Journal of Information Systems and Communication
ISSN: 0976-8742 & E-ISSN: 0976-8750, Volume 3, Issue 1, 2012, pp.-365-370.
Available online at http://www.bioinfo.in/contents.php?id=45

TRACKING MIGRATORY DATA IN DISTRIBUTED ENVIRONMENTS

TWISHA DHINGRA[1], VIKAS DAHIYA[2] AND VIKAS SINGHAL[3]

[1]Department of Computer Science & Engineering, GTBIT, New Delhi.
[2]SSA, Bally Technologies.
[3]Member Consulting Staff, Mentor Graphics, Noida.
*Corresponding Author: Email- twisha.dhingra@gmail.com

Received: January 12, 2012; Accepted: February 15, 2012

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.
Keywords- ATM (Asynchronous Transfer Mode), Tracking, Resource Migration

Citation: Twisha Dhingra, Vikas Dahiya and Vikas Singhal (2012) Tracking Migratory Data in Distributed Environments. Journal of Information Systems and Communication, ISSN: 0976-8742 & E-ISSN: 0976-8750, Volume 3, Issue 1, pp.-365-370.

Introduction

The bandwidth requirements for data traffic within commercial organizations have been increasing steadily for some time, both in the LANs and in WANs. 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 internetworks 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.

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. ATM will become the network topology of choice.

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.The problem with the available mobile computing techniques is they require many number of message to track the position of the data resource. The techniques given in [12] 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 propose a new technique, which utilize the features of ATM network. After that, we will compare this technique with existing similar kind of technique.

Techniques For Resource Tracking

Network model

In ATM 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 communication time for the connection which is already established is: $T(n) = nr + 2p + (n+1)$ (1) Where,

T(n)= Total Communication Time

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

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

Where,

C(n) = Total connection established time

p' = Total processing delay at one site

R = Constant time for route configuration at each ATM switch

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

$$T'(n) = C(n) + T(n) \qquad (3)$$

…(3)

where, T'(n) = Total message transmission time

Existing techniques for tracking of migratory data

There are many proposed resource tracking techniques for mobile network but only those were consider in [12] 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:

a. Operation kick off site: It issues an operation-request message.
b. Objective resource: The data resource to be accessed.
c. Target site: It holds one of the Objective resources.

The ECF technique

When a site need some data resource suppose D_I, it sends an operation request message according to the entry in its lp-table against data resource D_I. Then, if the site that receives this request does not hold the data resource D_I, 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_I, and the result together with current position of D_I 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.

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_I. Then if the site, which receives the message, does not hold the entry for data resource D_I, it sends information of D_I in its own lp-table to the operation kick off site. By this way, the operation kick off site update the entry for D_I in its own lp-table based on the reply message, and sends the operation-request message again to the new position of D_I. 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 [12] 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 established, from target site to operation kick off site, the operation result is sent. Simultaneously, by using the same 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.

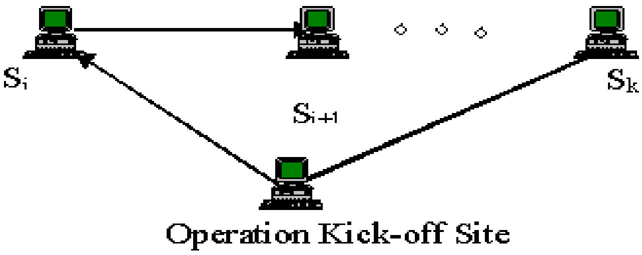

Fig. 2.3- Message flow in ECF

Proposed Technique

In 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.

a. Present position of data resource
b. Previous position of data resource
c. Resource migration count

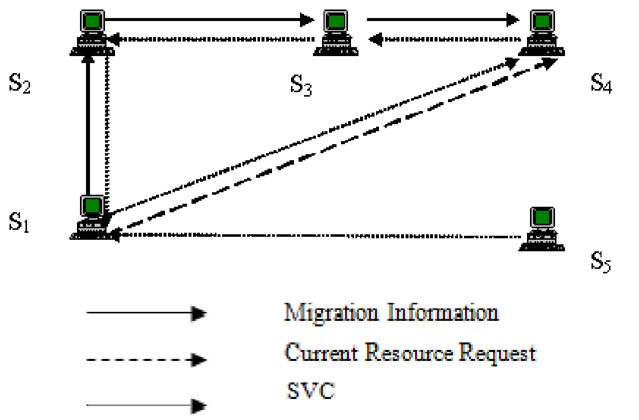

Fig. 3.1- Back-Tracking Chain Forwarding

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

Various Terms and Messages Used

Two kinds of messages have been used in the implementation of BCF. First is *control message*, which are used to control the flow of the program. Second is used to exchange the information between the various sites and can be known as information exchange messages.

