TRACKING MIGRATORY DATA IN DISTRIBUTED ENVIRONMENTS

Dr. U.S. Pandey (Associate Professor , University of Delhi)
Mr. Jayant Sharma (M.phil, M.K. University, Madurai)
Ms. Twisha Dingra (M.Tech, YMCA, Faridabad)

ABSTRACT

Before the emergence of the broadband networks like ATM (Asynchronous Transfer Mode), active migration of data resource like databases and annals was a very costly operation and the performance was a big issue. The amount of data transmitted via conventional networks was the heaviest concern in the narrowband environments. Minimization of the size of the data was the most important aspect for performance improvisation. But with emergence of broadband networks, data transmission operations become very cost effective and the performance factor has improved dramatically. In broadband networks, proper and efficient use of bandwidth is the most important factor for performance improvement. In such environments, we can transmit the complete data resource easily. So the tracking of position of data resources becomes very important in broadband networks where we migrates data resources. In this paper, we discuss various position tracking techniques. Some of the techniques, which are discussed here, are based on conventional methods. Some of them use the feature of ATM networks. Here, we propose a new technique, which use the various features of ATM networks. We also compare the performance of proposed technique with one of existing technique, which also uses various features of ATM networks. This comparison will show that the proposed technique is optimal for given set of conditions.

Key Words: ATM (Asynchronous Transfer Mode), Tracking, Resource Migration

INTRODUCTION

The bandwidth requirements for data traffic within commercial organizations have been increasing steadily for some time, both in the local area networks and in the wide area networks. Workstations have been used to introduce multimedia applications to the desktop, including components of voice, video and image, besides growing amounts of data. This development requires networks of greater bandwidth than commonly present today with the capability of handling multiservice traffic on the same network.

One of the techniques for high-speed internet works is Asynchronous Transfer Mode (ATM) which is being developed for public networks capable of supporting many classes of traffic. ATM is capable of supporting a wide range of traffic. It is used for mobile computing and multimedia application [1, 2, 3], such as voice, video and image, on a large scale. Distributed processing applications are also supported by this technology very well.

ATM is a high-speed, packet switching technique that uses short fixed length packets called cells. Fixed length cells simplify the design of an ATM switch at the high switching speeds involved. The selection of a short fixed length cell reduces the delay and most significantly the jitter (variance of delay) for delay-sensitive services such as voice and video.

ATM will no doubt be the Local Area Network (LAN) and Wide Area Network (WAN) technology of the future. It only makes sense to use fixed-length data cells and to carry these cells from one user to another. The effective and efficient transfer of data from one point to another can be accomplished by using one of many technologies and ATM is without a doubt the most effective and efficient way to do this. Once the costs of ATM interface boards for PCs, ATM switches, bridges, hubs, and routers have reached an affordable level, network managers will jump on the ATM bandwagon with both feet.

Data from one user's PC will basically travel the entire network path to the other user's PC with very little interaction from software processes. Hardware and firmware processes do most of the processing and routing of data in ATM networks, and this is what makes the movement of data so fast and efficient. As the line between what constitutes a LAN or a WAN becomes blurred, it is ATM that will bridge the gap and provide network managers with a one-technology solution to all of their network traffic. This is happening rapidly, and many vendors are poised to begin selling ATM hardware and equipment. ATM will become the network topology of choice.

There are many different parameters to measure the performance of network system. To improve the performance of the system, we try to manage some of the parameters. In narrowband environment, lower volume of transferred data was one of the major parameters of performance improvisation. But in broadband networks like ATM, as the bandwidth is very high, efficient use of this bandwidth is one of the major factors to improve the performance of system. Because of the very high bandwidth, the transmission delay is very small even when data of huge volume is transmitted. So, to utilize the bandwidth of broadband systems properly, large volume data should be transmitted instead of small chunk of data volumes.

With this point in mind, migration of full data resource like database is identified as one of the most useful mechanisms in distributed environments. Several different approaches were followed in the narrowband system for minimization of load balancing [4, 5, 6, 7, 8, 9]. But all of these approaches proved to be very costly. On the other hand, the case in broadband environments, like ATM, is different. As the data transmission cost is very low in comparison to narrowband environments, resource migration can be employed for various purposes.

Database migration can be considered as one of the major database operation. It is because of the reason that time consumed in the transaction processing at remote machine [10, 11, 12] is very high in many situations. It is also shown in these papers that the time consumed in transaction processing is very small if we collect the necessary databases in the transaction initiation site (The site which performs various transactions on databases) through database migration. Let us take one example of this scenario. Let a transaction access a database of 50 megabyte at any site Z from any other site, which is at a distance of about $1.5*10^7$ meters and connected by an optical fiber of bandwidth 1.5GB.

Considering the speed of light equal to $2.1*10^{8}$ m/s, propagation delay will be at least $7.1*10^{-2}$ sec. even after ignoring all other delays and times. On the other side, the total time to migrate the full database is just $2.6*10^{-1}$ seconds. So if more then 4 times a transaction is occurred, the time required is $2.84*10^{-1}$ seconds. So, it is batter to migrate the database on the transaction initiation site. By this way, we can say that with broadband technology systems, database migration is one of the most important operations in distributed environments.

Now, the next demand, which come to the picture with the use of database migration in global network environments, is the continuous tracking of the each data resource. Various tracking techniques are used in different fields like mobile computing (e.g. [9,13, 14, 15, 16, 17, 18, 19, 20]) etc. These techniques are also applicable for tracking of data resource in distributed environment. But, all the above mention features of ATM network are not used by these mobile computing techniques, so more effective and sufficient techniques are required.

The problem with these mobile computing techniques is they require many number of message to track the position of the data resource. The techniques given in [20] use the various features of ATM network. These techniques use much shorter number of messages as compare to mobile computing technique.

