Journal of Network and Computer Applications 34 (2011) 575–580

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Enhanced Fast Spread Replication strategy for Data Grid

Mohammad Bsoul [a,*], Ahmad Al-Khasawneh [b], Emad Eddien Abdallah [b], Yousef Kilani [b]

[a] The Hashemite University, Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, Department of Computer Science and Applications (CSA), P.O. Box 150459, Zarqa 13115, Jordan

[b] The Hashemite University, Faculty of Prince Al-Hussein Bin Abdallah II for Information Technology, Department of Computer Information Sciences (CIS), P.O. Box 150459, Zarqa 13115, Jordan

ARTICLE INFO

Article history:
Received 10 April 2010
Received in revised form
14 November 2010
Accepted 5 December 2010
Available online 22 December 2010

Keywords:
Data Grid
Replication strategy
Fast Spread
Size
Number of requests
Frequency of requests
Last request time
Simulation

ABSTRACT

Data replication is used in Data Grid to enhance data availability and fault tolerance. However, replication should be used wisely because the storage size of each node reside on the Data Grid is limited. Thus, the node must accommodate only the important replicas. In this paper, a dynamic replication strategy that takes into account the number and frequency of requests, the size of the replica, and the last time the replica was requested is proposed. This strategy is an enhanced version of Fast Spread replication strategy. The simulation results show that the new proposed strategy attained better performance than Fast Spread with Least Recently Used (LRU) and Fast Spread with Least Frequently Used (LFU) in terms of total response time and total bandwidth consumption.

1. Introduction

A Data Grid consists of a collection of geographically distributed computer and storage resources located in different places, and enables users to share data and other resources (Figueira and Trieu, 2008; Cameron et al., 2004; Lamehamedi et al., 2002).

In the Data Grid, when a user requests a file, a large amount of bandwidth could be spent to send the file from the server to the client. Moreover, the delay involved could be high. Thus, it could be beneficial to create replicas of the same file at different locations. The main goals of using replication are to reduce access delay and bandwidth consumption (Ranganathan and Foster, 2001b). There are two kinds of replication: static and dynamic replication. Dynamic replication (Tang et al., 2005; Dong et al., 2008; Rasool et al., 2008; Chang and Chang, 2008; Park et al., 2003; Hong et al., 2008; Zhao et al., 2008; Wu et al., 2008) has an advantage over static replication (Cibej et al., 2005) because it can adapt to changes in user behavior. Some of the well-known dynamic replication strategies that have already been implemented are No Replication or Caching, Best Client, Cascading Replication, Plain Caching, Caching plus Cascading Replication, and Fast Spread (Horri et al., 2008; Chang et al., 2008; Ranganathan and Foster, 2001a).

Fast Spread is one of the best replication strategies especially for random request patterns (Ranganathan and Foster, 2001b). In this strategy which is our main concern in this paper, a replica of the requested file is stored at each node along its path to the requester. If the storage of one of these nodes is full, a group of existing replicas (that contains one or more replicas) needs to be replaced with the new replica. Fast Spread can use one of the many replacement strategies to determine the replicas that need to be replaced. Two of the well-known replacement strategies are LRU and LFU. Fast Spread with LRU discards the least recently used replicas first, while Fast Spread with LFU discards the least frequently used replicas first. The problem arises when this group of existing replicas is more important than the new replica. The aim of this paper is to propose a new strategy named Enhanced Fast Spread (EFS) strategy to solve this problem. The new proposed strategy is based on Fast Spread but superior to it.

Because Client nodes have limited storage capacity and as a result cannot accommodate all the replicas they request, only important replicas should be kept on their storages. By keeping only the important replicas, a large number of their replica requests can be served locally. EFS strategy takes that into account and keeps only the important replicas while the other less important replicas are replaced with more important replicas. This is achieved by using a dynamic threshold that determines if the requested replica

* Corresponding author. Tel.: +962 5 3903333x4573; fax: +962 5 3826625.
E-mail addresses: mbsoul@hu.edu.jo (M. Bsoul), Akhasawneh@hu.edu.jo (A. Al-Khasawneh), Emad@hu.edu.jo (E.E. Abdallah), ymkilani@hu.edu.jo (Y. Kilani).

