2010 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing

# An Efficient Replication Strategy for Dynamic Data Grids

Faouzi BEN CHARRADA, Habib OUNELLI, Hanène CHETTAOUI
*Department of Computer Sciences*
*Faculty of Sciences of Tunis*
*University Campus, Tunis 1060, Tunisia*
*f.charrada@gnet.tn,habib.ounelli@fst.rnu.tn,*
*hanene1ch@gmail.com*

***Abstract*—Data Grid provides geographically distributed storage resources for large-scale data-intensive applications that generate large data sets [1]. Because data is the important resource in data grids, an efficient management is needed to minimize the response time of applications. Replication is typically used in data grids to improve access time and to reduce the bandwidth consumption. In this paper, we propose a new replication strategy for dynamic data grids which takes into account the dynamicity of sites. Indeed, the dynamicity of sites is an important challenge in grids. Our strategy helps to increase file availability, to improve the response time and to reduce the bandwidth consumption. We evaluate our strategy through simulation using OptorSim. The results we have obtained appear to be promising.**

***Keywords*-Data grid; replication; dynamicity; OptorSim; Response time;**

## I. INTRODUCTION

Several scientific applications such as Particle Physics, High Energy Physics and Genetics, to cite a few, generate an important amount of data which can reach terabytes and even petabytes. It's difficult, even impossible, to store such amount of data in the same location. Moreover, an application may need data produced by another geographically remote application. For this reason, we resorted more to data grids. A data grid is composed of a set of geographically distributed sites where each one maintains a set of files. The data distributed across a grid must be available and accessible to several applications with a reasonable performance. Replication is often used as a solution to this problem. Replication consists in storing several copies of the same file in multiples sites across the grid in order to increase the data availability, to improve access time, to enhance fault tolerance and to reduce the bandwidth consumption. An effective replication strategy must answer the following questions [2][3]:

(a) Which files must be replicated?
(b) Where to place the candidate files for replication (new replicas) and
(c) How to select best replica of a file among many replicas available in the grid.

The majority of works on replication have considered only static grids (having a fixed number of sites). For several types of grids, this assumption is not realistic. Indeed, new sites can join the grid and others can leave to join possibly again later. This dynamicity becomes a fundamental constraint to establish an effective replication strategy.

In this paper, we propose a new replication strategy for dynamic grids. Our strategy exploits the replicas placement and file requests in order to converge towards a global balancing of the grid load.

We focus on read-only-access as most grids have very few dynamic updates because they tend to use a "load" rather than "update" strategy [4].

Our experiments, by simulation with an extension of *OptorSim* [5], show that our strategy is better than "No Replication" and "Best Client" [3][6].

In addition of the introduction, this article contains four sections and will be organized as follow. Section 2 gives an overview of previous work on grid replication. Section 3 describes our proposed replication strategy. Section 4 evaluates our strategy by simulation and presents the obtained results. We end the paper with our conclusions and future direction in section 5.

## II. RELATED WORK

Several approaches have been proposed in the literature for the replication problem in data grids. We discuss below some representative strategies.

Ranganathan and Foster [3][6] discuss various replication strategies for hierarchical data grids. They distinguish between replication and caching. Indeed, replication is defined as server side phenomenon and caching is assumed to be a client side phenomenon. They propose six replication strategies:1)No Replication; 2)Best Client; 3)Cascading Replication; 4)Plain Caching; 5)Caching plus Cascading Replication; 6)Fast Spread. We will focus our discussion on "Best Client" and "No Replication" strategies because we will compare our strategy with them:

1) No Replication: The data is available only at the root of the hierarchy. This strategy can not adapt to change in user behavior.
2) Best Client: The replica is created at the client that has generated the most requests for that file.

978-0-7695-4237-9/10 $26.00 © 2010 IEEE
DOI 10.1109/3PGCIC.2010.13

Ranganathan et al. [7] propose a replication strategy in a dynamic, decentralized P2P environment. The aim of their model is to maintain a level of data availability. The proposed strategy computes first the number of replicas needed per each file for a certain availability threshold. Then, the best site for the replica placement is determined based on the bandwidth and the storage cost.

Liu and Wu [8] propose a replica placement strategy in order to ensure a balanced workload between sites. They propose an algorithm for selecting the best sites for replica placement. The aim of this algorithm is to balance workload among the replicas. The algorithm determines the minimum number of required replicas and their locations when the maximum workload for each server and the data usage from each user are given.

Al-Mistarihi and Yong [2] propose a replication strategy in data grids. They select first the best file to be replicated based on the number of requests and the lifespan of each file. They place then the selected file in the best site based on the bandwidth and the number of requests.