In this paper, we will discuss several resource tracking techniques, which are already proposed, and then we will propose a new technique, which utilize the features of ATM network. After that, we will compare this technique with one of the existing similar kind of technique on the bases of simulation study. We evaluate the performance of these two techniques under several system parameters like scale of network and frequency of access of data resource.

The rest of the paper is organized as follows: In section 2, we present the previously discussed resource tracking techniques. New technique for resource tracking is proposed in third section. Section 4 will explain the simulation environment used for the two algorithms discussed in this paper. Section 5 will show the simulation result & corresponding graph plotted with the help of these results. Finally we will conclude this paper and discuss further work in section 6.

TECHNIQUES FOR RESOURCE TRACKING

This section of paper has been divided into two sub sections. The first sub section is titled with **Network Model,** which will describe the network model considered in this paper. Then, the second sub section is titled with **Techniques for tracking of migratory data.** This will discuss the various migratory data tracking techniques, which are already proposed. These techniques are conventional techniques which are used already either in distributed environment or in mobile environment.

Network model

Here, we assume is the basic ATM network environments. In ATM network, a connection is established with the destination site in advance before the communication is started. This network is a connection-oriented switching system. The connection is known as a virtual channel in the ATM

naming conventions. There are two types of connections: PVC (Permanent virtual channel) and SVC (Switched virtual channel).

PVC is established permanently for certain groups of ATM switches whereas SVC is established dynamically according to a request by a site. The advantage of an SVC is that, it released if the network gets congested or the connection is not used for a predetermined period of time. A connection can be a point-to-point connection or as a point to multipoint connection. Both these connection i.e. PVC and SVC can established in either of the two ways. The figure given below shows the setup for these connections.

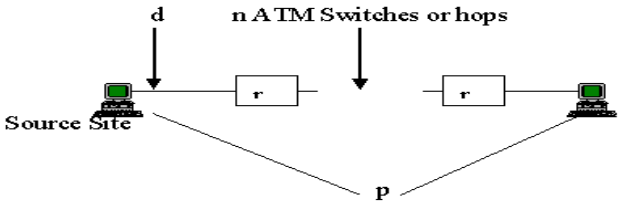

Fig 2.1: Message Transmission

Now, the communication time for the connection which is already established is:

Total communication time [T(n)]= Total routing time at all hops + the total processing delay at end sites + the total propagation delay between the end sites**.**

This is shown by the following equation:

$$T(n) = nr + 2p + (n+1) \quad \ldots (1)$$

Where,

n = Number of hops between two sites.

d = constant propagation delay between any two hops.

p = Total processing delay at one site.

r = constant routing time at any hope.

The assumption here is that the size of the message is small enough in comparison to the bandwidth of the communication channel so that the total transmition delay of message is negligible.

The communication time for the connection that is not already established and only establish at the time of start of the communication is different. Here we will also consider the time to setup a connection, which is known as connection establishment delay. The following are the steps to setup a SVC point to point connection:

1) First, a connection request signal will propagate from the source site to the destination site while establishing a connection.

2) Secondly, after the signal gets to the destination, the destination determines weather it will accept or reject the connection request.

3) Finally, if the destination site accepts the request, it will send the connection conformation signal to the source site through the same route, which has been established by first step.

So, the connection establishment time for the SVC connection through n hops is:

Total connection establishment time [C(n)] = Total routing time at all hops along with the total configuration time at all hops + the total processing delay at end sites +twice the total propagation delay between the end sites**.**

This is shown by the following equation:

$$C(n) = n(r+R) + 2p' + 2(n+1)\,d \qquad \ldots (2)$$

Where,

p' = Total processing delay at one site

R = Constant time for route configuration at each ATM switch

The figure given below shows the setup for connection establishment.

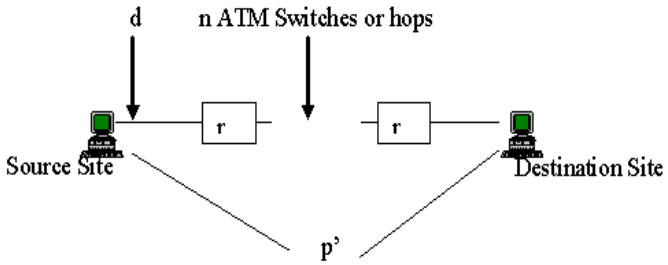

Fig 2.2: Connection establishment

So the total time required to send the message through a SVC is:

Total message transmission time [T'(n)] = Connection establishment time + Message transmission time

i.e. $T'(n) = C(n) + T(n) \qquad \ldots(3)$

Existing techniques for tracking of migratory data

Several techniques are proposed for the tracking of migratory data out of which we will discuss nine techniques here, which are proposed in [20]. Three of them are based on message broadcast and the other two are the techniques, which employ a specific router, called the default router, to maintain the information of the current position of a data resource. In next two techniques, one is based on the Pointer, which was proposed for management of data resource position, and the other based on the method, which was proposed, for the management of data items position in distribute systems. And then the last two techniques are the extension of the sixth and seventh technique. They used the basics of previously discussed techniques along with the advance features of the ATM network. The various techniques with their classifications are as follows:

Some of these techniques were initially proposed for mobile environments, which were then converted for distributed environments in [20]. There are many proposed resource tracking techniques for mobile network but only those were consider in [20] which can be easily applied for

resource tracking because the data resource consider here is migratory in nature with the same nature as that of mobile host.

Various terms used: We are using the following terms in this paper to describing these techniques.

Objective site

Operation kick off site: It is the site, which issues an operation-request message.

Objective resource: It is the data resource to be accessed.

Target site: It is the site, which holds one of the Objective resources.

1 The BN technique:

Every time, broadcast is the medium to report the new position of a data resource, when a data resource is migrated from one site to another site. By this way, every site of the network always knows the exact position of each data resource. Any site can directly send an operation request message to the target site, when it wants to access certain data resource.