1084-8045/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2010.12.006

should be stored at each node along its path to the requester. This strategy takes four factors into consideration when calculating the threshold: the number of requests, the frequency of requests, the size of the replica, and the last time the replica was requested. The number of requests shows how many times the replica has been requested by its node. The frequency of requests shows how many times the replica has been requested by its node within a specific time interval. Since the storage space is the main concern in the Data Grid, the size of replica is also an important factor in deciding if the replica should be stored. The number and frequency of requests in addition to the last time the replica was requested give an indication of the probability of requesting the replica again.

The rest of this paper is organized as follows. Section 2 describes the underlying network structure. Section 3 presents the new proposed strategy. Section 4 describes the metrics for measuring the performance of the strategies. Section 5 explains how the simulation is configured. Section 6 discusses the simulation results. Section 7 concludes the paper and describes future directions.

2. Network structure

The network structure consists of a Server node and a number of Client nodes that interact with each other. The Server node is the node with the main storage which contains all the data in the Data Grid. On the other hand, Client nodes are the nodes that issue requests. Each Client node has a storage that is small compared with that of a Server node and thus cannot accommodate all the requested replicas. Therefore, some of the replica requests are served non-locally.

In this structure, there is the shortest path from each Client node to the Server node. When a Client node requests a replica, it first searches its own storage. If it is stored there, it just uses it. Otherwise, the client traverses the shortest path until it finds a copy of the replica. Then, this copy is transferred to that client. Figure 1 shows the shortest path tree of the used network topology. The used network topology is a complete graph.

3. EFS strategy

As mentioned in the introduction section, Fast Spread strategy stores a replica of the file at each node along its path to the requester. If the nodes' storage is full and there is no space for storing the replica, a group of replicas needs to be removed from the storage in order to store the new replica. But what if that group of replicas that are to be removed are more important than the new replica? Fast Spread strategy does not take this into account and replaces that group with the new replica even if it is more important than the new replica.

On the other hand, the EFS takes this into consideration and replaces that group only if the value of that group is smaller than the value of the requested replica. The calculation of a group value (GV) is shown in (1), while the calculation of replica value (RV) is shown in (2). For a given node, the replica with the smallest RV is the least important replica, while the replica with the largest RV is the most important one. In the calculation of RV and GV, four

