assumed parameters, we can compute the values of A and B with the total number of tuples = 1,000,000, and by using equation 9:

$$\Delta T \alpha \sum_{\substack{j=2 \\ j \neq i}}^{N} n_j \quad \text{.....(10)}$$

$$A = \frac{400}{50{,}000} \qquad B = 3 * \frac{2000}{50000} + (20-2) * (100) * .2$$

$$\Delta T \cong A * \sum_{i=1}^{N} n_i - B, \qquad \Delta T \cong 0.008 * 1{,}000{,}000 - 360$$

$$\Delta T \cong 7640 \quad Seconds \qquad \Delta T \cong 2.12 \; hrs$$

This value of the time difference is large, and it shows how mobile agents are very useful in executing this query; this emphasizes the truth of the proposed concept. On the other hand the mobile agent causes overhead time on the system for small database sizes, which is clear in the "-" sign in the equation, meaning that, for certain values of database size, the communication time difference will be negative. So it can be concluded that the first approach is better than the second, or, rather, mobile agents cause overheads in the system. So that the mobile agent approach succeeds the communication time difference "ΔT" should be greater than zero, i.e. ΔT > 0; and this can be shown in the following formula derivation:

$$\frac{(\sum_{\substack{j=2 \\ i \neq 1}}^{N} n_j) * S_t}{f} - (3 * \frac{S_m}{f} + (N-2) * n_t * t_d) > 0$$

$$\frac{(\sum_{\substack{j=2 \\ i \neq 1}}^{N} n_j) * S_t}{f} > 3 * \frac{S_m}{f} + (N-2) * n_t * t_d$$

$$(\sum_{\substack{j=2 \\ i \neq 1}}^{N} n_j) * S_t > \{3 * S_m + (N-2) * n_t * t_d * f\}$$

The left-hand side of the inequality represents the database size. The term $3*S_m$ can be neglected with respect to the second term because it is too large. The last inequality can thus be simplified as follows;

$$(\sum_{\substack{j=2\\ i\neq 1}}^{N} n_j) * S_t > (N-2) * n_t * t_d * f$$

According to our example, and by substituting the values of N, n_t, t_d, and f, the inequality will be as follow:

$$(\sum_{\substack{j=2\\ i\neq 1}}^{N} n_j) * S_t > (20-2) * 100 * 0.2 * 50{,}000$$

$$(\sum_{\substack{j=2\\ i\neq 1}}^{N} n_j) * S_t > 17.16 Mbytes$$

This value (DB size) is a small value for a distributed database system – hence we ensure that the mobile agent strategy is fit for large database sizes. In some cases, where the D.B size is small there will be an overhead of migration of mobile agents. Then it is preferable to use the traditional approach (moving data) to thereby minimize the time cost needed for transferal of data.

Table 2 and *figure 4* show the communication time difference as the distributed database relation size varies

Table 2: Distributed Database Vs CTD

Distributed relation Size (kbyte)	Communication time (seconds)		
	N=20	N=50	N=100
1,000	-196 (-3.3 mins)	-796	-1796
10,000	1278 (21.3 mins)	678.4	-321.6
50,000	7832 (2.2 hr)	7232	6232
100,000	16024 (4.5 hrs)	15424	14424
500,000	81560 (22.6 hrs)	80960	79960
1,000,000	163480 (45.4 hrs)	162880	161880
10,000,000	1638040 (455 hrs)	1637440	1636440

