A Cost Model For The Placement Replications In Data Grid

Ghalem Belalem[1] and Farouk Bouharaoua[2]
[1]Department of Computer Science, Faculty of Science, University of Es Senia Oran,
BP 1524, El M' Naouer, Oran, Algeria
Ghalem1dz@Yahoo.fr
[2]Department of Computer Science, Faculty of Science, University of Mostaganem, Algeria
Farouk622000@yahoo.fr

ABSTRACT

Nowadays, the grid environment presents new challenges, such as the dynamic availability of various resources that are geographically distributed, the quick and the efficient access to data, reduction of time of latency and fault tolerance. These grids are concentrated on the reduction of the execution time of the applications that require a great number of processing cycles by the computer. In such environment, these advantages aren't possible unless by the use of the replication. This later is considered as an important technique to reduce the cost of access to the data in grid. In this present paper, we present our contribution to a cost model whose objective is to reduce the cost of access to replicated data. These costs depend on many factors like the bandwidth, data size, network latency and the number of the read/ write operations.

Key words: *Data Grid, replication, data placement, cost model, CERN.*

1. INTRODUCTION

Today, the utility of many internet services is limited by the availability rather than the execution. The replication is the most used approach to offer the highest availability of data. The experiment on the distributed systems show that the reproduction promotes high data availability, low bandwidth consumption, increased fault tolerance and improved scalability [1]. The replication is the process of creation and placement of the copies of entities software. The phase of creation consists in reproducing the structure and the state of the replicated entities, whereas the phase of placement consists in choosing the suitable slot of this new duplication, according to the objectives of the replication.

The replication cost is often linked to the deployment cost of the reproduction to the dynamic creation cost of the reproduction and to the emplacement replication costs towards the client.

In our approach, the placement algorithm of the replicas was designed by the cost model, formulated as an optimization problem that reduces at least the global cost of access to data, further to a focusing on the influence of the read/ write operations on the replication cost (report: number of readings, number of writings) in grids at a given moment. What will facilitate us to take a decision for the creation or the moving of replicas to adequate sites as well as their deleting.

In order to evaluate this cost model, a simulator called "GREP- SIM" was developed and implemented under a tree hierarchical topology inspired from "CERN" data grid architecture (the European Organization for Nuclear Research) [3]. The results show that the performances of the replications depend strongly on the replicas emplacement, and the number of the read/write operations effected on these replicas.

The remainder of the paper is organized as follows. In the second section we presents an overview of some costs models for the data replications. The third section will be reserved for our contribution in a cost model for replication in data grids. Experimentation results of our model on tree topology are the object of the fourth section. Finally, section five will be reserved for the conclusion and some future works.

2. RELATED WORKS

The primary reason to use a cost model is the ability to take optimal decisions for accessing to data replicas in the future use. The execution of such multidimensional and complex optimizations in a centralised way is very difficult, since the planning domain (attributes of resources) is very huge. The optimisation service of grid must be scalable in items of the number of network nodes (tens, hundreds, or even thousands) [4]. The grid is a dynamic environment where the resources status can change without any warning. By employing a model, we can exploit this dynamism to take decisions during the execution time of the work. Recently, there has been a noteworthy interest for the proposition of cost models in the grids environment.

The presented works in [5,6], in order to insure the efficient and rapid access to enormous and largely distributed data, the authors suggest a set of services and protocols of replication management, which offer higher data availability, a low bandwidth consumption and a great faults tolerance. Replication decisions have been taken according to cost model, which evaluate the

data access cost and execution gain. This model is function of many parameters just like the response time, the bandwidth, and the reproduction costs accumulated by execution time. To guarantee the scalability, the duplications are organized combining of hierarchical and flat topologies. This suggested model based on Fat-Tree topology [7], where the bandwidth increase leaves towards the root. Very interesting results were produced by the simulator NS [8].
The cost model proposed in [9] is well adapted to machines MIMD. However, it can't be applied to machines of SIMD type where the performances analysis in communication is more complex, because of the difficulty of separating the calculus. This model is according to routing time of a message between processors, initiation time (start up) and bandwidth. This work has shown that routing time of a message and the initiation time have an important influence on the choice of processors interconnection topology, data organisation, load balancing, the number of acceptable synchronization and the messages size.