Terms Used	
curr_id	Latest identified position of the DR
prev_id	Previous site from which current site has taken the DR
r_mig	Most recent known migration count of the DR
p_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
p_table[opr][x]	Position tables carried by op_req (opr) message
p_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

Three 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 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 r_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.

Position Table

In BCF, every site has a position table(p_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 ad=nd its initial value is -1, and r_mig, which indicate the most recent known migration count of the DR. Comparing these values, the latest position information about a DR is thus determined.

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), r_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 r_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. 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 r_mig value is also updated to the value carried by the msg_ti & prev_id is updated to site j.

Pseudo Code Description

Implementation initialization

When we start, each DR is assumed to be located at a different and a unique site. The initialization procedure is as follows:

```
for(i=1;i<=n;i++)  /*n sites*/
{        for(x=1;x<=m;x++)  /*m DRs*/
         {   set p_table[i][x].curr_id=x;
                 set p_table[i][x].r_mig =0;
                 set p_table[i][x].prev_id=-1;  } }
```

We also have to assign the value 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 */

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.

```
if(p_table[i][x].curr_id==i)                                   (1)
DR x access();
 else    { create a Op_req with site_id=i,                     (2)
DR_id=x and a copy of position  table at site i;              (3)
Send Op_req to p_table[i][x].curr_id;}                        (4)
```

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).

For each DR x

```
{ if(p_table[i][x].r_mig  <   p_table[rm][x].r_mig)                 (5)
{   set p_table[i][x].curr_id = p_table[opr][x].curr_id;            (6)
          set p_table[i][x]. r_mig  =  p_table[opr][x]. r_mig;}     (7)
else    {set p_table[opr][x].curr_id = p_table[i][x].curr_id;       (8)
set p_table[opr][x]. r_mig = p_table[i][x]. r_mig; }   }            (9)
 if(p_table[i][x].curr_id==i)  /*site i holds DR x*/                (11)
{  set p_table[i][x].curr_id = Op_req.site_id;
increment p_table[i][x]. r_mig;                                     (12)
send msg_ti (DR x along with its updated
r_mig value) to Req_msg.site_id;
create an Update_ll with site_id=i,
a copy of position table at site i and backtrace_left;              (13)
send an Update_ll to prev_id;  }                                    (14)
else {   create a Op_req with site_id=Op_req.site_id,               (16)
  DR_id=x and a copy of position table at site i;                   (17)
   send Op_req to p_table[i][x].site_id;  }                         (18)
```

Simulation Environment

System Model

In our simulation we assume the network topology is a complete binary tree. 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 no. of ATM switches 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: list of all the sites, from operation kick-off site to target site which must be piloted in the way,L={S_a,.,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 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: $T_{ECF} = \sum_{(k=1,N-1)} \{ T(A(L_K, L_{K+1})) + X(L_K, L_{K+1}) . C(A(L_K, L_{K+1}))\} + Max(T(A_{I,V}) + X_{I,V} . C(A_{I,V})). \sum_{(k=1,N-1)} T(A(L_K, L_{K+1}))$ (4)

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

Simulation Details:
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. 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. In our simulation experiment, our main concern is the communication time required for accessing data resource.
Now, from the above simulation results, we discuss the optimal method 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.

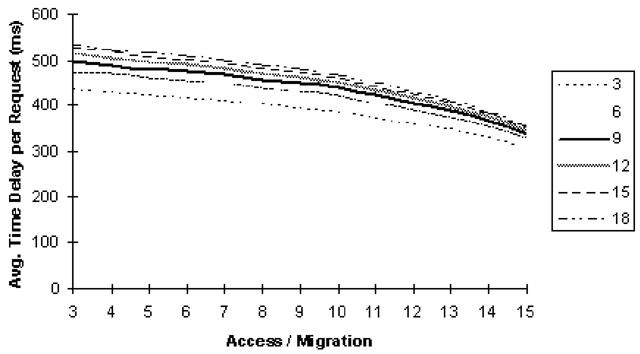

Fig: 5.1- Time Delay using ECF Algorithm for Binary Tree Topology (for n=32 and m=20)