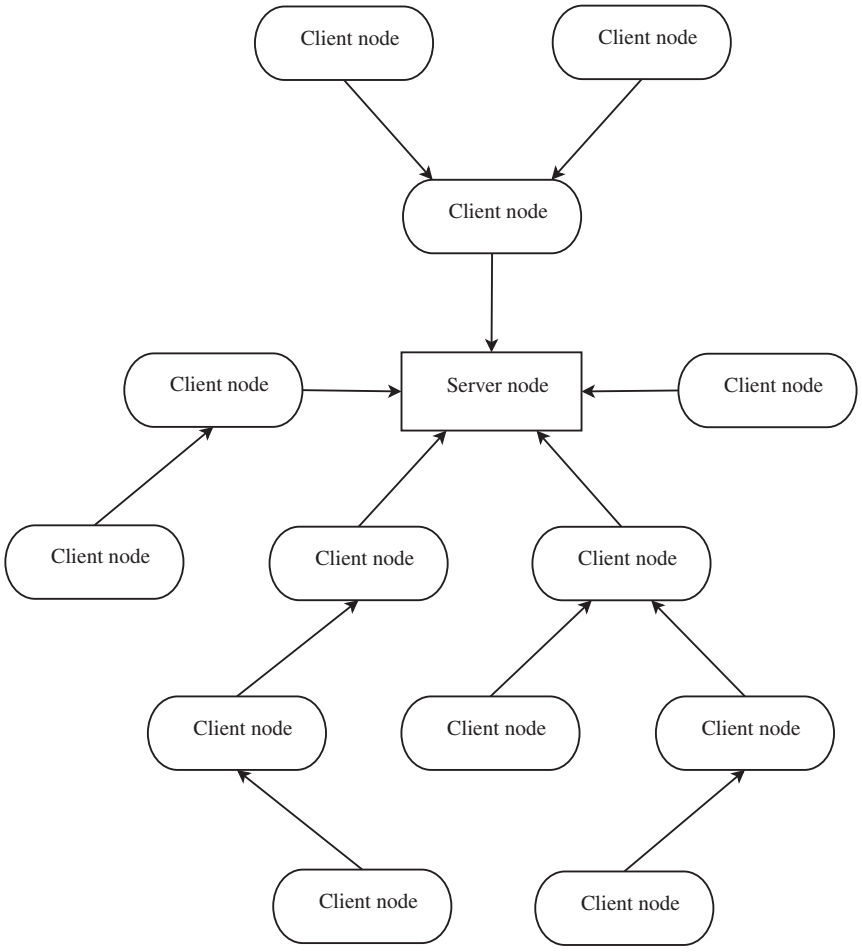

Fig. 1. Network structure.

important factors are considered. These factors are the number of requests, the frequency of requests, the size of the replica, and the last time the replica was requested. The number of requests, the frequency of requests, and the last time the replica was requested are important factors because they give an indication of the probability of requesting the replica again. Because Client nodes have limited storage space, the size of replicas is an important factor as well.

$$GV = \frac{\sum_{i=1}^{n} NOR_i}{\sum_{i=1}^{n} S_i} + \frac{\sum_{i=1}^{n} NORFSTI_i}{FSTI} + \frac{1}{CT - \frac{\sum_{i=1}^{n} LRT_i}{n}} \quad (1)$$

where n is the number of replicas in the group, NOR_i the number of requests of replica i in the group, S_i the size of replica i in the group, $FSTI$ the frequency specific time interval, $NORFSTI_i$ the number of requests of replica i in the group within the FSTI, CT the current time, and LRT_i the last request time of replica i in the group.

$$RV = \frac{NOR}{S} + \frac{NORFSTI}{FSTI} + \frac{1}{CT - LRT} \quad (2)$$

where NOR is the number of requests of the replica, S the size of the replica, $FSTI$ the frequency specific time interval, $NORFSTI$ the number of requests of the replica within the FSTI, CT the current time, and LRT the last request time of the replica.

In this strategy, each node stores the NOR of each replica resides on it. For a given node, the NOR of a replica is increased by one each time that replica is requested by that node.

Pseudocode 1 shows the algorithm of this strategy. For definitions of variables used in the pseudocode, refer to Table 1.

When a node requests an existing replica, it just uses it. However, if the replica does not exist on that node, it starts searching for it on every node on the shortest path from $RN+1$ to the main server, where $RN+1$ is the requesting node's successive node on the shortest path.

When the requested replica is found on one of the nodes on the shortest path, it is brought backward to the requesting node. In the original Fast Spread replication strategy, that replica is copied to every node it visits when it is brought backward to the requesting node. In contrast to Fast Spread, EFS does not necessarily copy that replica to every node it visits when it is brought backward. It is copied to the visited node under two conditions. The first condition arises when the visited node has enough free storage space to store it. The second condition arises when the node's free storage space is less than the size of the requested replica, and the replica is found more important than a group of existing replicas. The size of that group must be greater than or equal to the storage space still needed to make the node able to store the requested replica. In this case, that group of replicas is replaced with the requested replica. The requested replica is considered more important than that the group of existing replicas if its value is greater than the value of that group. In this strategy, it is assumed that the value of the free storage space is zero.