2 The DF technique:

Only the movement-tracking server is notified of the new position with every migration of a data resource. Thus, the exact position information of each data resource is along movement-tracking server. When the process kick-off site needs some data resource, it first sends the operation request message to the movement-tracking server and then the server forwards it to the target site.

3 The DQ technique:

Like DF technique, only the movement-tracking server is notified of the new position with every migration of a data resource. When the process kick-off site needs some data resource, it first sends the operation request message to the movement-tracking server. Now, the difference from DF is that here server does not forward the request to the target site but it sends back the information of target site to process kick-off site. Then process kick-off site send the request of data resource to the target site.

4 The BF technique:

In this technique, no server concept exits. Neither any site is notified of the migration of the data resources. This technique uses the concept of broadcasting. Operation request has been broadcasted by process kick-off site and then only the target site respond to this message and the other site simply discard it.

5 The BQ technique:

Like BF, this technique also uses the concept of broadcasting. Here as well no movement-tracking server exists. Instead of operation message, a query message has been broadcasted by process kick-off site to every other site to learn and identify the position of the objective resource. And then operation request has been sent by process kick-off site to target site.

The next two techniques are known to be as CF technique and CQ technique respectively. In these techniques, an operation-request message is successively forwarded along the migration track of the Objective resource has raised. In CF and CQ, each site initially has an information table called lp-table (the local position table) in which the initial position of each data resource is maintained. Then the lp-table has been revised locally at each site according to some rules known to be as update rules. These update rules say that if the data resource D_l migrates from site S_i to site S_j, the current position of D_l in lp-table at both S_i and S_j are updated from S_i to S_j. and if Operation kick off site S_i sends a message to site S_j to access data resource D_l according to its own lp-table but finds that D_l currently reside at another site S_k, the current position of D_l in the lp-table at S_i is updated from S_j to S_k.
Note that lp-table at each site is not always up-to-date. The way of accessing the data resource in CF and CQ are expressed as follows.

6 The CF technique:

As every sits has its own lp-table, when a site need some data resource suppose D_l, it sends an operation request message according to the entry in its lp-table against data resource D_l. Then, if the site that receives this request does not hold the data resource D_l, it forwards the request message according to the entry in its own lp-table. In the end, after consecutive message forwarding, it finally reaches the Target site. Then the operation is performed on D_l, and the result together with current position of D_l (i.e. the address of Target site) is sent from the Target site to Operation kick off site (using a switch virtual connection which is directly established from the Operation kick off site and the Target site).

7 The CQ technique:

Like CF, first an operation request message has been send by the operation kick-off site to the site whose entry exists in its lp-table against the required data resource say D_l. Then if the site, which receives the message, does not hold the entry for data resource D_l, it sends information of D_l in its own lp-table to the operation kick off site. By this way, the operation kick off site update the entry for D_l in its own lp-table based on the reply message, and sends the operation-request message again to the new position of D_l. This process continues until the Target site is found.
The main problem with CF and CQ is that these require many numbers of messages along with the tracking of data resources, if resource migration occurs frequently. Because of this reason, the communication time in these techniques is very long and the performance deteriorates because of this reason. The technique is given in [20] to overcome this problem. The paper is using the following feature of the ATM network to overcome this problem.
1) The transmission delay in broadband network is negligible for small volume of data because of the broader bandwidth.
2) An SVC connection in ATM network is released only because of two reasons; either when it is not used during certain predetermined period of time or when congestion occurs.

Taking these two facts into account, two other techniques were proposed. One is ECF and another is ECQ. The basic idea of ECF and ECQ is same as that of CF and CQ respectively. In these techniques, in addition to all the things, the resource migration count is recorded. Every time, the data resource migrates, the value of resource migration count increase. Therefore, the value i means the position tracking information represents the position of the data resource after i-th migration. We can determine the newer position value by comparing these values. The modification regulation of lp-table for these algorithms is customized accordingly. Update rules states that

I) If the data resource D_l migrates from site S_i to site S_j, the current position of D_l in lp-table at both S_i and S_j are updated from S_i to S_j. And there resource migration count is also updated to the value one greater then that of S_i. and,

II) If a site receives a message including the lp-table information, the resource migration count is in its own lp-table and that of lp –table in the message are compared. If they are not the same the older resource information is updated to newer one.

8 The ECF technique:

Similar to CF, when a site need some data resource suppose D_l, it sends an operation request message according to the entry in its lp-table against data resource D_l. Then, if the site that receives this request does not hold the data resource D_l, it forwards the request message according to the entry in its own lp-table. In the end, after consecutive message forwarding, it finally reaches the Target site. Then the operation is performed on D_l, and the result together with current position of D_l (i.e. the address of Target site) is sent from the target site to operation kick off site via the path along witch the message have been sent from operation kick off site to target site. And the lp-table at each site is modified according to modification regulation.

9 The ECQ technique:

Like ECF, first an operation request message has been send by the operation kick-off site to the site whose entry exists in its lp-table against the required data resource say D_l. Then if the site, which receives the message, does not hold the entry for data resource D_l, it sends information of D_l in its own lp-table to the operation kick off site. By this way, the operation kick off site update the entry for D_l in its own lp-table based on the reply message, and sends the operation-request message again to the new position of D_l. This process continues until the Target site is found. Then, the update of the lp-table takes place, at each site along which the message is sent.

Since the data volume of the lp-table is not large, transmition delay caused by the lp-table is negligible. In both of these algorithms, mention in [20] the operation kick off site sends the contents of its own lp-table together with the operation-request message. Then the update given above is performed at each site, up to the target site, along witch the message has been sent. By this way the, the lp-table of message and the target site become latest which can be received from the operation kick off site, the intermediates sites and the target site.