Rahman et al. [4] propose 4 replication algorithms in data grids. These strategies are based on utility and risk. The utility and risk are calculated using the current network and the user requests. Each algorithm select the best site to place a replica. An improvement is proposed in [9][10][11] by selection $p$ sites to place replica using the three models, namely p-median, p-center and multi-objective.

## III. Proposed strategy

In this section, we propose a replication strategy for dynamic data grids that helps to increase file availability, to improve the response time and to reduce the bandwidth consumption. Our strategy is evolutionary in the sense that it aims to achieve stability of the total load grid in time. Our strategy is made up of the following steps:

1) Selection of the best candidate files for replication;
2) Determination of the best sites for files placement which are selected in the previous step;
3) Selection of the best replica.

### A. Selection of the best candidate files for replication

The first step of our strategy is to identify the candidate files for replication based on the following parameters:

- The number of times that a file $F_j$ has been requested, noted $\#Request_{Fj}$ ;
- The number of available copies of a file $F_j$ on the whole grid system, noted $\#Replica_{Fj}$.

We consider that a file $F_j$ must be replicated if it has been requested too much and there are not enough copies of $F_j$ in the grid. For that purpose, a metric is proposed, called average weight of a file $F_j$, noted $AVG\_WEIGHT_{Fj}$, and defined by the following expression:

$$AVG\_WEIGHT_{Fj} = \frac{\#Request_{Fj}}{\#Replica_{Fj}} \quad (1)$$

The average weight of a file $F_j$ represents the average number of requests for a replica of $F_j$.

Similarly to the average weight of a file, we define a metric called average weight of the entire grid, noted $AVG\_WEIGHT$, and defined by the following expression:

$$AVG\_WEIGHT = \frac{\sum_{j=1}^{n} \#Request_{Fj}}{\sum_{j=1}^{n} \#Replica_{Fj}} \quad (2)$$

Where $n$ represents the total number of requested files in the grid.

These two metrics are used in Algorithm 1 for selecting the candidate files for replication. Algorithm 1 takes as input the requests number and replicas number of each requested file as a list of triplets, noted $S_F$= $\{(F_j, \#Request_{Fj}, \#Replica_{Fj})\}$.

The first loop calculates the average weight of each requested file and prepares variables for the average weight of the entire grid. The second loop selects the best candidate files for replication. A file $F_j$ is considered as a good candidate file for replication if its average weight exceeds the average weight of the entire grid.

**Algorithm 1** Selection of the candidate files for replication

**Require:** $S_F$= $\{(F_j, \#Request_{Fj}, \#Replica_{Fj})\}$
**Ensure:** $CandidateFile$: List of the candidate files for replication
  $Request\_Sum \leftarrow 0$ {Calculating the numerator of (2)}
  $Replica\_Sum \leftarrow 0$ {Calculating the denominator of (2)}
  $CandidateFile \leftarrow \emptyset$
  **for** $j = 1$ to $\mid S_F \mid$ **do**
    $AVG\_WEIGHT_{Fj} \leftarrow \frac{\#Request_{Fj}}{\#Replica_{Fj}}$
    $Request\_Sum \leftarrow Request\_Sum + \#Request_{Fj}$
    $Replica\_Sum \leftarrow Replica\_Sum + \#Replica_{Fj}$
  **end for**
  $AVG\_WEIGHT \leftarrow \frac{Request\_Sum}{Replica\_Sum}$
  **for** $j = 1$ to $\mid S_F \mid$ **do**
    **if** $AVG\_WEIGHT_{Fj} \geq AVG\_WEIGHT$ **then**
      $CandidateFile \leftarrow CandidateFile \bigcup F_j$
    **end if**
  **end for**
  return $CandidateFile$

### B. Determination of the best sites for candidate files placement

After selecting the best candidate files for replication, we must determine the best sites to place them. The choice of these sites is very important in order to minimize the response time and to reduce the consumption bandwidth for the next requests. In order to succeed this choice, our strategy takes into account the following parameters:

(a) Requests number for each file $F_j$ by each site $S_i$, noted $\#Request_{Si,Fj}$

(b) Utility of each site $S_i$ regarding to the grid, noted $Utility_{Si}$.