Pseudocode 1. EFS strategy

```
1  Initialize SOS to 0;
2  if RR exists on RN then
3  | UseRR;
4  else
5  | for i = 2 to NSPList.size do
6  | | if RR exists on NSPList(i) then
7  | | | for j = NSPList(i−1) to 1 do
8  | | | | if CNFSS ≥ RR.Size then
9  | | | | | Copy RR;
10 | | | | else
11 | | | | | for x = 1 to ReplicaList.size do
12 | | | | | | if SOS+CNFSS < RR.Size then
13 | | | | | | | SOS = SOS+SizeList(x);
14 | | | | | | else
15 | | | | | | | Break;
16 | | | | | | end
17 | | | | | end
18 | | | | | GV = (Σ_{y=1}^{x−1} NORList(y)) / (Σ_{y=1}^{x−1} SizeList(y)) + (Σ_{y=1}^{x−1} NORFSTIList(y))/FSTI + 1 / (CT − (Σ_{y=1}^{x−1} LRTList(y))/(x−1))
19 | | | | | RV_RR = NORRR/SRR + NORRRFSTI/FSTI + 1/(CT−LRTRR)
20 | | | | | if GV < RV_RR then
21 | | | | | | for y = 1 to x−1 do
22 | | | | | | | Delete ReplicaList(y), NORList(y), SizeList(y),
23 | | | | | | | NORFSTIList(y), LRTList(y);
24 | | | | | | end
25 | | | | | | Copy RR;
26 | | | | | end
27 | | | | end
28 | | | end
   | | end
   | end
29 end
```

Table 1
Definitions of pseudocode variables of EFS strategy.

Variable	Definition
RR	Requested replica
RN	Requesting node
CNFSS	Checked node's free storage space
FSTI	Frequency specific time interval
NORRR	Number of requests of RR
SRR	Size of RR
NORRRFSTI	Number of requests of the requested replica within the FSTI
CT	Current time
LRTRR	The last request time of the requested replica
SOS	The variable that contains the sum of sizes of a group of replicas on the checked node
NSPList	The list that contains the nodes on the shortest path from *RN* to the main server
ReplicaList	The list that contains the existing replicas on the checked node sorted in increasing order based on their RV. If two or more replicas have the same RV, they are sorted randomly
NORList	The list that contains how many times each replica in *ReplicaList* has been requested by the checked node
SizeList	The list that contains the sizes of the corresponding replicas in *ReplicaList*
NORFSTIList	The list that contains how many times each replica in *ReplicaList* has been requested within the FSTI by the checked node
LRTList	The list that contains the last request times of the corresponding replicas in *ReplicaList*

Table 2
Metrics of strategies.

$M_1 = TRT \times C1$	TRT=Total response time, and $C1$ is a constant
$M_2 = TBC \times C2$	TBC=Total bandwidth consumption, and $C2$ is a constant

4. Comparison metrics

In the current work, there are n nodes $N_1, N_2, \ldots, N_n$, m groups $G_1, G_2, \ldots, G_m$ and w replicas $R_1, R_2, \ldots, R_w$. Each group contains a set of replicas.

In order to determine the best performing strategy, a suitable set of metrics need to be defined. Two metrics are used to measure the performance of different strategies: total response time and total bandwidth consumption. Both metrics are need to be minimized. Response time is the elapsed time between sending a request for a replica and receiving the requested replica. Minimizing it means that the clients will obtain the replicas they request in less time. If the requested replica exists on the requesting node, the response time is considered zero. The sum of all response times for the duration of the simulation is calculated. Bandwidth consumption is the bandwidth consumed for data transfers when a node requests a replica that does not exist on it. The bandwidth consumed for data transfers should be reduced, because it is limited. The sum of all bandwidth consumptions for the duration of the simulation is calculated.

Table 2 shows the metrics used to measure the performance of strategies. Since the calculated values of total response times and total bandwidth consumptions through the duration of the simulation are expected to be so large, the constants $C1$ and $C2$ are used to minimize their values. In the simulations, the value of each of those constants is set to 0.001.