Finally, using the SVC connection, which is directly establis[illegible] om target site to operation kick off site, the operation result is sent. Simultaneously, by using t[illegible] me SVC connection, the latest lp-table contents are send backward from the target site to every site through which the message has passed. As no connection establishment is required, this process does not take much time. Each site, which receives the latest lp-table contents, updates its own lp-table in the same way. The figure 2.3 shows a message flow in ECF.

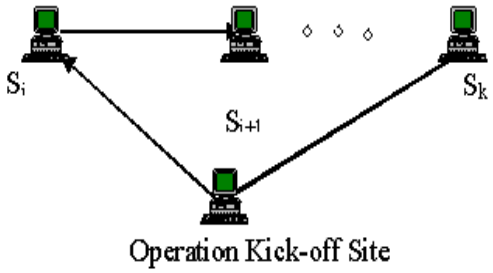

Fig 2.3: Message flow in ECF

PROPOSED TECHNIQUE

In the previous section we have discussed many position-tracking algorithms for migratory data. All these techniques have their pros and cones. In this section we will discuss our proposed technique, which is called as Back-Tracking Chain Forwarding (BCF) technique. The principle of this technique is also same like ECF and ECQ. In this technique also we have a lp-table and it also pass the messages according to the entries in its lp-table. The only difference is that, this technique has one more entry in its lp-table and that is the previous position of the data resource. So, for this technique, we make the following entries in the lp-table.

Present position of data resource
Previous position of data resource
Resource migration count

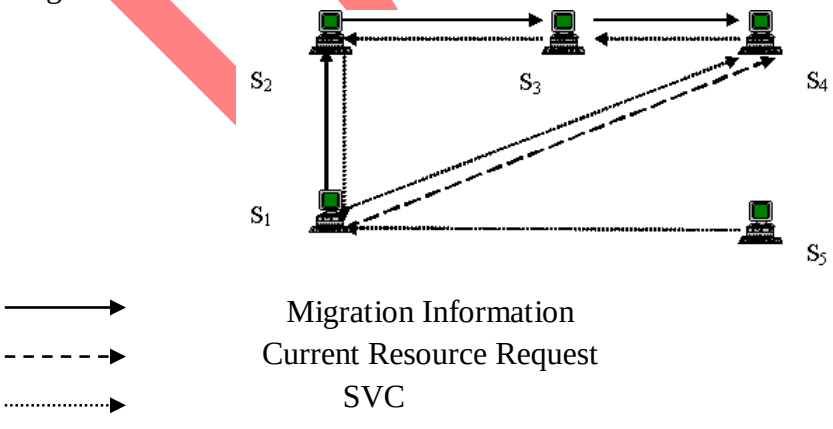

Fig 3.1: Back-Tracking Chain Forwarding

First we will explain this technique with the help of the figure 3.1 which will give the basic idea about this technique and then we will explain it via pseudo code. The simulation results, in section 5, shows that this technique (BCF) takes nearly half the time to ECF, where the number of operation per migration count is more then eight.

Suppose a data resource D_1 is along with site S_1 initially. So site S_1 is the parent site for this data resource and the assumption is that this information is stored with every site in their lp-table. Then if some site needs this data resource D_1, it will send the request to the target site S_1. Suppose after three transaction the data resource is along with site S_4, via the path S_1-> S_2-> S_3-> S_4, and site S_4 is the latest position of the data resource. Now, According to BCF, when site S_3 migrates the data resource to site S_4, it will also send a message to the previous site from which it has taken the data resource and according to the figure 3.1 this site is S_2. Similarly, the site S_2 will send this information to its previous site according to its lp-table. This information is sending in the form of lp-table from which the migration count and newer position field get updated for the previous site. Now suppose, site S_5 require the data resource at this point of time. It will check the entry of its lp-table and will send a request to the site S_1. As site S_1 has the latest information that the data resource is along with site S_4, instead of going through the path S_1-> S_2-> S_3-> S_4, it will directly forward the request of site S_5 to site S_4 and site S_4 will send the required data resource to site S_5. Again, along with sending the data resource and updating its lp-table, site S_4 send the update message to its previous site i.e. to site S_3 according to the figure 3.1. by this way we keep the latest information in the n previous sites.

Pseudo code representation of BCF:

Before giving the pseudo code for BCF, we have explained various terms used to describe this algorithm, the composition of lp-table, various type of messages used in the code and the update rules, which are used to update the lp-tables at various sites during message flow.

Various Terms and Messages Used

A number of terms have been used as notation to represent the pseudo code of BCF. These trams are explained in this section. Two kinds of messages have been used in the implementation of BCF. First kind of messages is control message, which are used to control the flow of the program, and they are PARSAC specific methods. Second kind of messages is to exchange the information between the various sites and can be known as information exchange messages. We will discuss information exchange messages here to explain the idea of BCF.

Terms Used	
curr_id	Latest identified position of the DR
prev_id	Previous site from which current site has taken the DR
resource_mig	Most recent known migration count of the DR
position_table[i][x]	Refers to the position table entry at site i corresponding to DR x.
Backtrace_left	Total number of previous site update in a message

position_table[opr][x]	Position tables carried by op_req (opr) message
position_table[upll][x]	Position tables carried by Update_ll (upll) message
Site_id	ID of operation kick-off site
message.field	Certain message field carried out by various messages
Update_ll.backtrack	Number of sites to be updated in backward direction.

Information Exchange Messages Used:
Four types of information exchange messages have been used here and they are as follows.

1. Operation Request Message (Op_req): Any site that requires some DR but does not hold it, create this message and send it. Every Op_req consists of the information of site_id of the requesting site. Along with this information, it also carries the position table of the operation kick-off site (The site which generate this message or the requesting site). Operation kick-off site send this message according to the entry in its position table. If a site that receives this message does not hold the requested DR, it forwards the Op_req to a site according to its own position table. After successive forwarding, the Op_req reaches the target site. The Op_req updates position tables at all the sites it visits, with the copy of position table it carries, using update rules explaind below in 3.1.3.