For the suggested model CASEY in [10] which concentrate on data placement problem. It depends on the sites number, the read number effected by a site by a time unity called: read volume by site, the cost of a read by a given site on the copy placed in another site, the write number effected by a given site by a time unity, write cost realized by a given site of the copy placed on another site. Finally, the storage cost, by time unity, of a copy placed on a given site. The optimal placement returns back to minimize a global objective function made up of parameters group described in above passage. This model is not well adapted on a large-scale environment, because it belongs to the NP-complete problems class. Extra hypothesis can simplify it neglecting storage and write costs.

3. A COST MODEL AND REPLICAS PLACEMENT

A possible data distribution of important sizes consists in replicating them so as to place them on a set of servers. The interest of this fragmentation and this distribution is to dispatch up the load on different servers, and to avoid the bottlenecks appearance. Many placement strategies can be used in these types of systems [11,12,13,14,15]. In this paper, we have used a distribution on a tree network topology.

THE GRID TOPOLOGY

The data grid topology used as a support for the proposed model is described below in figure 1, and inspired of the architecture CERN. It is made up of 5 levels: the root (level 0), 3 sublevels that contain nodes (level 1 till 3), and the last level is for leaves (level 4). All the nodes including the root, representing the servers, can have data replicas, except the last level of the leaves, which represents the clients from where the requests come.

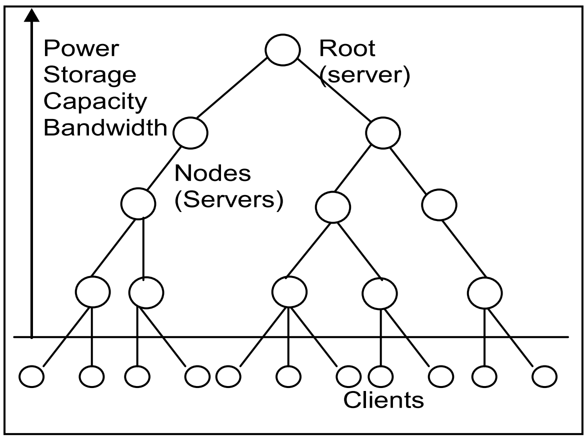

Figure 1 : Logical topology of the grid (CERN)

In this topology, every node admits one child or many except the leaves, and every node has only one immediate father except the root.

A COST MODEL

The replication cost model suggested in our approach is designed as an optimization problem of replicas placement, which reduces at least the sum of the access cost in a grid, basing on the described parameters in table 1.

pn	The immediate father of a node *n*
R_d	The group of nodes containing a replica of data *d*
BW(n)	Bandwidth between the nodes *n* and *pn*. (BW(n) =1/ bandwidth(n, pn))
Size(d)	Size of data *d*
$Way_d(n1,n2)$	A set of nodes met along the way of node *n1* till node *n2*, except node *n2*.
TE_{dn}	Processing cost of write operation on data *d* situated at the node *n*.
TL_{dn}	Processing cost of read operation on data *d* situated at the node *n*.
NE_{dn}	Write number affected on the data *d* passed through the node *n*.
NL_{dn}	Read number affected on the data *d* passed through by the node *n*.

Table 1: The parameters used in the model

Let $n \in R_d$, the nearest node serving a set of clients that query (read and write) data *d*. We are going to calculate the global access cost of data *d* situated in the node *n*, which is the sum of: transfer cost of data *d*, processing calculus cost of operation (write or read) by the node *n*, and the propagation cost of data *d* updating towards the other nodes belonging to R_d.