The last parameter models the dynamic behavior of the grid. As a site can, at any time, leave the grid and possibly join again later, its utility changes dynamically. This parameter is re-calculated for each requesting site $S_i$ according to (3).

$$Utility_{Si} = 1 - \frac{\#Failure_{Si}}{\#SiteRequest_{Si}} \quad (3)$$

Where:
$\#Failure_{Si}$ is the number of times where the site $S_i$ is failing and $\#SiteRequest_{Si}$ is the total number of times where the site $S_i$ have been requested.
As shown in (3), the utility of a site $S_i$ involves the number of times where $S_i$ did not answer to a file request for a reason such as $S_i$ is not in the grid. It should be noted that a request of a file $F_j$ is directed to a site $S_i$ if it has $F_j$.

We consider that a site $S_i$ is a "good" candidate to replicate $F_j$ if $S_i$ requests $F_j$ frequently and $S_i$ is very reliable for the grid. To quantify this criterion, we propose a metric called estimated number of requests for a file $F_j$ by a site $S_i$, noted $\#ESTIM\_REQUEST_{Si,Fj}$, and defined by (4).

$$\#ESTIM\_REQUEST_{Si,Fj} = \#Request_{Si,Fj} * Utility_{Si} \quad (4)$$

Similary, we define a metric called average number of requests for a file $F_j$ by $n$ sites, noted $AVG\_\#REQUEST_{Fj}$, and defined by the expression (5).

$$AVG\_\#REQUEST_{Fj} = \frac{\sum_{i=1}^{n} (\#Request_{Si,Fj} * Utility_{Si})}{n} \quad (5)$$

These two metrics are used in Algorithm 2 that places the best candidate files for replication (Algorithm 1) in the best sites.

Algorithm 2 takes as input the requesting sites and their utilities as a set of pairs, noted $S_U = \{(Si, Utility_{Si})\}$, and a matrix, noted $RequestM_{F,S}$, where rows are the requesting sites and columns are the candidate files for replication . A cell (row $i$, column $j$) represents the value of $\#Request_{Si,Fj}$.
The outer loop examines all candidate files. The first inner loop calculates the numerator of (5). The second inner loop selects the best sites for placement of a candidate file $F_j$ for replication. A site $S_i$ is selected to replicate a file $F_j$ if the estimated number of requests for $F_j$ by $S_i$ exceeds the average number of requests for $F_j$ by all sites. The result is a set of pairs $(S_i, \{F_k\})$ where $\{F_k\}$ is the set of files to replicate in $S_i$.

**Algorithm 2** Placement of the candidate files for replication in the best sites

**Require:** $RequestM_{F,S}$, $S_U = \{(S_i, Utility_{Si})\}$
**Ensure:** Placement $P$ ={$(S_i, \{F_k\})$}
  $P \leftarrow \emptyset$
  **for** each file $F_j$ in $RequestM_{F,S}$ **do**
    $CandidateSite \leftarrow$ The set of sites that have requested $F_j$
    $Est\_Request\_Sum$ = 0
    **for** each site $S_i$ in $CandidateSite$ **do**
      $Est\_Request\_Sum \leftarrow Est\_Request\_Sum + (\#Request_{Si,Fj} * Utility_{Si})$ {Calculating the numerator of (5)}
    **end for**
    $AVG\_\#REQUEST_{Fj} \leftarrow \frac{Est\_Request\_Sum}{|CandidateSite|}$
    **for** each site $S_i$ in $CandidateSite$ **do**
      $\#ESTIM\_REQUEST_{Si,Fj} \leftarrow \#Request_{Si,Fj} * Utility_{Si}$
      **if** $\#ESTIM\_REQUEST_{Si,Fj} \geq AVG\_\#REQUEST_{Fj}$ **then**
        $P \leftarrow P \cup (S_i, F_j)$
      **end if**
    **end for**
  **end for**

### C. Selection of the best replica

When a file $F_j$ is requested, either to be replicated or to satisfy a request, the selection of the best replica arises. The selection of the best replica does not depend only on bandwidth, as in the literature, but also on the utility function of the sites of the grid. Indeed, if we transfer a file $F_k$ from a site $S_j$ to a site $S_i$, it is preferable that $S_j$ is the nearest to $S_i$ to reduce transfer time. This is even better than if $S_j$ is most reliable for the grid (and therefore the most available). A site, always connected to the grid, has not the same utility as another site that connects and disconnects frequently.
For these reasons, our strategy will jointly exploit these two parameters (bandwidth and utility). Note by $BW_{Si,Sj}$ the bandwidth between $S_i$ and $S_j$. The selection of the best site containing the requested file $F_k$ is achieved by (6).

$$Max(BW_{Si,Sj} * Utility_{Sj}) \quad (6)$$

The experiments presented in the next section of our replication strategy seem to support the effectiveness of our approach.