5. Simulation setup

In this paper, an event-driven simulator written in Java is used for evaluating three different replication strategies which are Fast Spread with LRU (O'Neil et al., 1993), Fast Spread with LFU (Prischepa, 2004), and our proposed strategy named EFS. As mentioned before, Fast Spread with LRU discards the least recently used replicas first, while Fast Spread with LFU discards the least frequently used replicas first. The performance of these strategies is measured under three different scenarios. In the first scenario, the probability of requesting any of the replicas is the same. In the second and third scenarios, there is a group named the most wanted group (MWG) that contains 10% of the total number of replicas. Each node has its own MWG. The probability of making a request for replicas in MWG is higher than the probability of making a request for the rest of replicas. The probability of requesting a replica in MWG is 30% in the second scenario, while it is 50% in the third scenario. The second and third scenarios have a better representation for the real situation in many cases, where some information (replicas) are requested more than the other information. For instance, the probability of requesting an account information for a customer who lives in the same area of a bank branch is higher than the probability of requesting an account information for a customer who lives in a different area. For this bank branch, the account information for same area customers can be considered the MWG.

In each simulation, all the nodes are set to employ one of the strategies under a specific scenario. For each strategy, the simulation is run three times; one for each of the scenarios. Since three strategies are evaluated under three scenarios; the total number of simulation runs is nine. Under each scenario, the performance of EFS strategy is compared with the performance of the other two strategies. In all simulations, the used network topology is a complete graph.

The inter-arrival times of nodes' requests and replicas' sizes follow uniform distribution. The communication latency (CL) between two directly connected nodes is calculated as follows:

$$CL = \frac{FS}{NB} + \frac{D}{PS}$$

where FS is the frame size, NB the network bandwidth, D the distance between the two directly connected nodes, and PS the propagation speed.

Since total bandwidth consumption and total response time are the only metrics measured in the simulations, it is assumed that the time required for replacing a group of replicas and storing the new replica is negligible (equal to zero).

Table 3 shows the simulation parameters and their values.

Table 3
Simulation parameters.

Parameter	Value
Number of nodes	20
Number of replicas	1000
Size of each replica	Between 100 and 1000 Megabit
Number of generated requests	100,000
The inter-arrival times of nodes' requests	Between 0 and 99
Number of groups	10
Storage space for every client node	50,000 Megabit
Storage space for server node	So large so it can accommodate all the replicas on the Data Grid
Number of replicas within each group	100
Network bandwidth	10 Mbps
Distance between every two directly connected nodes	Between 1 and 1000 km
Propagation speed	6×10^3 Kmps
Frequency specific time interval	10,000
$C1$	0.001
$C2$	0.001

6. Simulation results and discussion

In this section, the performance of various strategies is measured under the three different scenarios mentioned in the previous section. Figures 2 and 3 show the total response time and the total bandwidth consumption, respectively, achieved by various replication strategies under different scenarios.

6.1. Scenario one

In this experiment, the performance of three replication strategies is evaluated under the assumption that the probability of requesting any of the replicas is the same. From scenario one of Figs. 2 and 3 it can be seen that the EFS strategy is the superior one. This is because, when the node's storage is full, the EFS strategy replaces a group of replicas with the requested replica only if it is found that it is more important than that group. In the other two strategies, when the storage is full, a group of existing replicas must be replaced with the requested replica even if the requested replica is less important that the replaced group of replicas. Thus, any replica in the replaced group might be requested in the near future and in this case the node must bring it from another node which takes time and consumes the valuable bandwidth.

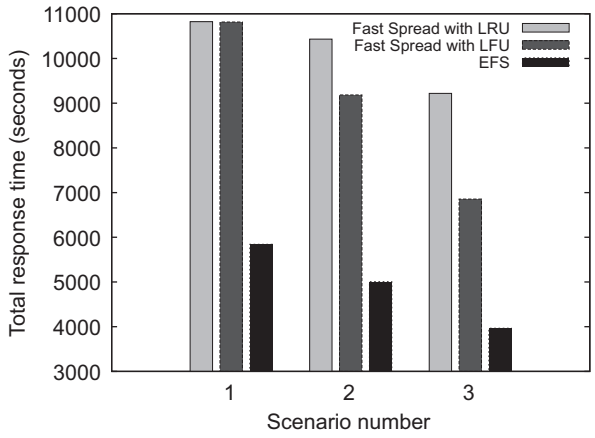

Fig. 2. Total response time achieved by three replication strategies under three different scenarios.