CT_d(n1, n2): transfer cost of data *d* from node *n1* to node *n2*.

$$CT_d(n_1,n_2) = Size(d) * \sum_i BW(i); i \in Way_d(n_1,n_2) \quad (1)$$

To maintain the replicated data coherence, we use strict protocol ROWA[2], from where
CU_{dn} : propagation cost of updating data *d* situated at node *n*.

$$CU_{dn} = \sum_{k} CT_d(n,k); k \in R_d - \{n\} \tag{2}$$

When a client *c* interrogates data *d* located in the closest node *n*, the costs are evaluated to:
CE_{dn}: write cost on data *d* situated at node *n*

$$CE_{dn} = CT_d(c,n) + TE_{dn} + CU_{dn} \tag{3}$$

CL_{dn}: read cost on data d situated at node n

$$CL_{dn} = CT_d(c,n) + TL_{dn} \tag{4}$$

Let's admit, in what follows, that all the nodes of the same level have the same characteristics (power and high storage capacity), and the same bandwidth between a node and its sons.
Therefore, we can deduce that the write cost of the node *n* (CE_{dn}) is fixed for the interrogations of descending clients (leaves) of this node *n*. The same for the read cost (CL_{dn}). Consequently, we can evaluate the global access cost:

$CostG_{dn}$: global access cost to data d situated at node n

$$CostG_{dn} = NL_{dn} * CL_{dn} + NE_{dn} * CE_{dn} \tag{5}$$

The optimization problem of the global cost function will be to find the minimum cost of N servers containing asked data *d*, we will have:

$$Cost\_Min = \underset{i \in N}{Min}\{NL_{di} * CL_{di} + NE_{di} * CE_{di}\} \tag{6}$$

PLACEMENT ALGORITHM

A node can serve several clients that formulate requests on data, when this last is the nearest node to them containing a reply of this data.
Going by the described cost model above, placement algorithm takes the decision of moving, creating or even deleting the replicas.
This decision is taken according to the number of writings and readings effected on the data replica *d* situated in node *n* by the clients in a given time.
It has to be noticed that in case of only reads are occurred by clients ($NE_{di} = 0 \ \forall \ i \in R_d$), it is evident that The best solution is the one where all the replicas will be placed on all the nodes of the before last level (level 3).
Yet, for the case where exist writes only ($NL_{di}=0 \ \forall \ i \in R_d$), the best solution is to have no replica. This is because of updating propagations.

Let Asc(n) a set of ascendants (ancestors) of node *n* and Des(n) a set of descendants of node *n*.
Let's suppose at the given moment, that the global cost $CostG_{dn}$ is the smallest cost among the ones of Asc(n)+Des(n). The increase of the number of writings NE_{dn} by a certain value (to be identified) modifies the least cost which will be equal to another cost of node $an \in$ Asc(n) ($Cost\_Min=CostG_{dan}$), while the increase of the number of reading by a certain value (to be identified) modifies the least cost value which will be equal to another cost of node $dn \in$ Des(n) ($Cost\_Min=CostG_{ddn}$). Therefore, the aim is to find the interval of NE_{dn} values and the interval of the NL_{dn} values so that the cost $CostG_{dn}$ stays always the lowest ($Cost\_Min=CostG_{dn}$), in other words, the replica of data *d* will always be situated at the node *n*. This leads us to find a trade-off between the number of writings NE_{dn} and the number of readings NL_{dn}.

Let T_{dn} the ratio of the read number on the write number occurred on data *d* situated in node *n*.

$$T_{dn} = NL_{dn} / NE_{dn}$$

Let Son(n) a set of nodes, which are at the same moment the immediate sons of node *n* and ancestors of clients that interrogates data *d*.
So that $CostG_{dn}$ the cost of node *n* be the smallest cost of all nodes belonging to Asc(n)+Des(n)+n.

($CostG_{dn} = \min\{NL_{di} * CL_{di} + NE_{di} * CE_{di}\}$
$\forall$ i $\in$ Asc(n)+Des(n)+n)

T_{dn} must check the following equations:

$$\left.\begin{array}{l} \forall sn \in \text{Son}(n) \\ \textbf{Si } n \text{ est racine } \textbf{Alors } T_{dn} \prec \dfrac{CE_{dsn} - CE_{dn}}{CL_{dn} - CL_{dsn}} \\ \textbf{Si } n \in \text{avant dernier niveau } \textbf{Alors } T_{dn} \geq \dfrac{CE_{dn} - CE_{dpn}}{CL_{dpn} - CL_{dn}} \\ \textbf{Sinon } \dfrac{CE_{dn} - CE_{dpn}}{CL_{dpn} - CL_{dn}} \leq T_{dn} \prec \dfrac{CE_{dsn} - CE_{dn}}{CL_{dn} - CL_{dsn}} \end{array}\right\} \tag{7}$$