## IV. Experimental evaluation and comparison

We use the ***OptorSim*** simulator to evaluate and compare our strategy to "No Replication " and "Best Client" strategies. These two strategies, as well as our strategy, have several common parameters. As our strategy, "Best Client" uses the files requests to select sites and candidate files for replication. To highlight the importance of the dynamic

placement, we compare our strategy to "No Replication" which considers the static placement.

***Note***: Our proposed strategy is noted "Periodic Optimiser" in the experiments.

**Experiments with OptorSim**

Recall that we have made extensions to the ***OptorSim*** simulator in order to model dynamic grids.

The experiment consists in submitting a variable number of jobs as is done in all strategies. At the end of each simulation, we get the total response time and the *ENU* parameter [5][2][12] (*The Effective Network Usage*) which is the most important parameter in quantifying the effectiveness of any replication strategy in data grids. The *ENU* parameter is defined by (7).

$$ENU = \frac{N_{remote\ file\ accesses} + N_{file\ replications}}{N_{remote\ file\ accesses} + N_{local\ file\ accesses}} \quad (7)$$

*ENU* defines the ratio of files transferred to files requested. A low value of *ENU* indicates that the replication strategy is effective [5] [12].

Table I shows the response time in *ms* of each strategy and for different numbers of jobs. The last column shows the percentage gain of our strategy compared to other strategies[1]. We notice, from this table, that our strategy is more effective when the number of jobs increases. It should be noted, on this subject, that the experiments made by "No Replication" and "Best Client" are limited to a low number of jobs. In practise, most grids manage a very high number of jobs. In this case, the percentage gain of our strategy goes far beyond that of "No Replication" and "Best Client".

Table I
RESPONSE TIME AND PERCENTAGE GAIN OF OUR STRATEGY COMPARED TO OTHER STRATEGIES

| Number of jobs | No Replication | Best Client | Periodic Optimiser | Gain in % |
|---|---|---|---|---|
| 100 | 3711 | 3644 | 4333 | -18,90% |
| 300 | 9349 | 9374 | 9661 | -3,33% |
| 600 | 18247 | 12755 | 11792 | 7,54% |
| 1000 | 30007 | 22143 | 19217 | 13,21% |
| 2000 | 85680 | 64959 | 48066 | 26% |
| 3000 | 115987 | 72125 | 48732 | 32,43% |
| 4000 | 145253 | 90312 | 50557 | 44,01% |
| 5000 | 178821 | 121136 | 53067 | 56,19% |
| 5500 | 192424 | 113227 | 66976 | 40,85% |
| 6000 | 204477 | 133281 | 70197 | 47,33% |
| 6500 | 218672 | 141192 | 72098 | 48,94% |

The same experiments of Table I are represented by curves in Figure 1. The response time of our strategy improves when the number of jobs increases. Indeed, the replication of several files in several sites decreases the remote access number.

[1] $\frac{Min(Value_{Best\ Client}, Value_{No\ Replication}) - Value_{Periodic\ Optimiser}}{Min(Value_{Best\ Client}, Value_{No\ Replication})} *$ 100

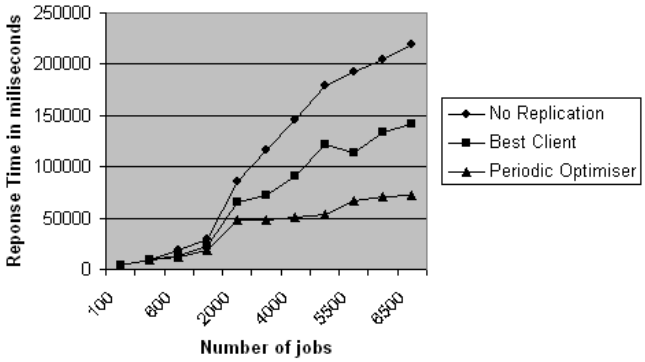


Figure 1. Response time for various replication strategies

Table II and Figure 2 show the *ENU* evaluation for the three strategies under the same conditions. Again, our strategy gives a *ENU* much better than "No Replication" and "Best Client". The gain is about 50 % and more when we reached the 2000 jobs.