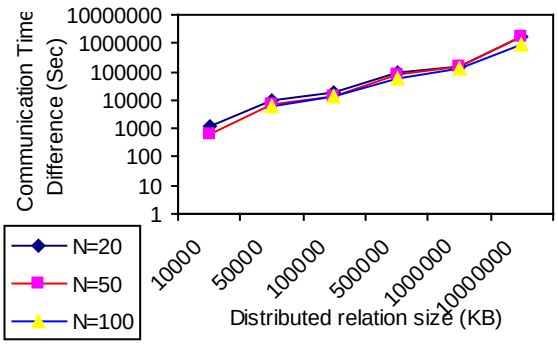

Figure 4: Distributed Database vs CTD

To study the effect of the network transmission speed on the communication time difference, equation 8 can be written as :

$$\Delta T \cong (\sum_{\substack{j=2 \\ i\neq 1}}^{N} n_j) * S_t - 3 * S_m\} * \frac{1}{f} - (N-2) * n_t * t_d \quad \text{.....(11)}$$

$$\Delta T \cong C * \frac{1}{f} - D$$

$$\textit{Where,} \quad C = (\sum_{\substack{j=2 \\ i\neq 1}}^{N} n_j) * S_t - 3 * S_m \qquad \textit{and} \qquad D = (N-2) * n_t * t_d$$

The second factor is the data bit rate or the network speed of transmission; and from equation 11 it is clear that the communication time difference is great for a large value of the ratio of database size to frequency; so we can say that the mobile agent approach well suits networks with low speed of transmissions or ones with large database sizes. Yet for very high-speed networksone should first see whether a mobile agent will be the thing most suitable to use or not. For the mobile agent strategy to be suitable, ΔT should be greater than zero, or we can say;

$$f < C/D$$

$$f < \frac{(\sum_{\substack{j=2 \\ i\neq 1}}^{N} n_j) * S_t - 3 * S_m\}}{(N-2) * n_t * t_d} < \frac{(\sum_{\substack{j=2 \\ i\neq 1}}^{N} n_j) * S}{(N-2) * n_t * t_d} \quad \text{.....(12)}$$

When we have tested our distributed database system - and if this inequality is not valid we can see that the mobile agent approach will not be the best choice. This may happen with small database sizes, with a large number of sites and also given a relatively very high speed of transmission. Assume the following distributed database system: N=20, t_d=0.2 seconds, n_t=100 tuples, and S_m=2000 bits;

$$(\sum_{\substack{j=2 \\ j\neq i}}^{N} n_j) = 100{,}000{,}000 \quad \text{tuples} \qquad\qquad S_t = 400$$

$$\textit{So,} \qquad (\sum_{\substack{j=2 \\ j\neq i}}^{N} n_j) * S_t = 400 * 10^9 \, bits$$

$$\Delta T \cong \{400 * 10^9 - 3 * 2000\} * \frac{1}{f} - (20-2) * 100 * 0.2$$

$$\Delta T \cong 400 * 10^9 * \frac{1}{f} - 360$$

Substituting in equation 11

Table 3: The Effect of Varying Network Speed on CTD

Speed of Network **(Kbit/sec)**	**Communication Time Difference (seconds)**		
	Relation size =46.5 GB	**Relation size =1 GB**	**Relation Size=500 MB**
10	39062140	858633	419070
50	7812140	171439	83526
100	3905890	85539	41583
200	1952765	42589.6	20611
500	780890	16820	8029
1,000	390265	8230	3834
500,000	421.25	-343	-351
1,000,000	40		

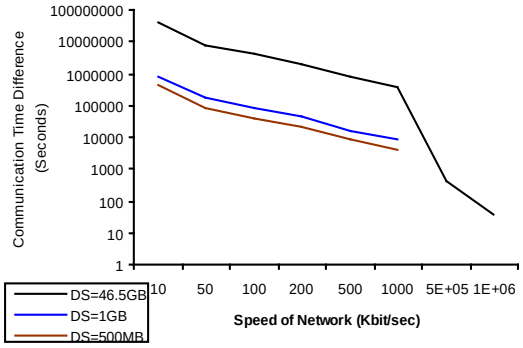

Figure 5: Effects of Varying Network Speed

Table 3 and *figure 5* show the effects of changing transmission speed on the communication time difference that exists between the first approach and the mobile agent approach, gives with different values for the distributed relation size. It is clear that the speed of transmission has a great effect on the communication times of both strategies, and the speed of transmission variations may be due to traffic load over the network, the load over the network communication, in addition to the being various network types with different speeds of transmission.