2. Update Message (Update_ll): When any site receives a Op_req for the DR it holds, it generates this message (Update_ll). The target site takes out information regarding the site from which it takes the data resource from its own position table. The Update_ll is then sent backwards from the target site to the site from which it takes corresponding DR. It also carries site_id of the target site and a copy of position table maintained at the target site. Along with this, this also contains the latest value of variable backtrack_left, which help it to go to n previous site. It updates the position table at previous site of DR from where it comes to current site. Then this Update_ll keep updating n previous sites, where $n = \log_2 N$ and N is total number of sites in tree network, from where this DR passes or up to the site in which prev_id entry equal to -1, i.e. the parent site of data resource, which ever occur first. Here n is the depth of the binary tree, which is the network model, chose for simulation and it is discussed in 5.2 sub section. This message contains

3. DR Migration Message (msg_ti) : When any site receives a DR migration message for the DR it requested, it update its position table entry. This message is the requested DR along with its resource_mig and prev_id. Sites update the entry corresponding to it, in both the position tables at the target site and the requesting site, according to update rules.

3.1.1 Position Table

In BCF, every site has a position table to store various location information related to every Date Resource (DR). This table has one record corresponding to every DR to store the latest information about that DR. So it has m records for m DRs. Each record have three fields: curr_id, which indicate

the latest identified position of the DR, prev_id, which indicate the id of previous site from which current site has taken the DR, and resource_mig, which indicate the most recent known migration count of the DR. The initial value of prev_id is set to -1 for each DR and it change in lp-table of any site for any particular DR only when that DR comes to this site. Otherwise, it remains unaltered. The value of resource_mig increases by one every time the DR migrates. Therefore, the value i of resource_mig means that the corresponding position information represents the position of the DR after i^{th} migration. Comparing these values, the latest position information about a DR is thus determined.

3.1.2 Update Rules

1. If site i receives a message along with a copy of position table of the site that sends it (i.e. op_req or msg_ti), resource_mig of all DRs in its own position table and that of the position table in the message are compared. If they are not the same, the older DR information, other then the prev_id, is updated to the newer one.
2. If site i sends msg_ti to site j, then the position table entry corresponding to the DR at site i is updated. The current position at site i is updated to site j and the resource_mig value is updated to the value one higher than that already stored at site i and the value of prev_id remain unchanged.
3. If site i receives msg_ti from site j, then current position of the corresponding DR in the position table at site i is updated to site i and the resource_mig value is also updated to the value carried by the msg_ti and the prev_id is updated to site j.

3.1.3 Pseudo Code Description:

This section describes the algorithm by using the various terms and the messages discussed above. Initially, we have to assign the values to some of the variables, which we are using in this description. These are the variables, which start with some value and then change them in the course of execution of the algorithm.

1 Implementation initialization

Initially we assign the values to the various entries of the position table on every site. When we start, each DR is assumed to be located at a different and a unique site. A DR, say DR x, is therefore assumed to be initially located at a site x. Thus, a site can hold maximum of one DR. The value of resource_mig corresponding to all DRs at all sites is also initialized to zero. And the prev_id corresponding to all DRs are initialized to –1. The initialization procedure is as follows:

```
for(i=1;i<=n;i++) /*n sites*/
{
        for(x=1;x<=m;x++) /*m DRs*/
        {
           set position_table[i][x].curr_id=x;
                   set position_table[i][x].resource_mig =0;
```

```
                    /*entries in the position tables are initialized*/
   set position_table[i][x].prev_id=-1;
        }
}
```

We also have to assign the value to the variable backtrace_left, which start from initial value when some update message is generated and then decrement by the time it become 0.

```
        set Update_ll[i][x].backtrace_left=n;
```

/*Where n = $\log_2 N$ and N is total number of sites in tree network */

2 Sequence of Execution:

Each site i in the network is driven by the following events: i generates a Op_req message, i receives an Op_req message, i generates an Update_ll message, i receives an Update_ll message, i receives an msg_ti message and i generates an msg_ti message. It should be noted that each of these actions is processed without interruption. It is also assumed that all operations on a DR are atomic.

2.1 Site i generates a Op_req for DR x

If site i holds DR x (line 1), then operations can be performed at site i, otherwise the request is forwarded to a site given by its position table (lines 2-4).

/*This procedure is executed by site i when it generates a Op_req for DR x*/

```
 if(position_table[i][x].curr_id==i)                    (1)
  /*site i holds DR x*/
 DR x access();
 /*DR x is residing at site i.
 all operations on DR
 x can take place locally.
 no requests need to be generated*/
  else    /*site i does not hold DR x*/                  (2)
 {
 create a Op_req with site_id=i,
 DR_id=x and a copy of position
 table at site i;                                       (3)
  Send Op_req to position_table[i][x].curr_id;          (4)
 /*Forwards the request to the site given by the
  entry corresponding to DR x in its position table*/
 }
```

2.2 Site i receives a Op_req for DR x

Site i updates its position table with the position table carried by the Req_msg according to update rules (lines 5-11). If site i holds DR x, it updates its position table entry for DR x, increments the

migration count of DR x, and migrates DR x to the requesting site (lines 12-15). The target site then sends an update message to update position tables at sites visited by the corresponding request message (lines 16-17). If site i does not hold DR x, it forwards the request to a site given by its position table (lines 18-20).