Let a replicas distribution on the grid in a given moment, and let the cost of the node *n* the smallest cost of all nodes belonging to Asc(n)+Des(n)+n ($Cost\_Min=CostG_{dn}$). Placement algorithm applied on the node *n* whose its T_{dn} value has changed will be as follows:

Algorithm :

```
x, z : node
x:=n                                    /* initialization */
If  $T_{dx}$'s value don't check equation  (7)  Then
  If  $T_{dx} < \frac{CE_{dx} - CE_{dpx}}{CL_{dpx} - CL_{dx}}$  Then
    While $T_{dx} < \frac{CE_{dx} - CE_{dpx}}{CL_{dpx} - CL_{dx}}$  Do
      $NE_{dx} := NE_{dx} + NE_{dpx}$          /* Updating
      $NL_{dx} := NL_{dx} + NL_{dpx}$               $T_{dx}$   */
      x :=px
    EndDo
    Move the replica of node n towards the node x
    For any z $\in$ Des(x) And z is server of data d  Do
      Calculate $CostG_{dx}$                /*Old*/
      $NE_{dx} := NE_{dx} + NE_{dz}$
      $NL_{dx} := NL_{dx} + NL_{dz}$
      Calculate $CostG_{dx}$               /* New*/
      If New $CostG_{dx} \leq$ Old $CostG_{dx}+CostG_{dz}$  Then
          Delete the replica of data d  from node z
      EndIf
    EndDo
  Else
    For any z $\in$ Son(x) Do
      Create a replica of data d on node z
    EndDo
    Delete the replica of data d  from node x
  EndIf
EndIf
End Algorithm.
```

After the arrival of the requests, we check the T_{di} values $\forall$ data *d* and $\forall$ $i \in R_d$. Thus, according to T_{di}'s values, we take decisions of the emplacement of concerned replicas.
We notice that the suppression of replicas, which the requests number produced by the clients for these replicas is inferior at a given threshold, gives rise to a clear and remarkable amelioration in the global access cost to the grid data.

4. SIMULATION

The presented cost model seems to be very well adapted to hierarchical architectures.
A simulator called "GREP-SIM" was developed in order to evaluate this model. This simulator allows us to generate a hierarchical topology of a tree inspired of data grid architecture "CERN".
In this simulator, every node in the network is able to specify its storage capacity, its processor performance, the local replicas, and a detailed history of the requests passing through this node (the passage of a request by this node at the time of its routing from a client toward its server node), where we can associate two variables for every data in the grid specifying respectively the read and write number occurred on this data and that are passed through this node.
This detailed history of nodes will indicate us the number of the produced requests by every client on every data. If the number of produced requests on data exceeds a certain threshold, the concerned client will be considered as the best client for this data.
Many algorithms were proposed for the emplacement of replicas in the closest server to the best client, by using several solutions to select the most appropriate server to contain its replicas [16,17,18].
In order to compare our model, we shall use two other placement models. The first one is based on the best client algorithm where the replica will be placed on the closest node of this client [19]. The second one is based on the common father algorithm. Contrary to the first model, which favours the best client and penalizes another client, which have produced a number of requests exceeding the threshold, the common father place the replica on the closest common node of the two clients.

4.1 SIMULATED GRID MODEL

The topology of simulated grid, described in the following schema in figure 2, is compound of 15 nodes. A root, 14 intermediary nodes and 16 leaves representing the clients. The bandwidth improves from inferior level to the superior one.