4.3 Number of Sites Effect

To study the effect of the number of sites (Ns) on CTD, equation 8 can be put as follows:

$$\Delta T \cong \{\frac{1}{f} * [(\sum_{\substack{j=2 \\ i \neq 1}}^{N} n_j) * S_t - 3 * S_m] + 2 * n_t * t_d\} - (n_t * t_d) * N \quad \ldots\ldots\ldots\ldots\ldots\ldots (13)$$

Substituting in equation 13 using previously distributed database parameters and *f= 50,000 kbit/sec* gives the results in table 4. It is clear from both *table 4* and *figure 6* that the number of sites having an effect the communication time differences is able to be neglected for very large database sizes. As the database size gets smaller, the number of sites will have an increased effect on the communication time –and, as it increases, the trend will move away from the mobile agent approach because of the overhead caused by the migration of mobile agents in addition to database access delays.

Table 4: Effect of Number of Sites on the CTD

Number of sites	**Communication Time Difference (seconds)**			
	DRS=46.5 (GB)	DRS=500 **(MB)**	**DRS=100 (MB)**	**DRS=20 (MB)**
5	8000000	83846	16717	3295
50	7999000	82946	15817	2395
100	7980000	81946	14817	1395
150	7997000	80946	13817	395
250	7995000	78946	11817	-1604
400	7992000	75946	8817	
550	7989000	72946	5817	
750	7985000	68946	1817	
1000	7980000	63946	-3182	

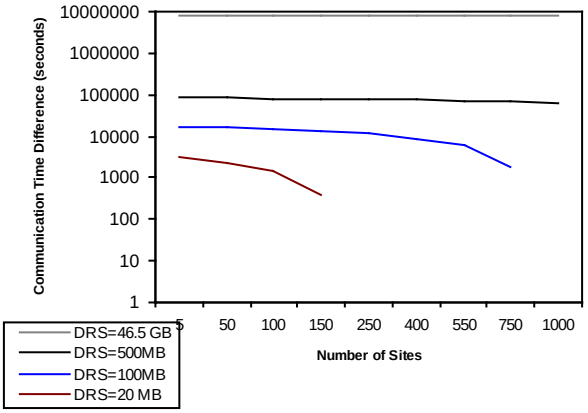

Figure 6: Effect of Number of Sites on the CTD

5. The second proposed Approach

The above-proposed approach assumes a low system load and low traffic in the communication network. As the network load gets higher, the proposed approach

performance will be greatly affected -and the total execution time of a query can be expected to become higher than the best selected plan from the distributed query optimizer. Hence the mobile agent paradigm will cause overhead on such a system. The migration of a large number of mobile agents on the communication network may be faced with collision problems, which will in turn affect the performance of the chosen proposed approach.

The second proposed approach is based mainly on reducing the number of migrating agents while also resorting to the transmission and reception of messages between mobile agents. This approach can be applied in a case of parallel joins on different sites, and can be used after the query optimizer has chosen the best global plan. The distributed query optimizer will decide the best plan via which to execute the query. The approach then replaces the links in relations, located on different sites, a number of mobile agents. These agents will communicate via messages to completely determine the number of tuples required and to build the required relations. The mobile agents finally transfer these results to the site, where the optimizer will choose whether to continue the query execution.

5.1 Second Approach Analysis

Before a query enters our system the query is de-composed into sub-queries. With each query there could be a number of joint execution orders for that query. Our pre-processing approach divides up each of the joint execution orders into a separate plan, one that contains the joint order as well as vital database statistics (relation size, tuple size, location of relation, etc.). We chose to implement the distributed system approach as presented in [3, 12] as the basic idea of the chosen approach to thereby develop a query optimizer making appropriate modification when necessary.

The System query execution optimizer decides [a] The best way to access a single relation, and [b] The best way to join two relations, the best way to join three relations, and so on, until all the relations have been joined).