/*This procedure is executed by site i when it receives a Op_req for DR x */

```
 For each DR x /*update the position table*/                          (5)
 {
 if(position_table[i][x].resource_mig <                              (6)
                 position_table[rm][x].resource_mig)
 {
    set position_table[i][x].curr_id = position_table[opr][x].curr_id;  (7)
            set position_table[i][x]. resource_mig  =                 (8)
                          position_table[opr][x]. resource_mig;
 }
 else                                                                (9)

 {
    set position_table[opr][x].curr_id                               (10)
 = position_table[i][x].curr_id;
 set position_table[opr][x]. resource_mig                            (11)
 = position_table[i][x]. resource_mig;
          }
 }
   if(position_table[i][x].curr_id==i)                               (12)
 /*site i holds DR x*/
          {
 set position_table[i][x].curr_id                                    (13)
 = Op_req.site_id;
 increment position_table[i][x]. resource_mig;
 /*increment the mig_count for DR x*/                                (14)
 send msg_ti (DR x along with its updated
 resource_mig value) to Req_msg.site_id;
 /*migrate DR to the requesting site*/                               (15)
 create an Update_ll with site_id=i,
 a copy of position table at site i and backtrace_left;              (16)
 send an Update_ll to prev_id;
 /*sends the Update_ll to the last site from where
  this site get the    DR x to update its position table*/
                                                                     (17)

 }
```

```
else /*site i does not hold DR x*/                    (18)
{
  create a Op_req with site_id=Op_req.site_id,
  DR_id=x
   and a copy of position table at site i;            (19)
   send Op_req to position_table[i][x].site_id;       (20)
   /*forwards the request to the site given by
   the entry corresponding to DR x in its
   position table*/
}
```

2.3 Site i receives a msg_ti for DR x

Site i now holds DR x. The position table entry corresponding to DR x at site i is updated (lines 21-26).

```
/*This procedure is executed by site i when it receives a msg_ti for DR x*/
 {
 set position_table[i][x].curr_id = i;                 (21)
 set position_table[i][x].prev_id =  msg_ti.site_id;   (22)

 set position_table[i][x].mig_count                    (23)
 = msg_ti. resource_mig;
 DR x access();                                        (25)
 /*updates entries in the position table at site i     (26)
 corresponding to DR x. site i can now start
 operations on DR x*/
 }
```

2.4 Site i receives an Update_ll corresponding to DR x

The Update_ll carries a copy of position table of the site that sends it. It updates the position table at maximum n site it visits or all sites it visits, which ever is lesser, with the copy of position table it carries (lines 27-40). It also carries a variable backtrack, which gives the number of sites it has to visit and the initial value of this variable is n. These are the same sites that the corresponding Req_msg has passed through.

```
/*This procedure is executed by site i when it receives an Update_msg*/
 For each DR x    /*update the position table*/       (27)
 {
 if(position_table[i][x].resource_mig <                (28)
     position_table[upll][x]. resource_mig)
 {
```

```
        set position_table[i][x].curr_id                                    (29)
= position_table[upll][x].curr_id;
set position_table[i][x]. resource_mig                                      (30)
= position_table[upll][x]. resource_mig;
}
else                                                                        (31)
{
   set position_table[upll][x].curr_id = position_table[i][x].curr_id;      (32)
   set position_table[upll][x]. resource_mig =                              (33)
        position_table[i][x]. resource_mig;
}
}
if(Update_ll.backtrack >0 && position_table[i][x].prev_id != -1)
/*to check if position tables at max. n sites or all
the site, which ever is lesser, visited by the
DR x have been updated by the Update_msg*/                                  (34)
{
Update_ll.backtrack = Update_ll.backtrack-1;
 /*Decrement the backtrack count by one at
every previous site */                                                      (35)
create an Update_msg with site_id = I,
a copy of position table at site I;                                         (36)
If (prev_id <> 0)
                                                                            (37)
{
        send an Update_ll to location_table.prev_id;
 }                                                                          (38)
 Else
                                                                            (39)
 {
          exit;
 }                                                                          (40)
}
```

SIMULATION ENVIRONMENT

This section discusses the environment of the simulation and the assumptions used in the simulation. In environment, we will discuss the system model we have chosen.

System Model:

In our simulation we assume the network topology is a complete binary tree as shown in the figure 4.1. We are assuming the depth of the tree as (n-1). So, the total no of sites in the network will be 2^n. Let $S_1, S_2 \ldots S_2^n$ be the sites in the network.

The total communication time in the simulation is the maximum of either the time of the completion of transmission of lp-table or the time the operation request message is sent to the time the result of operation received.

Various notations used for the expression of communication time are as follows:

$A_{i,j}$ Minimum number of ATM switches or hops b/w site S_i & S_j.

$X_{i,j}$ 0 if b/w two site S_i & S_j, an SVC connection exist

1 if b/w two site S_i & S_j, an SVC connection does not exist

L A list of all the sites, from operation kick-off site to target site which must be piloted in the way i.e. $L = \{S_a,\ldots..,S_k\}$.

N Total number of element in L

L_i Subscript of the i-th element.

n Power of 2 in a complete binary tree i.e. $\log_2 x$, where x is the total number of nodes in a complete binary tree.

Now, using equation (1) and (2), the communication time for ECF and BCF is expressed by the following equations (4) and (5).