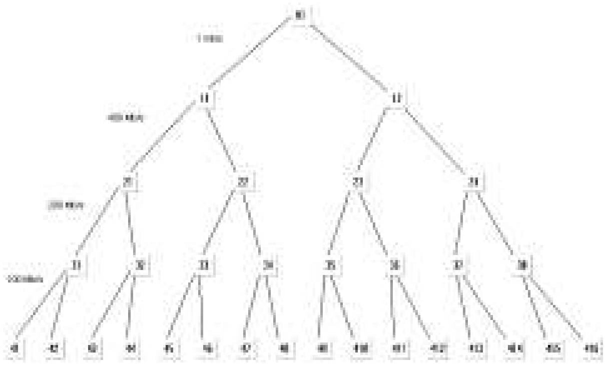

Figure 2: Simulated model of the grid topology

4.2 EXPERIMENTATIONS AND RESULTS

The experimentations were held on definite simulation parameters in the table 2.

Data number	4
Data size	From 1 Gb to 4 Gb
Replicas number	From 1to 8 per data
Requests number	800
Threshold	50

Table 2: Simulation parameters

In the beginning, a random distribution of replicas has been done and saved along the simulation.
The simulator calculates the response times of every request for this distribution. Then, it recalculates the responses time for the same requests but by applying models of the best client, common father and finally our model, which we have named it, the best ancestor.
For the same distribution, we execute five different experimentations (scenarios), where the read and write

numbers produced by the clients are modified, as it is mentioned in the following table 3:

Scénario	Read number	Write number
1	800	0
2	770	30
3	740	60
4	600	200
5	450	350

Table 3 : Read/Write number per scenario

The obtained results are illustrated by two graphs.
The figure 3 represents the average time response per scenario and the figure 4 represents the average time response per clients of the third scenario.

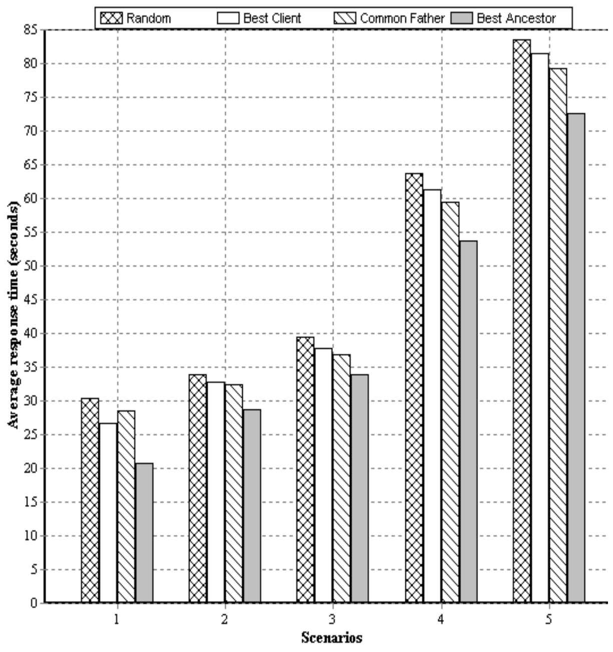

Figure 3: Average time response per scenario

The results show that the global access cost in the grid is clearly improved when using our model. Indeed, we notice in the first graph a decrease of 11, 41% of the average time response of this model compared to the common father model, of 12.72% compared to the model of the best client and of 16, 51% compared with the random distribution.

We were anxious to present the second graph (Figure 4) to show some exceptional cases that can appear. For instance, the averages time response for the clients 413 and 416 that are favoured by the model of the best client are the best times, versus the clients 414 and 415 that are penalized.