At each stage, all permutations are scored, with the cheapest plans being kept for the next time. The end result is a sequential ordering of the sub-joins. The problem arises when these subjoins include the joining of relations that are vertically fragmented or distributed on different sites; relations sholud be transferred to a global site to thus execute the joining operation. This includes the transferring of unwanted data through the network.

Our approach suggests that after the query optimizer estimates the best plan, if it contains a number of joins on the distributed relations, it is to be based on when we are able to know the number of tuples needing to be transferred through the network, that is before execution of the joining operation. This can be achieved via mobile agents, where at the initiating site (the sender of the query) the query optimizer creates a number of mobile agents that are encapsulated with sub-queries of the original one. Each sub-query is aimed at a relation at a different site, where it is to be executed there at the remote site. A number of tuples is generated at each site, which contains a part of the required tuples. The rest are distributed over the other sites. The joining operation implies two relations at different sites; and every mobile agent in

one relation should know its partner at the other site so a communication can be established between them.

First of all, the one with the smaller relation size will be known as the master - and the second as slave. The master agent accesses its database and performs a projection on the first attribute of the joining operation, and gets its elements. Then, it sends a message containing one element to the slave agent. The slave agent in turns compares this element with the second attribute of the joining operation elements to thereby select the matching tuple. Then, sends a responding message to the master agent ask it whether the element it sent matches one of the elements in the second relation -and the master agent will then determine whether to add that tuple to the results or not. The process continues till all the elements of the first attribute have been tested with those of the second attribute. Finally, there will be a number of tuples existing on the two sites, and they together constitute the required tuples; the optimizer will subsequently decide where to transfer both results to continue execution of the query.

This approach is similar to the first one except that the migrating mobile agents are replaced with only two mobile agents communicating with each other. Such as approach can be added to the distributed query processing model described in [3, 12], where the query optimizer will have several plans to follow and the plan with the lowest cost will be the best plan. Estimating the cost of the proposed mobile agent approach and comparing it to the best plan suggested by the optimizer can achieve this. Where the joining operation may take less time than the mobile agent approach (this can exist when the optimizer decides to move the smaller relation to the other site and process the joining operation there).

Cost Model

In order to decide whether to use this approach or not, the optimizer should estimate its cost as:

Cost = CPU Cost + Communication Cost

The CPU cost is neglected in this study; hence the total cost will be equal to the communication cost. T (Cost) = T (Transfer) + Access Delay Time

=T(Transfer of mobile agents) + T (Transfer of Messages) + T(Transfer of tuples) + Access Delay Time

For one joining operation: Cost = Cost (migrating two mobile agents + accessing two relations) + Cost (sending messages + accessing the second relation) + Cost (receiving messages) + Cost (transferred tuples)

$$Cost = \left(\frac{2 * S_m}{f} + 2 * t_d\right) + n_m * \left(\frac{S_z}{f} + t_d\right) + \frac{n_m * S_z}{f} + \frac{n_r * (S_{tm} + S_{ts})}{f}$$

Where, S_m = the average size of a mobile agent (byte), f = network speed of transmission (byte/sec), t_d= Database average access delay time (sec), n_m= number of master relation tuples, n_r= total number of produced tuples on both sides to be

transferred to another site so as to complete the query execution, S_z= size of message (in bytes), S_{tm}= size of master relation tuples (byte), and S_{ts}= size of slave relation tuples (in bytes). By knowing the number of tuples in each relation; we can say that n_m= % Unique * Card (R1).

The number of resulting tuples can be estimated via the query optimizer, so the query optimizer is able to estimate the time cost of this approach and compare it to the time cost of a single join, and then to see if the proposed approach cost is lower or higher, then decide which plan to choose for the query execution.