ECF:

$$T_{ECF} = \sum_{(k=1,\ N-1)} \{ T (A (L_K, L_{K+1})) + X(L_K, L_{K+1}) . C (A (L_K, L_{K+1}))\}$$
$$+ \ Max (T (A_{I,V}) + X_{I,V} . C (A_{I,V}) . \sum_{(k=1,N-1)} T (A (L_K, L_{K+1})) \quad \ldots(4)$$

BCF:

$$T_{BCF} = \sum_{(k=1,\ N-1)} \{ T (A (L_K, L_{K+1})) + X(L_K, L_{K+1}) . C (A (L_K, L_{K+1}))\}$$
$$+ \ Max (T (A_{I,V}) + X_{I,V} . C (A_{I,V}) . \sum_{(k=1,n)} T (A (L_K, L_{K+1})) \quad \ldots(5)$$

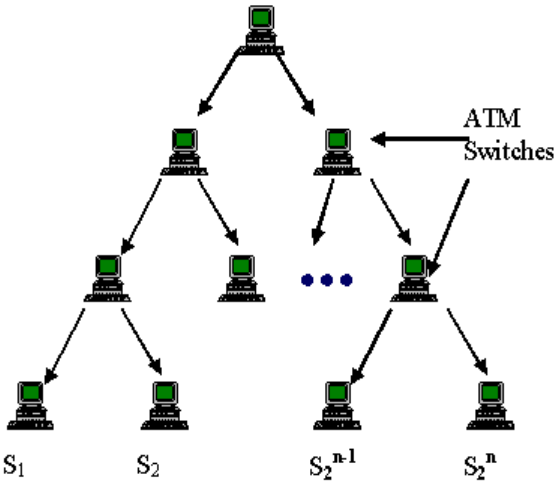

Fig 4.1: Network topology for the simulation environment

Simulation Details:

The basic parameters used in the simulations are as follows. The numbers of data resources considered are 20 (D_1 to D_{20}), and we assume that initially each data resource D_i is located at site S_i. The parameters about data transmission are totally based on the consideration of ATM technology. For simplicity, each site initiates the operation in equal probability, and the objective resource chosen is also chosen with equal probability. The intervals of access and resource migration are based on exponential distributions, and an SVC connection is released when it is not used for more then 20 minutes.

Also for our tree structure network topology, we are taking n = 5 i.e. the number of site will be 32. In this simulation, 5000 operations are performed by both the (ECF and BCF) techniques. Mean access interval is changed by 30 seconds from 30 seconds to 3 minutes and the access/migration i.e. access to resource migration ratio is changed by 1 from 3 to 15.

RESULT AND DISCUSSION

The techniques, which are not ATM specific, are evaluated in terms of traffic load. But, in ATM network, the size of transmitted data is not as important as response time. Therefore, in our simulation experiment, our main concern is the communication time required for accessing data resource.

Below, in fig 5.1 and fig 5.3, which are 2-D representation of ECF and BCF respectively, x-axis indicates the access per migration ratio and y-axis indicates average time delay per request. These are plotted for different values of mean access interval for different Access/Migration ratio and it shows the variance in average time delay per request. While, in fig 5.2 and 5.4, which are 3-D in nature for ECF and for BCF respectively, x-axis indicates access/migration ratio, y-axis indicates the mean access interval and z-axis indicates the average time delay per request. The figures show that access/migration ratio affects both ECF and BCF. The average time delay decrease as

access/migration ratio increase. On average, the average time delay changes from nearly 500ms to 400ms in case of ECF, which is from 500ms to 100ms in case of BCF.

Now, from the above simulation results, we discuss the optimal method (i.e. the method which give shorter average time delay) in each access interval and access/migration ratio. Figure 5.1 and 5.3 shows that for the access/migration ratio 3 to 15, average time delay varies from 450ms to nearly 380ms for access interval equal to 30 sec. in case of ECF. And for the same set of condition, average time delay varies from 425ms to 75ms in case of BCF. So in comparison to ECF, BCF gives much batter performance in case of lowest access interval. Also, in case of highest access interval i.e. for access interval 180 sec., the average time delay varies from 550ms to nearly 400ms in case of ECF which varies from 510ms to nearly 120ms in case of BCF for access/migration ratio 3 to 15.

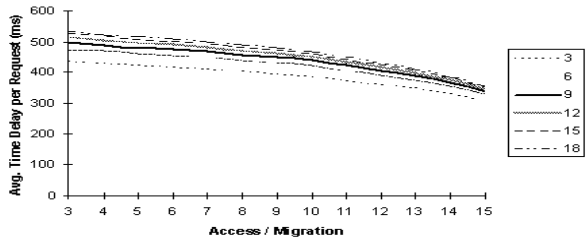

Fig: 5.1: Time Delay using ECF Algorithm for Binary Tree Topology (for n=32 and m=20)

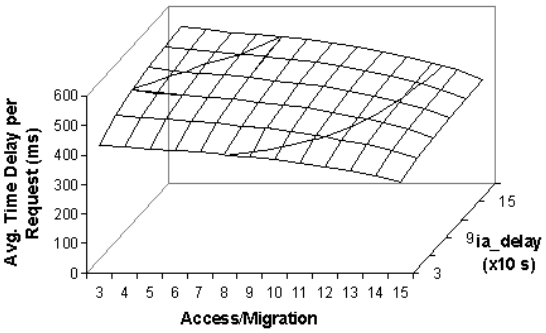

Fig 5.2: Time Delay using ECF Algorithm for Binary Tree Topology (for n=32 and m=20)

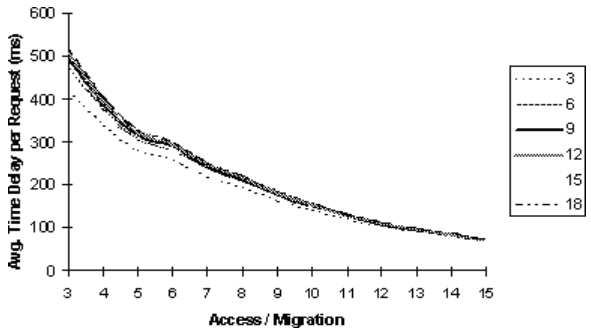

Fig. 5.3: Time Delay using BCF Algorithm for Binary Tree Topology (for n=32 and m=20)