Table II
ENU AND PERCENTAGE GAIN OF OUR STRATEGY COMPARED TO OTHER STRATEGIES

| Number of jobs | No Replication | Best Client | Periodic Optimiser | Gain in % |
|---|---|---|---|---|
| 100 | 1 | 0,75 | 0,54 | 28,86% |
| 300 | 1 | 0,51 | 0,38 | 26,18% |
| 600 | 1 | 0,49 | 0,36 | 26,94% |
| 1000 | 1 | 0,5 | 0,34 | 31,49% |
| 2000 | 1 | 0,56 | 0,32 | 43,66% |
| 3000 | 1 | 0,44 | 0,3 | 32,76% |
| 4000 | 1 | 0,43 | 0,24 | 43,46% |
| 5000 | 1 | 0,46 | 0,21 | 53,49% |
| 5500 | 1 | 0,44 | 0,22 | 50,15% |
| 6000 | 1 | 0,45 | 0,21 | 52,88% |
| 6500 | 1 | 0,44 | 0,20 | 53,39% |

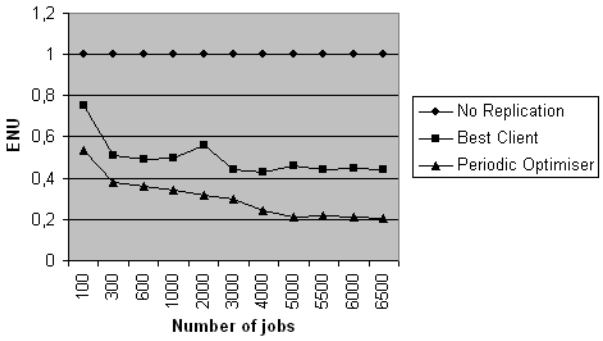


Figure 2. ENU for various replication strategies

## V. Conclusions and future work

Data replication is considered a major technique for reducing data access and bandwidth consumption in data grids. Replication aims also to improve data availability. In this paper, we propose a replication strategy for data grids which takes into account the dynamicity of sites. Indeed, the dynamicity of sites is an important challenge in grids. Our empirical data suggests that our strategy helps to increase the data availability, to improve the response time and to reduce the bandwidth consumption. We propose firstly an algorithm which selects the best candidate files for replication based on requests number and copies number of each file. We selects then the best sites for the new replicas placement based on requests number and utility of each site regarding to the grid. We focus also in selecting of the best replica with taking account the bandwidth and the utility of each site. The obtained experiments show promising results. In future work, we will study a deletion strategy when a limited storage space is available. Additionally, we will study the impact of the period length on the efficiency of our replication strategy.

## References


[1] Z. S. H. Lamehamedi, B. Szymanski and E.Deelman, "Data replication strategies in grid environments," in *Proc. (IEEE) International Conference on Algorithms and Architectures for Parallel Processing*, Oct. 2002, pp. 378–383.

[2] H. E.AL-Mistarihi and C. Yong, "Replica management in data grid," *International Journal of Computer Science and Network Security*, vol. 8, pp. 22–32, 2008.

[3] K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high performance data grid," in *Proc. (Springer-Verlag) Second International Workshop on Grid Computing*, Nov. 2001, pp. 75–86.

[4] K. B. R. M.Rahman and R. Alhajj, "Replica placement in data grid: Considering utility and risk," in *Proc. International Conference on Information Technology: Coding and Computing*, Apr. 2005, pp. 354–359.

[5] J. F. A. M. C. N. K. S. D. G.Cameron, R. Carvajal-Schiaffino and F. Zini, "Optorsim v2.1 installation and user guide," 2006.

[6] K. Ranganathan and I. Foster, "Design and evaluation of dynamic replication strategies for a high-performance data grid," in *Proc. International Conference on Computing in High Energy and Nuclear Physics*, Sep. 2001.

[7] A. I. K. Ranganathan and I. Foster, "Improving data availability through dynamic model-driven replication in large peer-to-peer communities," in *Proc. (IEEE) IEEE/ACM International Symposium on Cluster Computing and the Grid*, May 2002.

[8] P. Liu and J. Wu, "Optimal replica placement strategy for hierarchical data grid systems," in *Proc. (IEEE) IEEE International Symposium on Cluster Computing and the Grid*, May 2006, pp. 417–420.

[9] K. B. R. M.Rahman and R. Alhajj, "Replica placement strategies in data grid," *Journal of Grid Computing*, vol. 6, pp. 103–123, 2008.

[10] ——, "Replica placement design with static optimality and dynamic maintainability," in *Proc. (IEEE) IEEE International Symposium on Cluster Computing and the Grid*.

[11] ——, "Performance evaluation of different replica placement algorithms," *International Journal of Grid and Utility Computing*, vol. 1, pp. 121–133, 2009.

[12] A. D. A. M. C. Nicholson, D. G.Cameron and K. Stockinger, "Dynamic data replication in lcg 2008," *Concurrency and Computation: Practice and Experience*, vol. 20, pp. 1259–1271, 2008.