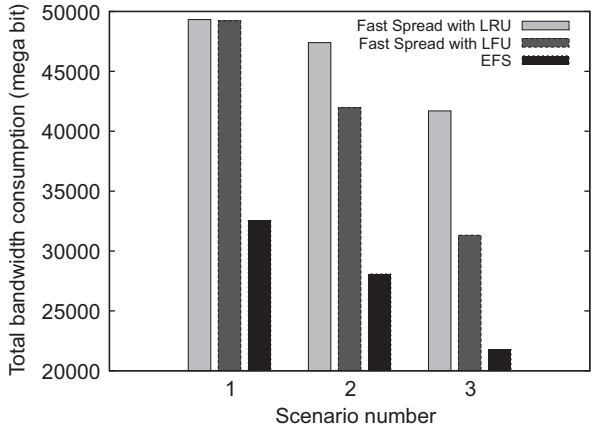

Fig. 3. Total bandwidth consumption achieved by three replication strategies under three different scenarios.

Table 4
Total response time reduction achieved by EFS.

Compared with Fast Spread with LRU
Scenario1 46.01%
Scenario2 52.01%
Scenario3 56.98%
Compared with Fast Spread with LFU
Scenario1 45.98%
Scenario2 45.48%
Scenario3 42.14%

Table 5
Total bandwidth consumption reduction achieved by EFS.

Compared with Fast Spread with LRU
Scenario1 34.03%
Scenario2 40.74%
Scenario3 47.81%
Compared with Fast Spread with LFU
Scenario1 33.89%
Scenario2 33.08%
Scenario3 30.51%

6.2. Scenario two

It is assumed in this experiment that the probability of making a request for replicas in MWG is 30%, while the probability of making a request for the rest of replicas is 70%. From scenario two of Figs. 2 and 3, it appears that the performance of all strategies is better than their performance in the previous scenario. The reason is that many of the replicas kept in each node's storage belong to MWG, and these replicas are requested more frequently than the other replicas in other groups. As a result, the probability of finding the requested replicas locally is relatively high if compared with the probability in the previous scenario. It also appears that the improvement in the performance of Fast Spread with LFU and EFS is larger than the improvement in the performance of Fast Spread with LRU. This is because of using NOR in Fast Spread with LFU and EFS to determine the replicas that will be first deleted. Therefore, the nodes keep most of the replicas in those MWG that usually have the largest NOR. EFS achieved the best performance in this scenario too because of the same reason mentioned in scenario one.

6.3. Scenario three

This experiment differs from the previous experiment in the value of probability. In this experiment, the probability of making a request for replicas in MWG is 50%, while the probability of making a request for the rest of replicas is 50%. Scenario three of Figs. 2 and 3 shows that all the strategies achieved the best performance in this scenario. This is because of increasing the probability of requesting a replica in MWG from 30% to 50%. Hence, the possibility of serving the requests locally is higher in this scenario. EFS keeps it superiority in this scenario, too.

Table 4 shows the total response time reduction achieved by EFS, while Table 5 shows the total bandwidth consumption reduction achieved by EFS.

7. Conclusion

In this paper, a new replication strategy named EFS has been presented. EFS strategy keeps only the important replicas while the

other less important replicas are replaced with more important replicas. EFS uses a dynamic threshold that determines if the requested replica should be stored at each node along its path to the requester. The performance of this strategy has been compared with Fast Spread with LRU and Fast Spread with LFU by event-driven simulations with different scenarios. The evaluation shows that EFS is the best strategy in all scenarios in terms of total response time and total bandwidth consumption. EFS strategy achieved a reduction in total response time between 46.01% and 56.98% when compared with Fast Spread with LRU, while it achieved a reduction in total response time between 42.14% and 45.98% when compared with Fast Spread with LFU (refer to Table 4). On the other hand, EFS strategy achieved a reduction in total bandwidth consumption between 34.03% and 47.81% when compared with Fast Spread with LRU, while it achieved a reduction in total bandwidth consumption between 30.51% and 33.89% when compared with Fast Spread with LFU (refer to Table 5).