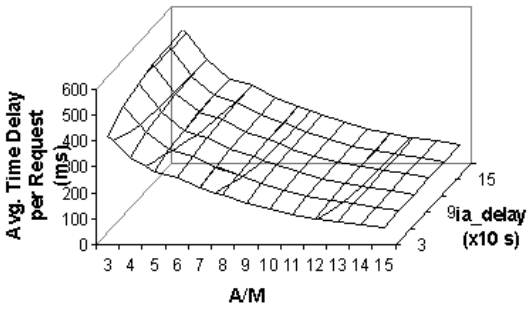

Fig 5.4 Time Delay using BCF Algorithm for Binary Tree Topology (for n=32 and m=20)

CONCLUSIONS

In this paper we have simulate two technique i.e. ECF and BCF, which use various features of ATM. By simulation results, we have compare the performance of these two methods. The simulation results show that the optimal technique can be identified according to the frequency of access and resource migration. If we compare the graph number 5.1 and 5.3 we find that for access/migration ratio equal to 9 and access interval equal to 90 sec., we have communication time nearly equal to 480 ms for ECF while it is merely 180ms in case of BCF for the same set of conditions, which is nearly 1/3 of ECF. This means that the decrease in the amount of communication time from ECF to BCF, as the Access/Migration count increase, is awesome. This comparison shows the dominancy of BCF over ECF in ATM environment.

In our simulation experiment, we have taken the basic situation, i.e., each site initiates the operation in equal probability, and the objective resource chosen is also chosen with equal probability. But for practical use of these methods, it may be necessary to study the simulation result in more complicated situation, i.e., environment where each site initiates the operation in different probability, and the objective resource chosen is also chosen with different probability.

REFERENCES

1. Nishio, S. and Shimojo, S., "Status Update of Databas Systems through Multimedia Computer Networks," IEICE Trans. Commun., Vol. 78, No.7,pp. 946-951 (July 1995).
2. Cha H., Kim J, and R. Ha, "Bandwidth constrained smoothing for multimedia streaming with scheduling support," Journal of Systems Architecture. Vol. 48, Issue 11-12, pp. 353-366 (April 2003).
3. C. Bewick, R. Pereira, M. Merabti, Network Constrained Smoothing: Enhanced Multiplexing of MPEG-4 Video, Proceedings of the Seventh International Symposium on Computers and Communications (ISCC'02), p.114, July 01-04, 2002
4. Devine, R., "Design and Implementation of DDH. A Distributed Dynamic Hashing Algorithm." Proc. of 4th Int'l Conf. On foundation of Data Organization and algorithms (FODO). Chicago (1993).
5. Johnson, T. and Krishna, P. "Lazy updates for Distributed Search Structure," Proc. Of ACM SIGMOD '93, pp337-346 (1993).
6. Matsliach, G. and Shmueli, O.," An Efficient Technique for Distributing Search Structures," Proc. of 1st Int'l Conf. On Parallel and Distributed Information Systems (PDIS) 1991.
7. Litwin, W., Neimat M.A., and Schneider, D.A.,"LH* - Linear Hashing for Distributed Files," Proc. Of ACM SIGMOD '93, pp327-336 (May 1993).
8. Vingralek, R., Breitbart, Y., and Weikum, G., "Distributed File Organization with Scalable Cost /Performance," Proc. Of ACM SIGMOD '94.
9. Severance, C., Pramanik, S., and Wolberg, P., "Distributed Linier Hashing and Parallel Projection in Main Memory Databases," Proc. of 16th Int'l Conf. On Very Large Databases (VLDB '95), pp674-682 (1990).
10. Hara, T., Harumoto, K., Tsukamoto, M., and Nishio, S., "Database Migration for Transaction Processing in ATM Networks," Proc. of Int'l Conf. on Information Networking (ICOIN-11).
11. Hara, T., Harumoto, K., Tsukamoto, M., and Nishio, S., "Location Management for Database Migration in ATM Networks," Trans. IEICE D-I.
12. Nishio, S. and Tsukamato, S., "Toward New Multimedia Information Base in Broadband Networks," Trans. IEICE D-II Vol 79-D-II, No.4,pp460-467 (April 1996).

13. Xie J. and Akyildiz A., "A Noval Distributed Dynamic Location Management Scheme for Minimizing Signaling Cost in Mobile IP," IEEE Transaction on Mobile Computing, vol. 1, no. 1, pp. 163-175, July-Sept. 2002.
14. Levy H. and Naor Z., "Locating mobile users in personal communication service networks," Wireless Networks, pp. 467-477, 1999.
15. Talukdar A.K., Badrinath B.R and Acharya A., "A Resource Reservation Protocol for an Integrated Service Network with Mobile Hosts," Wireless Networks, pp. 5-19, 2001.
16. W. Ma and Y. Fang, "Two-level pointer forwarding strategy for location management inPCS networks," IEEE Transaction on Mobile Computing, vol. 1, no. 1, pp. 32-45, Jan.-Mar. 2002.
17. Li J., Pan Y. and Xiao Y., "A Dynamic HLR Location Management Scheme for PCS Networks," IEEE INFOCOM 2004.
18. J. Ho and I, Akyildiz, "Dynamic hierarchical database architecture for location management in PCS networks," IEEE/ACM Trans. Networking, vol. 5, no. 5, pp. 646-660, Oct. 1997.
19. J. Li, H. Kameda and K. Li, "Optimal dynamic mobility management for PCS networks," IEEE/ACM Trans. Networking, vol. 8, no. 3, pp. 319-327, 2000.
20. Takahiro, H., Kaname, H., Masahiko, T., and Shojiro, N., "Location Management Techniques of Migratory Data Resource in ATM Networks," ACM publication.
21. Banerjee, s., Li, V.O.K., and Wang, c., "Distributed Database Systems in High Speed Wide Area Network," IEEE Journal on Selected Areas in Commun., Vol.11, No. 4, pp617-630 (May 1993).