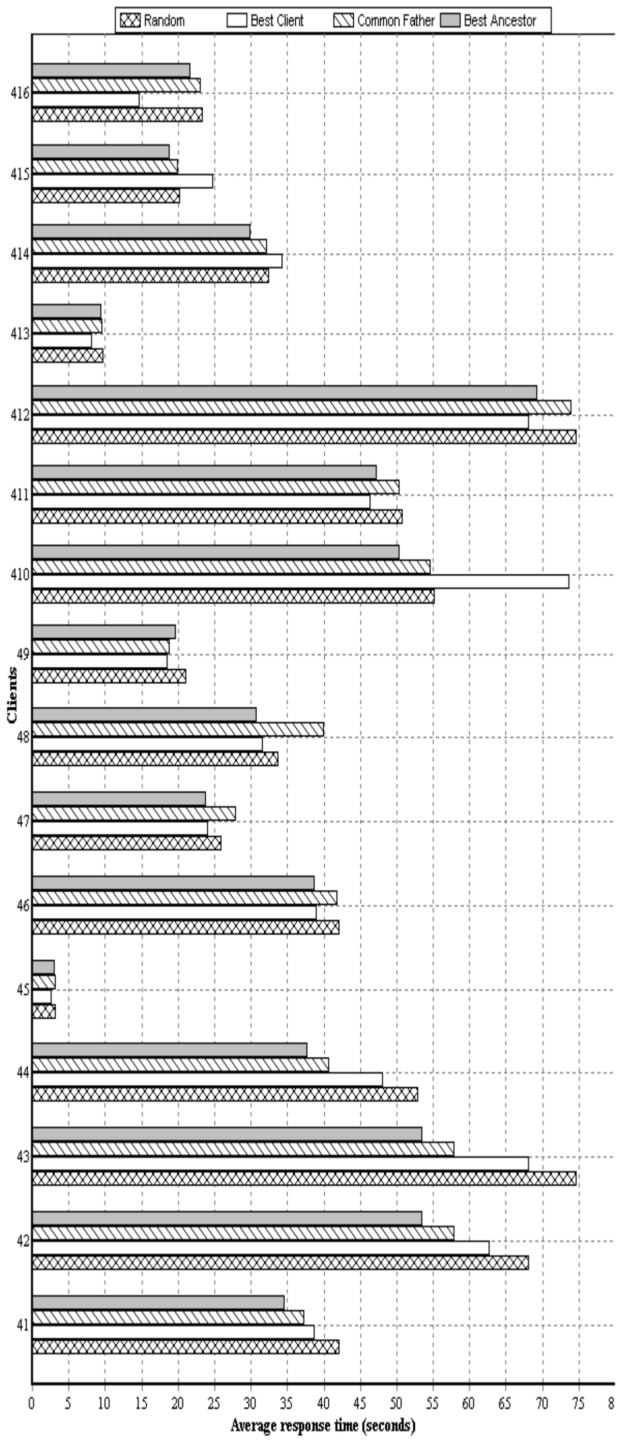

Figure 4: Average time response per clients

During the simulation study, we have noticed that if we make dynamic suppression of replicas weakly requested (the requests number produced on these replicas doesn't exceed a certain threshold), the performances will improve even more.
After the suppression of the replicas that the request number produced on them doesn't exceed a threshold equal 10, the results presented in figure 5 show a decrease of 13.39% of the average time response of this model compared to the common father model, of 14.78% compared to the model of the best client and de 18.37% compared with the random distribution.

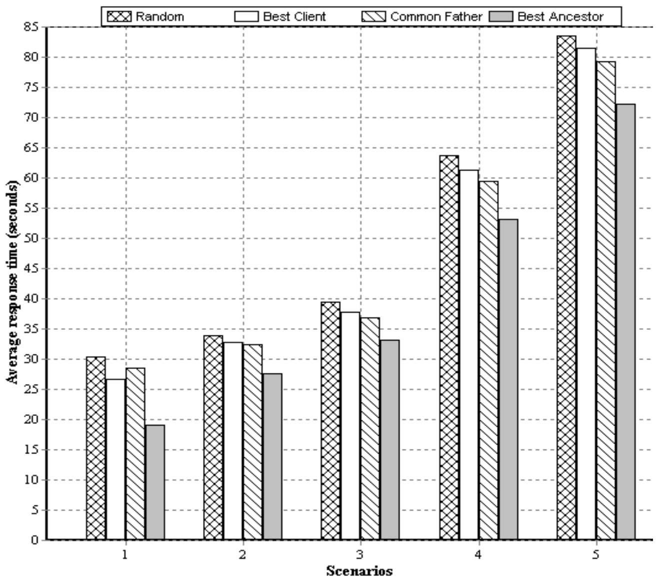

Figure 5: Average time response per scenario after suppression.