In future work, there are two main areas of consideration: considering more factors to determine the importance of different replicas, and undertaking further experimental investigations.

References

Cameron DG, Millar AP, Nicholson C, Carvajal-Schiaffino R, Stockinger K, Zini F. Analysis of scheduling and replica optimisation strategies for data grids using Optorsim. Journal of Grid Computing 2004;2(1):57–69.

Chang R, Chang H. A dynamic data replication strategy using access-weights in data grids. The Journal of Supercomputing 2008;45(3):277–95.

Chang R, Chang H, Wang Y. A dynamic weighted data replication strategy in data grids. In: AICCSA '08: proceedings of the 2008 IEEE/ACS international conference on computer systems and applications. Washington, DC, USA: IEEE Computer Society; 2008. p. 414–21.

Cibej U, Slivnik B, Robic B. The complexity of static data replication in data grids. Parallel Computing 2005;31(8):900–12.

Dong X, Li J, Wu Z, Zhang D, Xu J. On dynamic replication strategies in data service grids. In: ISORC '08: proceedings of the 2008 11th IEEE symposium on object oriented real-time distributed computing. Washington, DC, USA: IEEE Computer Society; 2008. p. 155–61.

Figueira S, Trieu T. Data replication and the storage capacity of data grids. Berlin, Heidelberg: Springer-Verlag; 2008. pp. 567–75.

Hong L, Xue-dong Q, Xia L, Zhen L, Wen-xing W. Fast cascading replication strategy for data grid. In: CSSE '08: proceedings of the 2008 international conference on computer science and software engineering. Washington, DC, USA: IEEE Computer Society; 2008. p. 186–9.

Horri A, Sepahvand R, Dastghaibyfard G. A hierarchical scheduling and replication strategy. IJCSNS International Journal of Computer Science and Network Security 2008;8(8).

Lamehamedi H, Szymanski B, Shentu Z, Deelman E. Data replication strategies in grid environments. In: Proceedings of the fifth international conference on algorithms and architectures for parallel processing. IEEE Computer Society Press; 2002. p. 378–83.

O'Neil E, O'Neil P, Weikum G. The LRU-K page replacement algorithm for database disk buffering. In: Proceedings of the 1993 ACM SIGMOD international conference on management of data. ACM; 1993. p. 297–306.

Park S, Kim J, Ko Y, Yoon W. Dynamic data grid replication strategy based on internet hierarchy. In: Second international workshop on grid and cooperative computing, 2003. p. 838–46.

Prischepa V. An efficient web caching algorithm based on LFU-K replacement policy. In: Proceedings of the spring young researchers colloquium on database and information systems. IEEE; 2004. p. 23–6.

Ranganathan K, Foster I. Design and evaluation of dynamic replication strategies for a high-performance data grid. In: International conference on computing in high energy and nuclear physics. Beijing, China, 2001a.

Ranganathan K, Foster I. Identifying dynamic replication strategies for a high-performance data grid. In: GRID '01: proceedings of the second international workshop on grid computing. London, UK: Springer-Verlag; 2001. p. 75–86.

Rasool Q, Li J, Oreku G, Munir E. Fair-share replication in data grid. Information Technology Journal 2008;7(5):776–82.

Tang M, Lee B, Yeo C, Tang X. Dynamic replication algorithms for the multi-tier data grid. Future Generation Computer Systems 2005;21(5):775–90.

Wu J, Lin Y, Liu P. Optimal replica placement in hierarchical data grids with locality assurance. Journal of Parallel and Distributed Computing 2008;68(12):1517–38.

Zhao W, Xu X, Xiong N, Wang Z. A weight-based dynamic replica replacement strategy in data grids. In: APSCC '08: proceedings of the 2008 IEEE Asia-Pacific services computing conference. Washington, DC, USA: IEEE Computer Society; 2008. p. 1544–9.