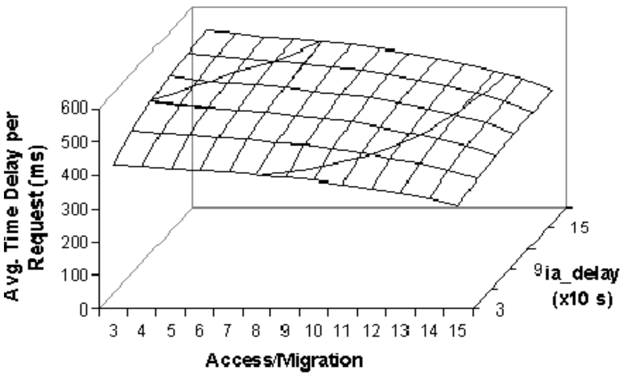

Fig 5.2- Time Delay using ECF Algorithm for Binary Tree Topology (for n=32 and m=20)

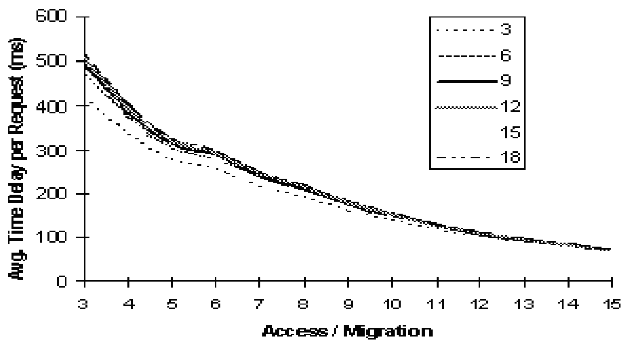

Fig. 5.3- Time Delay using BCF Algorithm for Binary Tree Topology (for n=32 and m=20)

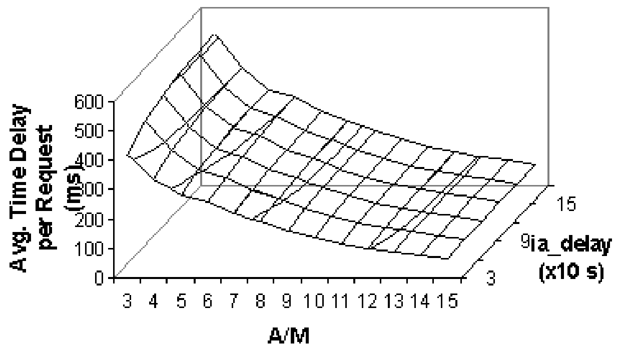

Fig. 5.4- Time Delay using BCF Alogrithm for Binary Tree Topology (for n=32 and m=20)

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.We conclude that 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. (1995) *IEICE Trans. Commun.,* 78 (7), 946-951.
[2] Cha H., Kim J. and Ha R. *Journal of Systems Architecture*. 48 (11-12), 353-366.
[3] Bewick C., Pereira R., Merabti M. (2002) *International Symposium on Computers and Communications* (ISCC), 114.
[4] Matsliach G. and Shmueli O. (1991) 1st *Int'l Conf.* On *Parallel and Distributed Information Systems*.
[5] Litwin W., Neimat M.A. and Schneider D.A. (1993) *ACM SIGMOD,* 327-336.
[6] Hara T., Harumoto K., Tsukamoto M. and Nishio S. (2011) *Int'l Conf. on Information Networking* (ICOIN).
[7] Hara T., Harumoto K., Tsukamoto M. and Nishio S., *Trans. IEICE D-I*.
[8] Xie J. and Akyildiz A. (2002) *IEEE Transaction on Mobile Computing*, 1(1), 163-175.
[9] Levy H. and Naor Z., *Wireless Networks*, 467.
[10] Ma W. and Fang Y. (2002) *IEEE Transaction on Mobile Computing*, 1(1), 32-45.
[11] Li J., Pan Y. and Xiao Y. (2004) *IEEE INFOCOM.*
[12] Takahiro H., Kaname H., Masahiko T., and Shojiro N. *Location Management Techniques of Migratory Data Resource in ATM Networks*.