5. CONCLUSION AND FUTURE WORKS

In this work, we have studied the problem of the global access cost in a grid system for the replicated data. The problem was formulated such as an optimization subject of a cost model linked mainly to two cost functions: read cost and write cost by node, and linked to two essential parameters: reads number and writes number by node. So, this model allowed us to conceive an efficient algorithm of dynamic placement of replicas.

The logical data grid model is based on a tree architecture where the bandwidth is variable according the tree levels and the nodes of the same level have the same characteristics. The results show that the use of the dynamic replication algorithm based on our model improves the access performances to data in the grid. These results are promising. Nevertheless, they are based on specific work environments.

In the future work we project to validate our model on real data grids by implementing the proposed approach in a more realistic environment such as Globus. Furthermore, this approach studies will be focused on the contribution of the prediction of the requests types'arrival by using the statistic laws (normal distribution, Poisson distribution,…).

REFERENCES

[1] M. Mat Deris, J.H. Abawajy and H.M. Suzuri, "An Efficient Replicated Data Access Approach for Large-Scale Distributed Systems", IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2004), April 19 - 22, 2004, Chicago, Illinois, U.S.A, 2004.

[2] S. Goel, H. Sharda and D. Taniar, "Replica synchronisation in grid databases", Int. J. Web and Grid Services, pp. 87-112, Vol. 1, N°. 1, 2005.

[3] CERN DataGrid Project: http://www.cern.ch/grid

[4] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the Grid : Enableing Scalable Virtual Organizations", International J. Supercomputer Applications, Vol. 15, N°.3, 2003.

[5] H. Lamehamedi, B. Szymanski, Zujun Shentu, and E. Deelman. "Data Replication Strategies in Grid Environments". In Proc. 5th International Conference on Algorithms and Architecture for Parallel Processing, ICA3PP'2002, pp. 378-383, Bejing, China, October 2002.

[6] H. Lamehamedi, Z. Shentu, B. Szymanski and E. Deelman, "Simulation of Dynamic Data Replication Strategies in Data Grids", In: Proc. 12th Heterogeneous Computing Workshop (HCW2003). Nice, France.

[7] C. H. Leiserson, "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing," IEEE Transactions on Computers, Vol. C-34, N°. 10, pp. 892-901, October 1985

[8] NS network simulator http://www.mash.cs.berkeley.edu/ns

[9] B. Parhamu, "Introduction to Parallel Processing : Algorithms and Architectures", Plenum, 1999.

[10] M. Raynal, "Gestion des données réparties: problèmes et protocoles. T3 : une introduction aux principes des systèmes répartis", Eyrolles, 1992.

[11] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs", 3rd ed. Springer-Verlag, Berlin Heidelberg New York 1996.

[12] J. Xu, B. Li, D. Lun Lee,"Placement Problems for Transparent Data ReplicationProxy Services". IEEE Journal on selected areas in Communications, Vol. 20, N°. 7, September 2002.

[13] D. Milojicic, "Peer to Peer Technology", HP Labs Technical Report, HPL-2002-57, http://www.hpl.hp.com/tehcreports/ 2002/HPL-2002-57.html

[14] A. Ripeanu and I. Foster, "A Decentralized, adaptive replica location mechanism", Proceedings of 11th IEEE International Symposium on High Performance Distributed Computing (HPDC-11), Edinburgh, Scotland, July 2002.

[15] P. Garbacki, D. H. J. Epema, and M. van Steen, "The Problem of Broker Placement in Peer-to-Peer Overlay Networks", Technical Report, PDS, Delft University of Technology, May 2004.

[16] James D. Guyton and Michael F. S., "Locating Nearby Copies of Replicated Internet Servers", University of Colorado, TR CU-CS-762-95.

[17] J. Subhlok, P. Lieu, and B. Lowekamp, "Automatic Node Selection for High Performance Application on Networks", Proceedings of the 7th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, May 1999.

[18] R.L. Carter and M.E. Crovella, "Server Selection using Dynamic Path Characterization in Wide-Area Networks", Proceedings of IEEE Infocom'97, April 1997.

[19] K. Ranganathan and I. Foster. "Identifying Dynamic Replication Strategies for a High Performance Data Grid". In Proc. of the International Grid Computing Workshop, Denver, CO, November 2001.