6. Conclusions

This paper has presented two approaches via which to retrieve information a the distributed database. The first approach is based on mobile agents to execute queries, which collect information from a number of relations located at different sites; while the second is based mainly on reducing the number of migrating agents while resorting to the transmission and reception of messages between mobile agents. The validation of the first one has been demonstrated with an example. Both of the approaches show that the mobile agent technique should be seen as an *alternative approach* to the client-server traditional architectures. For the management of distributed resources, a comparison between a client-server solution and a mobile agent-based approach shows that mobile agent technology offers important advantages, such as flexibility and the scalability of the system, load balancing, on-demand services, low traffic in the network, and many others. These benefits are due to the way in which mobile agents treat distribution problems by using local interactions and mobile logic. Applying the mobile agent paradigm can thus be useful in avoiding the transfer of unwanted data over the network and, hence, in reducing communication costs, thereby keeping the network free for more important data transfers.

7. References

[1] St. Arbanowski, M. Breugst, I. Busse, T. Magedanz “Impact of Standard Mobile Agent Technology on Telecommunication”, 1997.

[2] R. Ahmad, S. Rahimi, L. Gandy, D. Ali, “A Multi-Agent Information Retrieval System,” Proceedings of the 10 International Conference on Industry, Engineering and Management Systems, Decision Support Systems, Cocoa Beach, Florid, pp. 52–57, 2004.

[3] J. Bjursell, S. Rahimi, D. Ali, M. Cobb, “Mobile Agents’ Applicability for Information Retrieval and Processing,” proceedings of the 9th Annual International Conference on Industry, Engineering, and Management Systems, Florida, pp. 235–242, 2003

[4] Bobak “Distributed and Multi-Database Systems”, Artech House Inc. 1996.

[5] A. Corradi, C. Stefanelli, F. Tarantino, ”How to employ mobile agents in system management”, Proceedings of the Third International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology, 1998

[6] G. Colin Harrison, D. M. Chess, and A. Kershenbaum, “Mobile Agents: Are they a good idea? Technical report”, IBM Research Division, T. J. Watson Research Center, March 1995. Available at http://www.research.ibm.com/massdist/mobag.ps.
[7] C. Demartini, R. Iosif, C. Raibulet, J. P. Thomesse ,“A DBR-based Approach for System Management”, Proceedings of the Field bus Conference FeT'99 Conference in Aagdeburg, September 23–24, 1999, pp. 437–444
[8] G. FOKUS “ MASIF: Mobile Agent System Interoperability Facilities Specification”, ftp://ftp.omg.org/pub/docs/orbos/97-10-05
[9] R. S. Gray, G. Cybenko, D. Kotz, and D. Rus, “Mobile agents: Motivations and State of the Art”, Handbook of Agent Technology. AAAI/MIT Press
[10] A. Krovi and S. Rahimi, “A Distributed Approach to Content-based Image Retrieval,” The 2003 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA’03), Las Vegas, Nevada, pp. 458–463, 2003.
[11] J.G. Lee, J. Y. Kang, E. S. Lee, “ICOMA: An Open Infrastructure for Agent-based Intelligent Electronic Commerce on the Internet”, Proceedings of the International Conference on Parallel and Distributed Systems, 1997, pp 648–655.
[12] T. P. Ng. “Optimal Data Migration Policies in Distributed Databases.” Proc. 15th Int. Computer Software and Applications Conf., 1991
[13] S. Rahimi, J. Bjursell, D. Ali, M. Cobb, “Preliminary Performance Evaluation of an Agent-based Geospatial Data Conflation System,” proceedings of The IEEE nternational Conference on Intelligent Agent Technology (IEEE-IAT 2003), Halifax, Canada, pp. 550–553, 2003.
[14] M. Tamer Ozsu, “Principles of Distributed Database Systems”, Prentice Hall, 1991.
[15] P. Wahjudi, S. Rahimi, D. Ali, M. Cobb, “Hybrid Agent: Providing an Integrated Network Mobile Agent Environment,” proceedings of 9 the Annual International Conference on Industry, Engineering, and Management Systems, Florida, pp. 231–234, 2003.